UNIVERSITY OF TORONTO

INSTITUTE FOR AEROSPACE STUDIES

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

Simplex

Team: #48

Members:

Xi Chen	995249431
Tianle Dai	995398430
Sida Wang	995324037

Course:	Engineering Design	AER201S
Instructor:	Professor M. R. Emami	
TA:	Bardia Bina	
Due Date:	April 16, 2008	

Simplex

Team: #48

Members:

Xi Chen	995249431
Tianle Dai	995398430
Sida Wang	995324037

Course:	Engineering Design	AER201S
Instructor:	Professor M. R. Emami	
TA:	Bardia Bina	
Due Date:	April 16, 2008	

ACKNOWLEDGEMENTS

We received enormous support from many people in this project. We would like to especially acknowledge the support of Prof. M.R. Emami, TA Bardia Bina and the owner of Creatron Inc. for their quality advice throughout the design and development process.

We owe the success of this project to the class of EngSci 1T0, many groups and individuals generously shared with us their experience and expertise in many aspects of the construction process. Special thanks go to Yuheng Huang, Xiaoyin Liu, Qiuchen Zhang, Lei Wu, Ze Wang, Tang Tang and He Huang. This list is far from exhaustive for such a list would probably be too long to include.

Last but not least, we acknowledge the enormous support we received from our parents, for tolerating the mess we made during the process and for tolerating the noises late at night.

ABSTRACT

Simplex is a robotic prototype that was designed and built over the past 4 months to solve the closet light inspection problem. The closet light inspection problem calls for a machine that can quickly and reliably determine and display the functionality and position of closet lights placed on an egg tray. This report contains the complete design and construction process, detailed description of the final prototype, suggestions for further improvements and a variety of related information such as budget, schedule, operating procedures etc.

Simplex takes a tray of closet lights. It inspects the tray row-by-row. It turns the lights on and off using an arm that cover all possible positions. Both functionality and position sensors are implemented using the same 5 phototransistors. The horizontal motion of the testing arm is powered by a stepper motor for accuracy and the vertical motion of the testing arm is powered by a DC motor for power.

The finished machine performed fairly well during the demo. The machine correctly determined the positions of 5 out of 5 lights and correctly determined the functionalities of 4 out of 5 lights in each of the two trials in 40 seconds and 30 seconds. The machine is fairly reliable since it went through about 15 successful full runs and over 100 test runs before the demo.

However, Simplex is not perfect; it often passes lights that have only one or two bright LED(s). Many suggestions for improvements are given in this report.

TABLE OF CONTENTS

SYMBOLS AND ABBREVIATIONS

Symbol or Abbreviations	**Explanation**
A/D	Analog to digital
ADC	Analog to digital converter
IR	Infrared
IC	Integrated circuit
LCD	Liquid crystal display
PERT	Project evaluation & review technique
LED	Lighting-emitting diode
GANTT	Chart developed by Henry GANTT
PIC	Peripheral interface controller
DC Motor	Direct current motor
τ	Torque
ω	Angular frequency
V	Voltage
I	Current
TN i.e. T10, T20, T40 etc.	Gear with N teeth

1 INTRODUCTION

Simplex is a robotic prototype that was designed and built over the past 4 months to solve the closet light inspection problem. The closet light inspection problem calls for a machine that can quickly and reliably determine and display the functionality and position of closet lights placed on an egg tray [citation].

Simplex is a simple machine that solves the complex problem of closet light inspection. Simplex takes a tray of closet lights. It inspects the tray row-by-row. It turns the lights on and off using an arm that cover all possible positions. Both functionality and position sensors are implemented using the same 5 phototransistors. It takes about 20 seconds to load the tray and it has a user-friendly interface. The microcontroller, the circuits, the sensors and two motors are all placed on a testing arm that moves row-by-row. The horizontal motion of the testing arm is powered by a stepper motor for accuracy and the vertical motion of the testing arm is powered by a DC motor for power.

Our machine has the following features that distinct itself from others,

- Simple circuit system, no logic elements used, no op-amps used
- Customize made PIC proto64 board
- Single phototransistor for both position by infrared reflection and functionality detection by intensity threshold
- Continuous operation despite row-by-row design (i.e. the starting position of the testing arm for the second run is the end position of the first run)
- Hold up to 12 logs with date and time accurate to seconds
- PIC is powered by 6 x 1.5V battery
- Isolated testing environment, black box. Operates under any external lighting condition
- Intuitive display of results in the form of a matrix,

  XXXXP
  XXXXX
  XXFXX
  XXXPX

  where P stands for pass, X for empty, F for fail.

The first section of the report contains important information about the overall machine. Including a description of the overall machine, the standard operating procedures of the machine, an evaluation of the criteria and task division.

The majority of the rest of this report is then divided into its three subsystems, namely electromechanical, circuits and PIC. Each section contains DETAILED information about the subsystem of concern including problem assessment, detailed description of the solution, supporting calculations, computer programs and results. Many details are also included in the appendix.

The rest of this report contains information about how it is integrated, how task scheduling worked as well as information on budget and how it can be improved.

Overall, this report contains the complete design and construction process, detailed description of the final prototype, suggestions for further improvements and a variety of related information such as budget, schedule, operating procedures etc.

2 DESCRIPTION OF THE OVERALL MACHINE

FIGURE 2-1, MACHINE OVERVIEW

FIGURE 2-2, WITH THE TOP LID OPEN

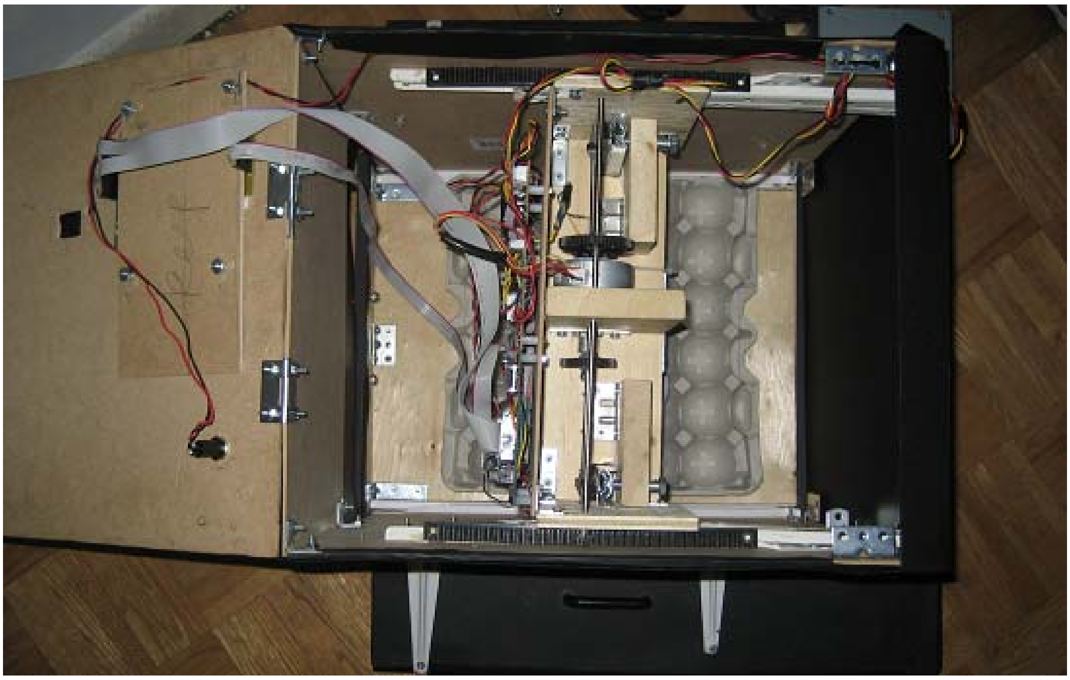


FIGURE 2-3, TOP VIEW WITH THE LID OPEN

The top lid can be opened to give access to internal machinery. The two thick ribbon cables connect the LCD and keypad to the PIC on the testing arm.

2.1 FEATURES

- Simple circuit system, no logic elements used, no op-amps used
- Customized PIC proto64 board
- Single phototransistor for both position by infrared reflection and functionality detection by intensity threshold
- Continuous operation despite row-by-row design (i.e. the starting position of the testing arm for the second run is the end position of the first run)
- Hold up to 12 logs with date and time accurate to seconds
- PIC is powered by 6 x 1.5V battery
- Isolated testing environment, black box. Operates under any external lighting condition
- Intuitive display of results in the form of a matrix,

```
XXXXP
XXXXX
XXFXX
XXXPX
```

where P stands for pass, X for empty, F for fail.

2.2 DESCRIPTION

Simplex takes a tray of closet lights. It inspects the tray row-by-row. It turns the lights on and off using an arm that cover all possible positions. Both functionality and position sensors are implemented using the same 5 phototransistors. The microcontroller (PIC16F877), the circuits, the sensors and two motors are all placed on a testing arm that moves row-by-row. The horizontal motion of the testing arm is powered by a stepper motor for accuracy and the vertical motion of the testing arm is powered by a D.C motor for power.

2.3 KEY INFORMATION

2.3.1 KEY QUANTITATIVE PARAMETERS

Parameter	Value	Unit	Comments
Weight	9	kg	
Operation Time	10~54	Sec	Depending on the rows of light
Cost	142.85	CDN$	Depends on the supplier. This estimate is based on unit price of [citation] for a min-order of 1.
Max Dimension	40x65x55	cm^3	
Height	40	cm	
Maximum Length	65	cm	Length of bottom board
Maximum Width	55	cm	Width of bottom board
Box Length	42	cm	Length of machine body
Box Width	35	cm	Width of machine body
Volume	~0.06	m^3	Volume of the box
AC RMS Voltage	110	V	
AC frequency	60	Hz	
Setup Time	85	sec	As timed on the demo by officials. Basic setup time + time to check the if lights are horizontal
Basic Setup time	10	sec	Time needed to load the tray

2.3.2 KEY QUALITATIVE INFORMATION

Name	Description
Horizontal Drive	1 x 12V Stepper motor driven by L293D. Rack and pinion mechanism.
Vertical Drive	1 x 12V 100rpm D.C. Motor driven by H-bridge made by TIP142 (Appendix C3) and TIP147 (Appendix C4). Rack and pinion mechanism.
Microcontroller	PIC16F877 on customized PIC proto64 board.
Material	Spruce STDRD plywood for bottom board, Pine wood for testing arm frame, thin 1/8'' composite board for the shell, black paper board on the outside
Functionality Sensor	PT481 Phototransistor (Appendix C2) driven by custom sensor driver. Intensity threshold.
Position Sensor	Infrared reflection position sensing by a combination of LTE5208 Infrared Diode (Appendix C2) and PT481 phototransistor (Appendix C2) driven by custom sensor driver. Intensity threshold.

2.4 THE TESTING ARM

The testing arm is our most important component. All the motors, circuits and control systems are all on this arm! (with the exception of LCD, keypad and power supply)

2.4.1 SIDE VIEW

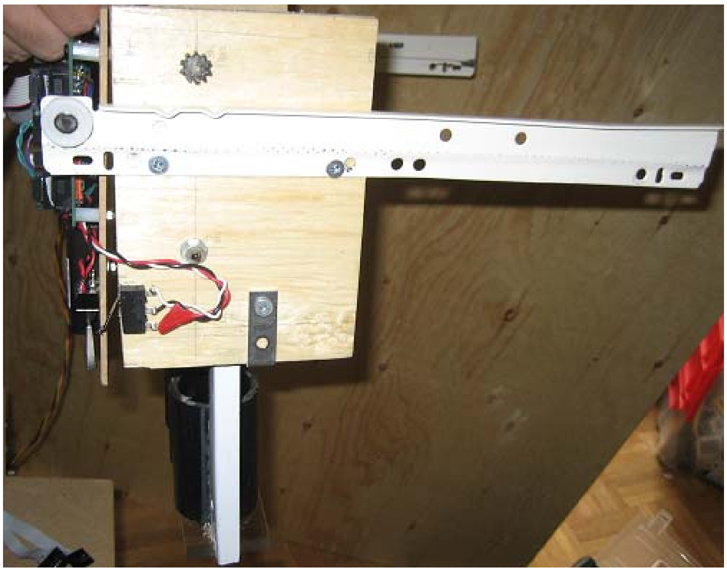


FIGURE 2-4 SIDE VIEW

2.4.2 BACK VIEW

FIGURE 2-5 BACK VIEW

2.4.3 BOTTOM VIEW

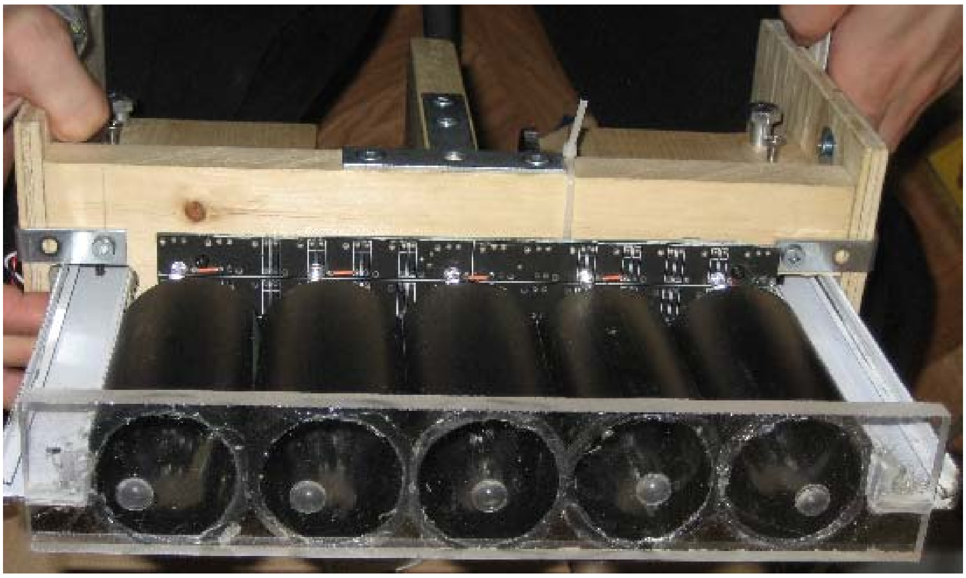


FIGURE 2-6 BOTTOM VIEW

2.4.4 TOP VIEW

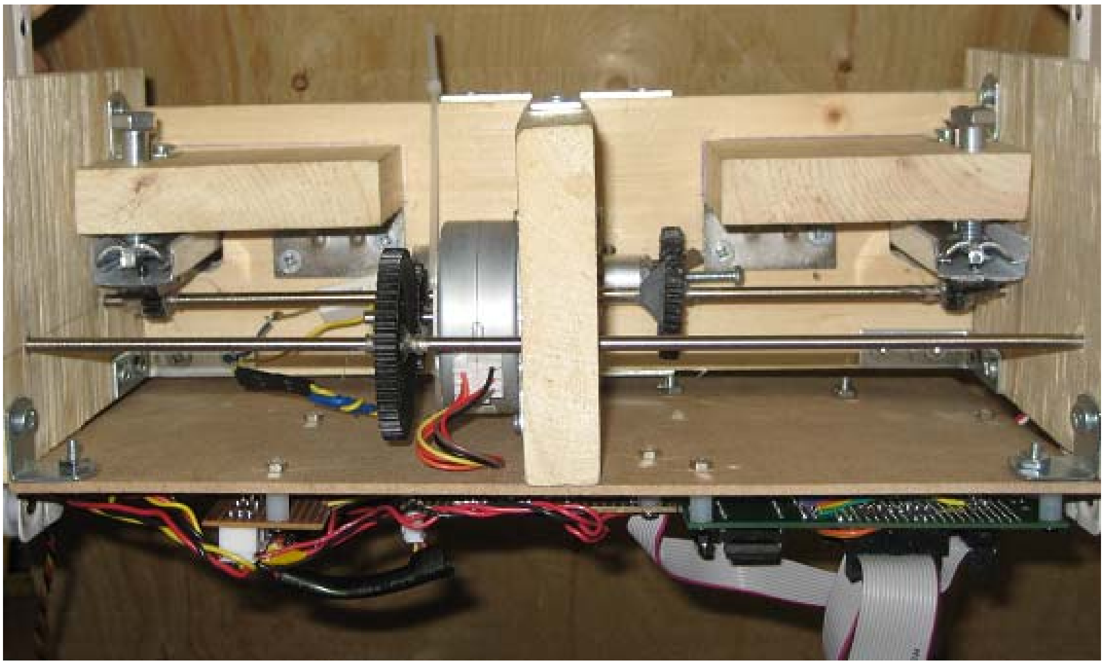


FIGURE 2-7 TOP VIEW

2.4.5 FRONT VIEW

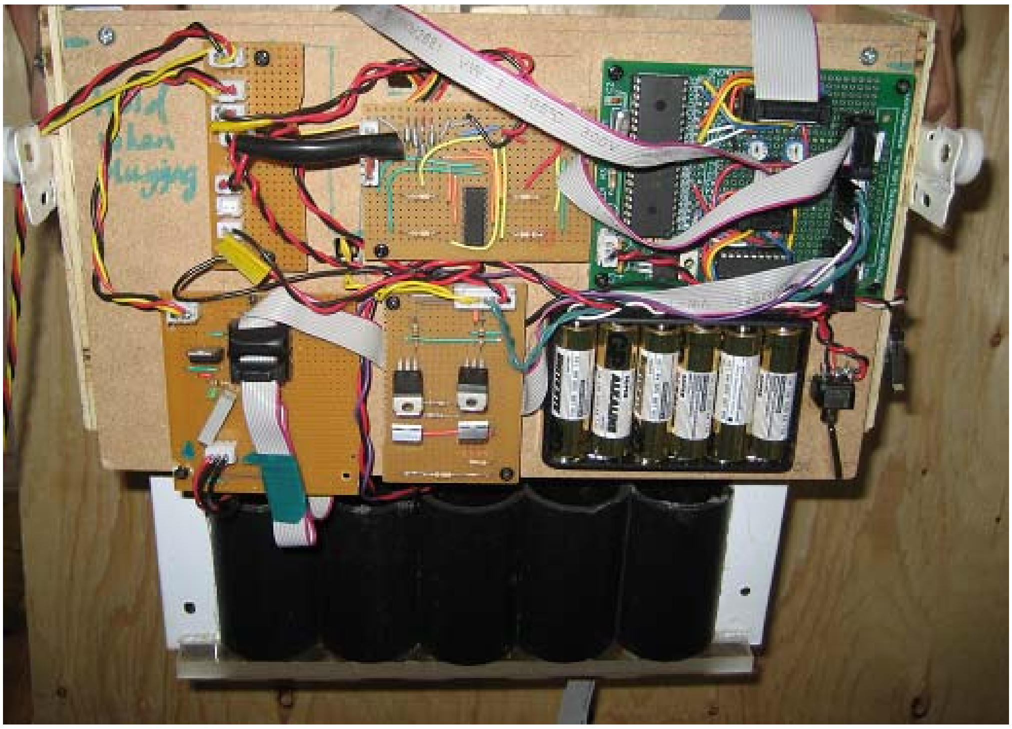


FIGURE 2-8 FRONT VIEW

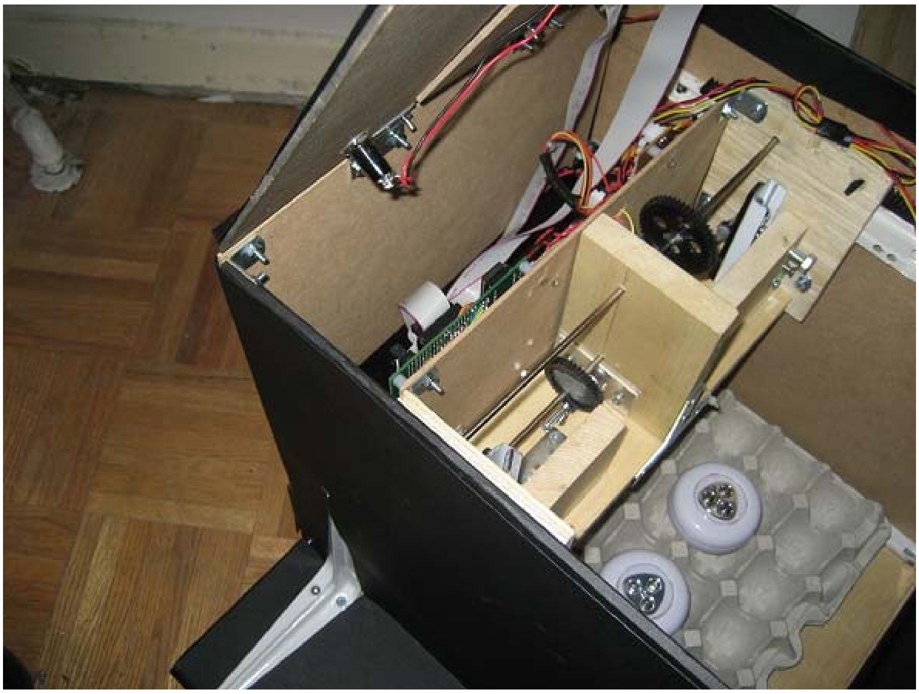


FIGURE 2-9, TESTING ARM IN THE MACHINE

2.5 CIRCUIT AND CONTROL

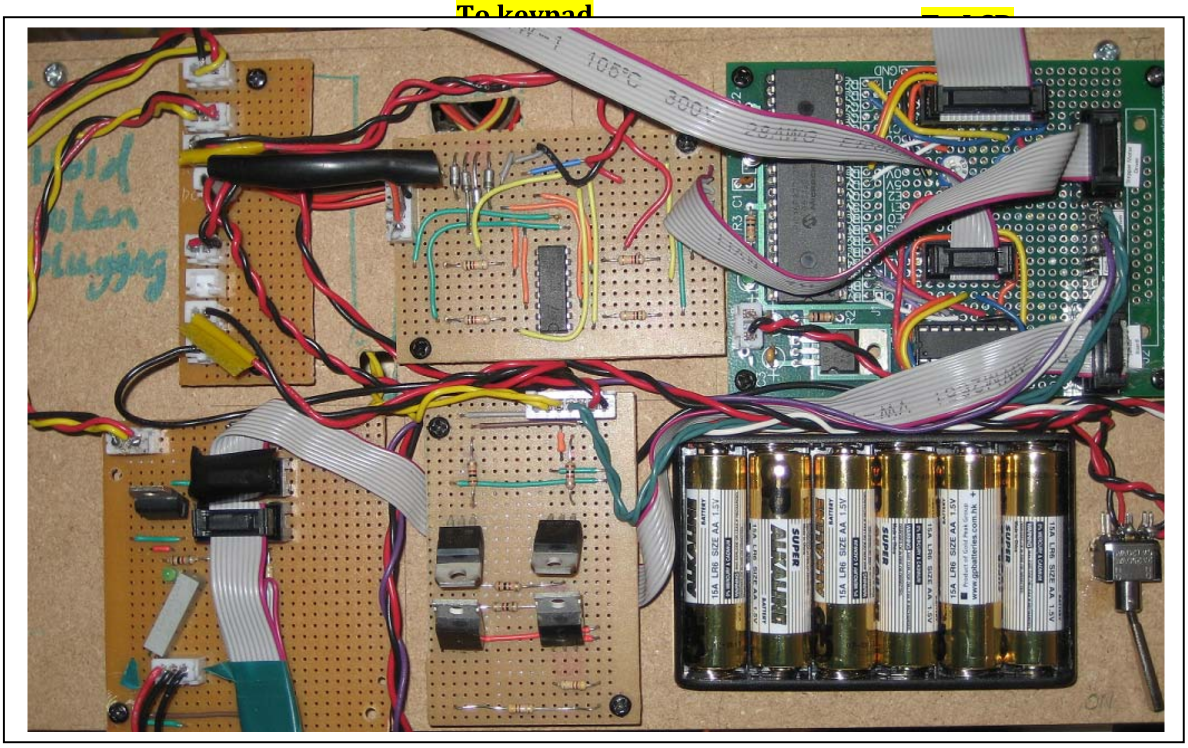


FIGURE 2-10, CIRCUIT AND CONTROL SYSTEM

Shown above is the entire circuit in Simplex. Refer to circuit section for details. The PIC microcontroller is mounted on its own board customly made for this project it is powered by 6 batteries.

2.6 MECHANISMS

FIGURE 2-11, STEPPER GEAR SYSTEM FOR HORIZONTAL MOTION

The stepper motor rotates a gear with 30 teeth, which turns a gear of 50 teeth and ultimately contacts the rack with a 10 teeth pinion (pinion is just another small gear).

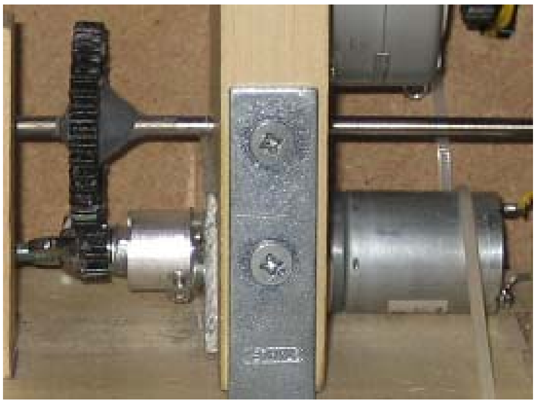


FIGURE 2-12 DC MOTOR FOR VERTICAL MOTION

The D.C. motor powers a gear with 10 teeth which then powers a gear with 40 teeth and ultimately contact the rack with a 10 teeth pinion (pinion is just another small gear).

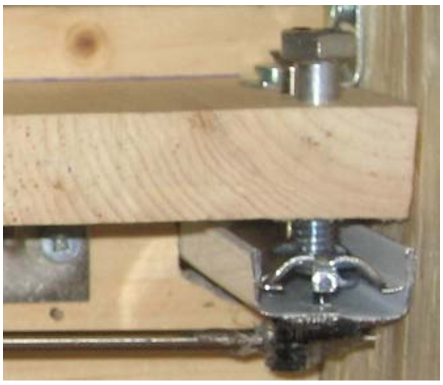


FIGURE 2-13 ADJUSTABLE VERTICAL MOTION PINION AND RACK VIEWED FROM TOP

2.7 CONTROL PANEL

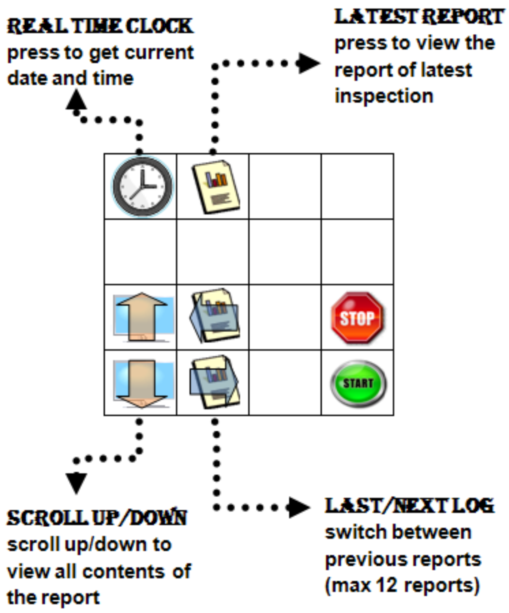


FIGURE 2-14, CONTROL PANEL INSTRUCTION

The control panel of the machine is relatively straight forward. The log contains detailed information of the results of previous runs; including statistics (# light, # pass and #fail), detailed locations, start time, end time and duration. Scroll up and down are used to view the many lines of each log and last/next log is used to switch between past logs. The stop button here is for emergency stop while another hard emergency stop is also provided.

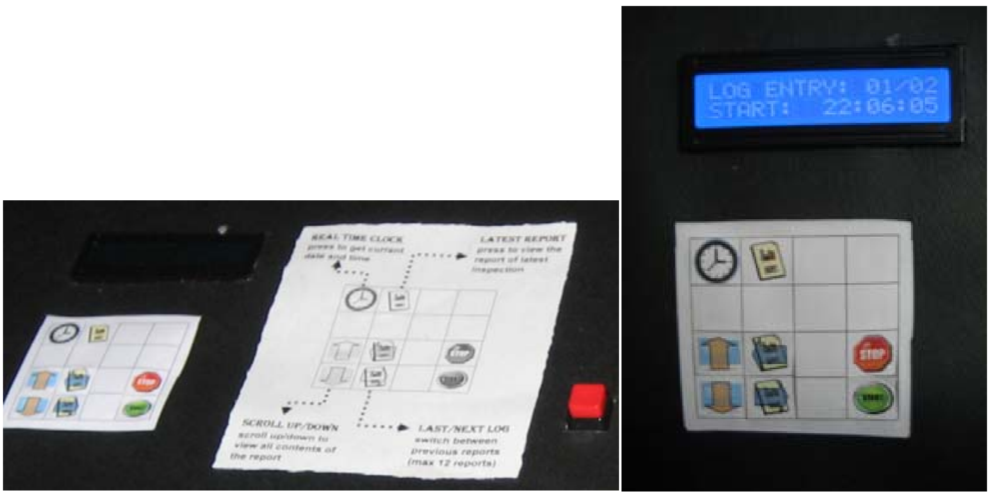


FIGURE 2-15, CONTROL PANEL

3 PROBLEM DIVISION

The development process was first separated into 3 components, namely the microcontroller, the electrical circuits and the mechanical system.

All members participated in the design process of every components. For example, all three members participated in the design of the mechanical parts. The circuit design was developed in close relation with the PIC pin layout and sometimes the design of the program was modified to accommodate the circuit

In terms of fabrication, the task division is as follows,

3.1 ELECTROMECHANICAL COMPONENTS

Task	Member(s)
Cutting, sawing and other machine shop tasks	Electromechanical
Drilling, screwing, assembling electromechanical parts	Electromechanical, Circuit
Drawing	PIC, Circuit
Shopping	All
Design	All

3.2 CIRCUIT COMPONENTS

Task	Member(s)
Soldering	All
Layout	Circuit, PIC
Drawing	PIC, Circuit
Shopping	All
Design	Circuit and PIC

3.3 PIC

Task	Member(s)
Program Design	PIC
Program Implementation	PIC

Members of this team never worked in isolation and the role divisions were never rigid. The electromechanical member helped with circuit components, the circuit member assembled some of the electromechanical parts and the microcontroller member helped both the circuit member and the electromechanical member. There was a lot of communication throughout the process and each member was aware of what others are doing.

4 PERSPECTIVE

4.1 THEORY/SURVEY

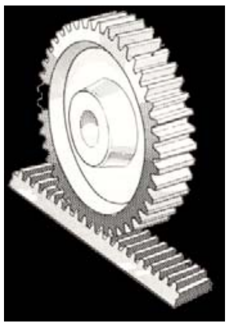
Currently, closet lights have been more and more frequently used in modern housing. As people tends to place more and more of their items into their closet for long term storage, the demand for closet light has increased over time. Especially for walk in closets, these lights are almost essential. In response to the high demand, companies and factories are always coming up with new and more efficient closet lights to meet the demand, because of this, new methods to quality check the closet lights are also in demand. Our machine fits the purpose of checking a cylindrical closet light with a thickness of 25±1mm and a varying diameter from 63±2mm to 73±2mm. This push light consists of 3 LEDs in the middle of a round dome. The push light is usually used for closet and cupboards. [9] This type of light generates very little heat, and it has a battery life up to 10 times longer than normal bulbs, and it can be attached to virtually any surface with a double sided adhesive pad. [9] This product is currently sold by many companies, including Lifemax, a company that provides electronics in Hong Kong for over 30 years. The manufacturer is based at a group factory in Dong Guan, China.

There are 3 components required in the design of this machine. …

1. A method for accurate and consistent horizontal movement of the robotic arm
2. A strong and reliable method to press the lights on the row of the egg tray
3. A reliable method to determine the brightness of the light emitted from the closet lights

Generally a good reliable method of controlling the accurate position of the required horizontal motion is by using stepper motor. A stepper motor is a brushless, synchronous electric motor that divides one rotation into a large number of steps. With the correct voltage provided to the terminals, the stepper motor will be activated to move one step at a time.

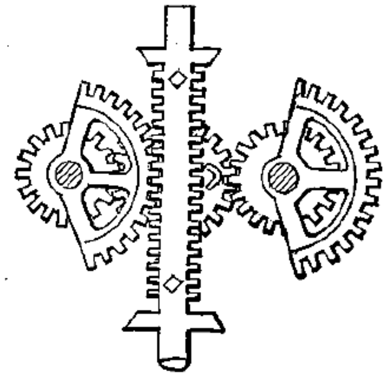
A good method to transfer the force from the stepper motor to horizontal motion is through a rack pinion system. The main purpose of a rack and pinion system is to transform rotary motion to linear motion and vice versa. [10]



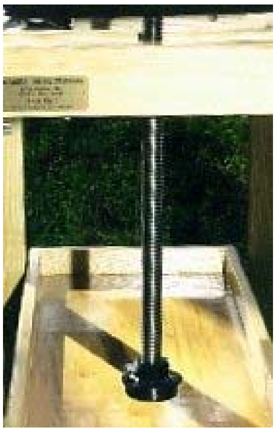
4-1 RACK AND PINION SYSTEM

The most common method of providing mechanical energy is through a DC motor, compared to a stepper motor, DC motor is less accurate in terms of precision of the distance travelled. However it will be able to generate much more power for the required task.

In order to convert the rotary motion from the DC motor to a vertical straight forward motion, there are many alternatives. As discussed previously, a variation of the gear pinion system can be used, as shown in the diagram below (to the left). Also, a more slow but power alternative would be the acme screw, a diagram of the acme screw is also shown below (to the right).



4-3 RACK AND PINION APPLIED TO VERTICAL MOTION[11]



4-2 ACME SCREW[12]

There are three types of light sensors that are available, photoresistors, phototransistors, and photodiodes.

Photoresistors are variable resistors depending on the light hitting the surface of the resistor. They can be calibrated to be very sensitive to light, however it is slow in terms of detection. It takes photoresistors a few milliseconds to respond to light but a few second to return back to its normal state.

Phototransistor is a transistor with its base lead replaced by a light sensitive surface. When there is no light hitting the phototransistor, it acts as a open circuit. However, when some light hits the surface of the phototransistor, it will generate a small base current which will allow a big collector to emitter current through the two remaining leads.

The idea behind photodiodes is to convert light energy directly into electric current. When a photon of sufficient energy hits the diode, it will excite the electrons causing current to flow. It is able to act as a small current source when there is lighting. They are most commonly used to detect fast infrared light pulses.

4.2 HISTORY

Throughout history, quality control has always played important. At ancient times, civil engineers spent a great amount of time to check if the building materials were up to the standard. For example, one of the specifications that were imposed in the early years was the four sides of the base of the Great Pyramid of Giza perpendicular to within 3.5 arc-seconds. Nowadays, buyers always tend to buy equipments that are more reliable and want reassurance when spending money. Companies are forced to provide this reassurance to the buyers to ensure their stance in the competitive market. As new products come into the market for mass production, engineers are often asked to design an automated system that inspects the new product to ensure its quality before the product reaches the market. As of today, RFPs relating quality control becomes more and more frequent in the market.

5 OBJECTIVES CONSTRAINS AND ACCEPTANCE CRITERIA

5.1 STATEMENT OF PROBLEM

The proposal addresses the RFP for a closet light test machine. At most 6 closet lights will be placed inside an egg-tray at 20 possible positions in the upward facing orientation. Each closet light has 3 LEDs and is powered by 3 batteries. Several situations might occur for each light,

A. All 3 LEDs light up when the button is pressed and remains on when the button is released.
   a. All 3 LEDs turn off when the button is pressed and released a second time.
   b. One or more LED does not turn off
B. Otherwise

Where A. and a. represent functional closet lights and B. and b. represent dysfunctional lights.

A machine is required to quickly and reliably determine whether the closet lights placed on an egg tray are functional or not. The machine should be able to determine the positions of both functional and dysfunctional lights on the egg tray. The machine should also be able to calculate its operation time.

5.2 GOALS AND OBJECTIVES

The goal of this design project is to build a closet light inspection machine such that meets all the design constrains and gets most evaluation points, as specified in the RFP.

5.2.1 CRITERIA AND OBJECTIVES

- $\text{Reliability} = \frac{\text{\# of lights whose functionality \& } location\ is\ successfully\ determined}{\text{total \# of lights inspected}}$, the reliability should be as high as possible, ideally it is 1.
  - Objective: Reliability> 0.8
- $\text{Runing Rate} = \frac{\text{\# of lights inspected}}{\text{total running time}}$, the running rate should be as high as possible.
  - Objective: Running Rate > 3 light / min
- $\text{Light Counter Reliability} = \frac{\text{\# of runs whose total \# of lights are correctly counted}}{\text{total \# of runs}}$, the light counter reliability should be as high as possible, ideally it is 1.
  - Objective: Counter Reliability>0.95
- $\text{Runtime Clock Error} = \left|1 - \frac{\text{machined measured running time}}{\text{real running time}}\right| \times 100\%$, the runtime clock error should be as small as possible.
  - Objective: it should be controlled to be less than 5%.
- $\text{Deactivation Failure Rate} = \frac{\text{\# of good lights that areleft on after inspection}}{\text{Each Run}}$, the deactivation failure rate show be as low as possible, ideally 0.
  - Objective: Deactivation Failure Rate ≤ 0.1

- Setup Time = time to setup (initialization and/or calibration) the machine, the setup time should be at little as possible.
    - Objective: This time should never exceed 2 minutes.
- Egg Tray Capacity = # of egg trays can be loaded to the machine every time, the egg tray capacity should be as high as possible.
    - Objective: Egg Tray Capacity = 1
- Log Length = total entries of log the machine can hold, the log length should be as long as possible.
    - Objective: Log Length ≥ 5
- Weight = total weight of the machine, the weight should be as low as possible.
- Backup Power, Boolean value.
    - Objective: achieve it
- Robustness Rating = the number of successive inspections that the machine can handle without any adjustments
    - Objective: Robustness Rating ≥ 5

Among these objectives, reliability and running rate are most crucial. The machine must be check more than 1 light/min to qualify. Light counter reliability, runtime clock error and deactivation failure rate are the second group of most important objects. The rest objectives are aimed at extra features.

6 BUDGET

Item	Descriptions	Location Used	#	Price	Total
TIP147T(transistor)	TO220	DC Driver	2	$0.96	$1.92
Resistor	470 1/8W	DC Driver/sensor	3	$0.01	$0.03
TIP142T(transistor)	TO220	DC Driver/Sensor	3	$1.10	$3.30
black connector	2 pins male	DC Motor	1	$0.28	$0.28
Ceramic resistor	12Ω 5W	DC Motor	1	$0.40	$0.40
DC Motor	Zheng Gearhead 100rmp	DC Motor	1	$8.00	$8.00
DC Motor Coupler		DC Motor	1	$2.00	$2.00
black connector	2 pins female	DC Motor/feedback	4	$0.15	$0.60
white connector	7 pins pair	DC/Stepper Driver	2	$0.40	$0.80
Resistor	1K 1/8W	DC/Stepper Driver/sensor	9	$0.01	$0.09
Resistor	22K 1/8W	feedback	3	$0.01	$0.03
Switch	0.5cm small push button	feedback	1	$0.22	$0.22
Switch	2cm microswitch	feedback	1	$1.20	$1.20
card board	black color 40"*26"	frame	2	$2.32	$4.64
Conner (L-shaped)	5"	frame	2	$0.99	$1.98
Conner (L-shaped)	8"	frame	2	$1.29	$2.58
Drawer track	40cm pair	frame	1	$3.99	$3.99
Handle	black plastic, pair	frame	1	$2.00	$2.00
handle	bronze	frame	1	$2.00	$2.00
Screws with nuts	3/16*1/2"	frame	8	$0.05	$0.40
thin wood	2'*2'*1/8"	frame	2.5	$1.35	$3.38
battery box	6 AA batteries	PIC	1	$1.95	$1.95
bus connector	16 pins pair	PIC	1	$0.35	$0.35
Capacitor	22pF	PIC	2	$0.05	$0.10
Capacitor	0.1µF	PIC	2	$0.05	$0.10
Capacitor	0.33µF	PIC	1	$0.05	$0.05
Capacitor	1µF	PIC	1	$0.05	$0.05
Crystal oscillator	10MHz	PIC	1	$0.80	$0.80
Green LED		PIC	1	$0.20	$0.20
L7805(power regulator)	TO220	PIC	1	$0.60	$0.60
LCD& keypad	LCD: 16×2, Keypad: 16keys	PIC	1	$6.00	$6.00
male stand	-+---	PIC	8	$0.01	$0.09
MM74C922N		PIC	1	$6.20	$6.20
PIC Board	PIC Proto64	PIC	1	$10.00	$10.00

PIC16F877		PIC	1	$10.00	$10.00
potentiometer	0-20KΩ	PIC	2	$0.30	$0.60
Resistor	10K 1/8W	PIC	3	$0.01	$0.03
socket	14 pins	PIC	1	$0.08	$0.08
socket	40 pins	PIC	1	$0.30	$0.30
white connector	2 pins pair	power	5	$0.11	$0.55
PC Power supply		power supply	1	$4.95	$4.95
white connector	3 pins pair	power/sensor	5	$0.15	$0.75
LTE4208(IR Emitter)		Sensor	5	$0.65	$3.25
Resistor	100 1/8W	sensor	2	$0.01	$0.02
TP481(phototransistor)		Sensor	5	$0.88	$4.40
bus connector	10 pins pair	sensor/PIC	5	$0.28	$1.40
L293D(motor driver)		Stepper Driver	1	$4.20	$4.20
Zener Diode	15V	Stepper Driver	4	$0.15	$0.60
Stepper Motor	PM55L-048	Stepper Motor	1	$8.00	$8.00
Acrylic plate	5cm (out of 30cm)	testing arm	0.2	$8.00	$1.60
Black Plastic Tube	4'	testing arm	0.5	$2.73	$1.37
Brace (T-shaped)	pairs	testing arm	1	$0.49	$0.49
Circuit Board	3"*4"	testing arm	2	$1.20	$2.40
Conner (L-shaped)	4 holes in a row, package of 4	testing arm	1	$0.49	$0.49
Door track	21cm (out of 110cm) & heads	testing arm	2	$1.14	$2.29
Pine wood	135*90×1, 280*90×1, 93*72×1	testing arm	1	$2.00	$2.00
Plastic Mounting	Package of 4	testing arm	4	$0.85	$3.40
plywood	155*105	testing arm	2	$0.60	$1.20
Screws	11/64*1 1/4"	testing arm	2	$0.27	$0.54
Sensor Board	5cm (out of 30cm)	testing arm	1	$1.00	$1.00
Wood screws	9/64*1/2"	testing arm	32	$0.05	$1.60
Conner (L-shaped)	1 hole on each side, package of 4	testing arm/frame	2	$0.49	$0.98
Conner (L-shaped)	2 holes parallel on each side, package of 4	testing arm/frame	1	$0.49	$0.49
Drawer track	30cm pair	testing arm/frame	1	$3.19	$3.19
plastic pinion	22cm	testing arm/frame	3	$2.00	$6.00
Screws with nuts	1/8*1/2"	testing arm/frame	31	$0.05	$1.55
bumper pad	package of 12	transmission	0.4	$3.69	$1.48
nut	11/64"	transmission	2	$0.12	$0.24
plastic gear set	T10×5, T30×1, T40×1, T50×1	transmission	1	$2.98	$2.98

shaft	11/64"*1'	transmission	2	$1.00	$2.00
washer	11/64"	transmission	2	$0.06	$0.12
				total	$143

TABLE 6-1 BUDGET

The budge depends on supplier. The price listed in the table is retailer price. If large quantity is bought, i.e. mass production of the machine, the price can be much lower.

This design saves budget compare to other design, for the following reasons:

- Customized PIC board is used instead of DevBugger
- Very little elements used in circuit, i.e. no comparator and logic gates
- Cheap wood is used for building frame

7 ELECTROMECHANICAL SUBSYSTEM

7.1 ASSESSMENT OF THE PROBLEMS

In order to keep up with the plan, several components of the robot are required.

1. There must be an outer frame from the robot where the robotic arm will be supported onto. The egg tray will be placed into the frame with the closet lights that needs to be checked.
2. A horizontal moving mechanism is required to move either the egg tray or the robotic arm horizontally. Since we are checking one row of the egg tray at a time. The horizontal mechanism must be able to move forward and backward, and must have high consistency on how far it will move each time.
3. A vertical mechanism is required to move the pressing arm down. The pressing face (IE the material that is required to turn the lights on) must be transparent so that light sensors can receive a reading when the lights are turned on.
4. There must be a convenient method for the closet lights to be placed into the frame. The position of the egg tray that will be placed into the frame should be fixed to reduce probable errors.
5. There must be space allocated inside the frame for circuit boards, since information hiding should be considered and also it will provide protection for the circuits.
6. Since there is a good chance that the closet lights are too close to each other and the light might cause interference, there also needs to be a mechanism to ensure that all the lights emitted are isolated.
7. Since the title of our robot is Simplex, the electrical mechanical parts of the robot should be as simple and elegant as possible.
8. The outer frame of the robot must fit in an 75cm * 75cm *75 cm envelope
9. The max budget of the electromechanical part is around $70, the rest will be given to the microcontroller and circuit member.
10. The weight of the Frame should not pass 7kg.

After considering all these key components of our design it is also required to make the outlook of our robot as aesthetically pleasing as possible. In addition, the machine should be easily disassembled to ensure that the debugging process will go more smoothly and also there will be less trouble when parts of the machine breaks. There should be mechanisms that will make the robot user friendly and durable.

7.2 SOLUTION TO THE PROBLEM

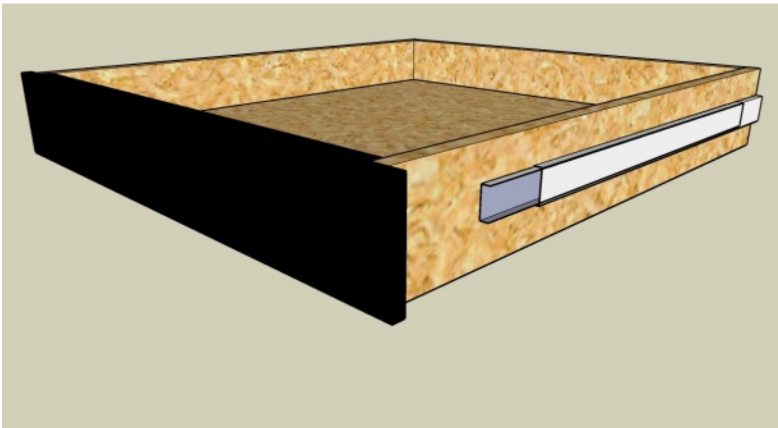
The key highlights to the solution for the electromechanical system are

1. A drawer with an egg tray mounted on top to allow the user to put the corresponding egg tray onto the same position with ease.
2. One arm will be made containing both vertical and horizontal motors; the circuit board will also be attached to this arm.
3. An acrylic plate will be used to press the closet lights. Since the acrylic plate is transparent, the phototransistors will be located on top of the acrylic plate on the

moving arm and still receive accurate reading from the closet lights. (Infrared LED will be located beside the photo transistor to check for the position of the lights)

4. The horizontal motion will be powered by a stepper motor to ensure accurate stepping, the vertical motion will be powered by a DC motor to give the correct about of torque.
5. 5 small cylindrical pipes will be used to isolate the closet lights from each other and avoid interference of the readings received on the photo transistor.

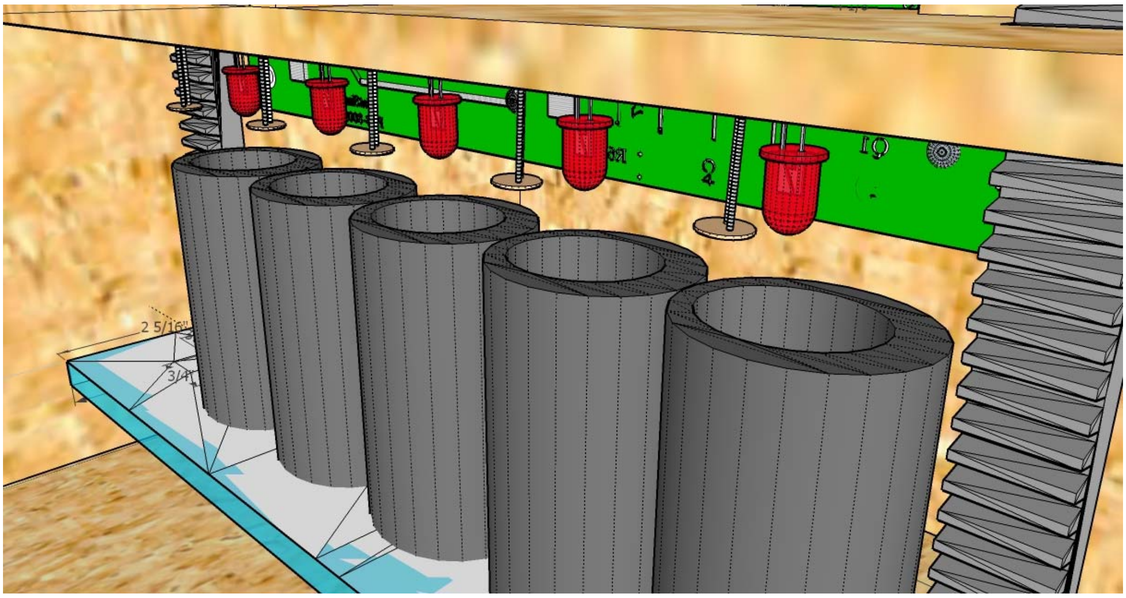
A more detailed explanation of the integration of the electromechanical system will be provided below, diagrams of the robot will also be provided in the following section:



7-1 DRAWER USED TO MOUNT THE EGG TRAY

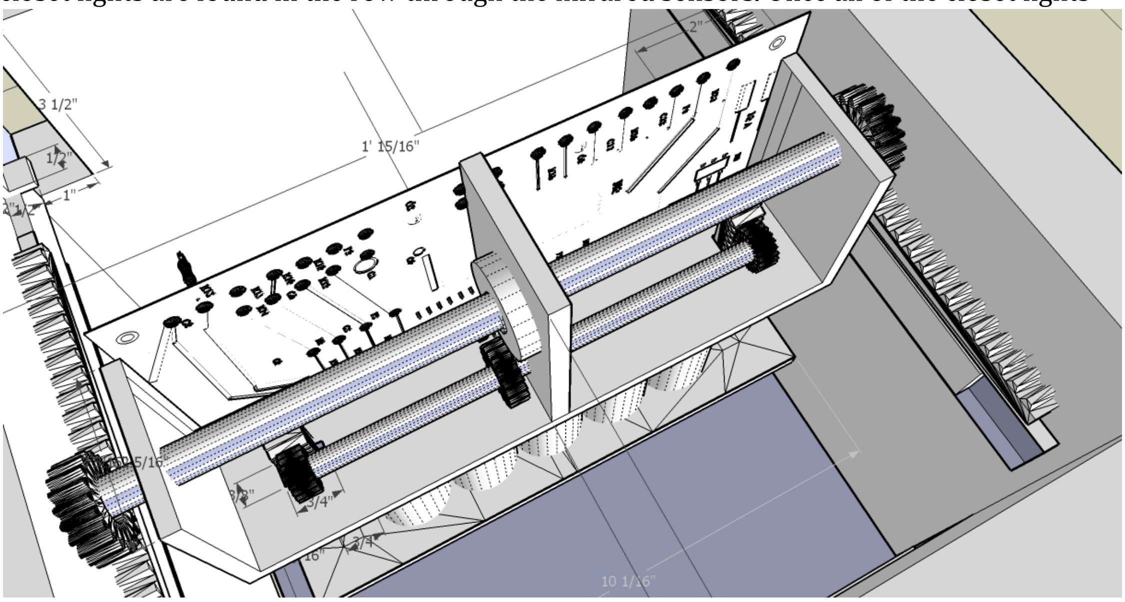
Our general design is very simple. At the bottom of our machine, there will be a drawer (Fig 1), which the user will pull open to stack the egg tray with the closet light onto another mounted egg tray, and then the user can push the drawer fully into the machine and press start to initiate the quality checking process.

There are 5 light sensors mounted above the acrylic plate along with 5 infrared LEDs (Fig 2). The main purpose of the LEDs is to accurately locate the positions for all the closet lights. The position of the LED is located by shining infrared LEDs onto the egg tray, and taking advantage of the shiny surface that is on all of the closet lights, a portion of the light emitted by the infrared LEDs will be reflected back onto the phototransistor located beside the LED on the positions where there is a closet light. Then, the phototransistor will be able to send a signal to the microchip indicating that there is a light at the particular position.



7-2 LIGHT SENSORS AND INFRA RED LEDS

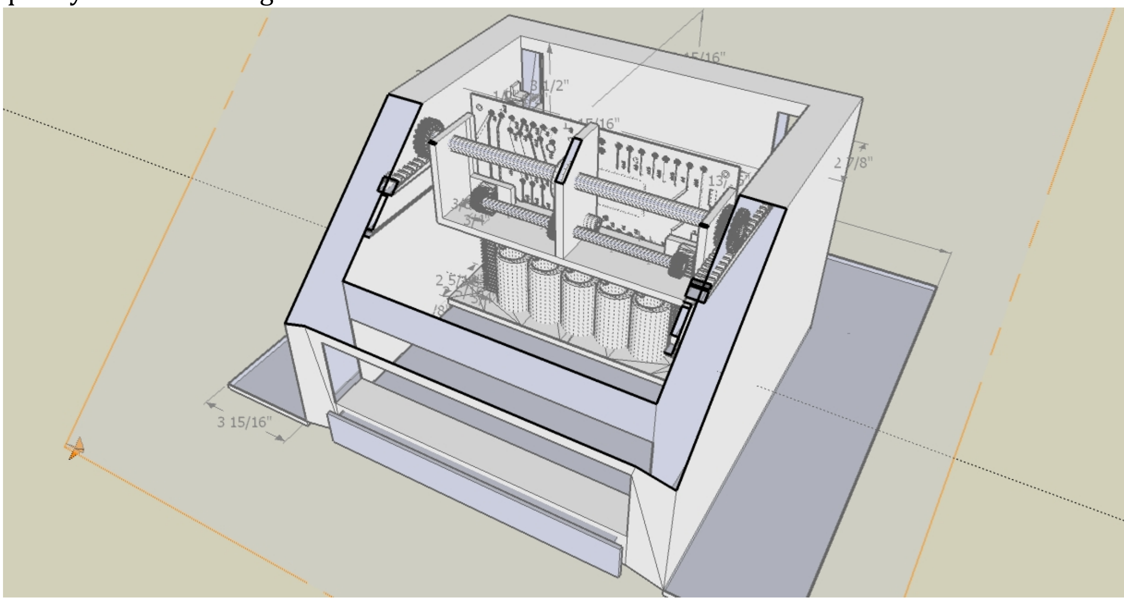
In addition to the lights and phototransistors, our design also consists of two moving parts. First of all, two pinion and rack system are used to turn on the closet lights. One end of the racks is attached onto the rectangular acrylic plate which will be used to turn on the lights; the respective pinions will be attached to one 100 step/minute gearbox motor. When the motor move, it will push the pinion which will force the rack to move down and provide sufficient power to press all the closet lights on that row (7-4). This motion will begin once closet lights are found in the row through the infrared sensors. Once all of the closet lights



have been turned on, the phototransistor will be able to pick up the light emitted on all of

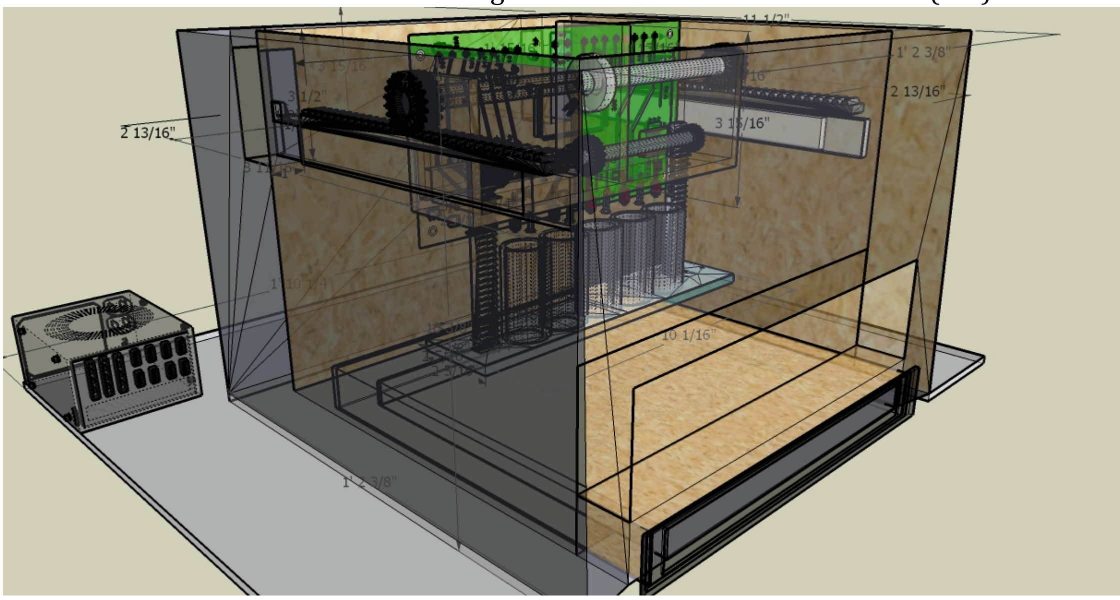
7-3 TOP VIEW OF TESTING ARM, HORIZONTAL MOTION the respective closet lights, and also measure the intensity of the closet lights. Knowing the intensity, the PIC microchip will be able to determine, rather all three LEDS on the closet light are fully functional. The acrylic plate will then turn off the lights on the row and another reading will be taken by the phototransistors to ensure all of the lights are off. Then the pushing arm will continue to the next row.

The second moving part in our prototype is to move the whole testing arm horizontally so that the whole testing process can continue to the next row of the egg tray. This movement is governed by a stepper motor to ensure accurate stepping. Another pinion and rack system will be implemented to make the horizontal motion (Horizontal motion). This stepping motor will only be activated when the current row is finished checking and the quality of the closet lights are determined.



7-4 WHOLE TESTING ARM INTEGRATED IN MACHINE

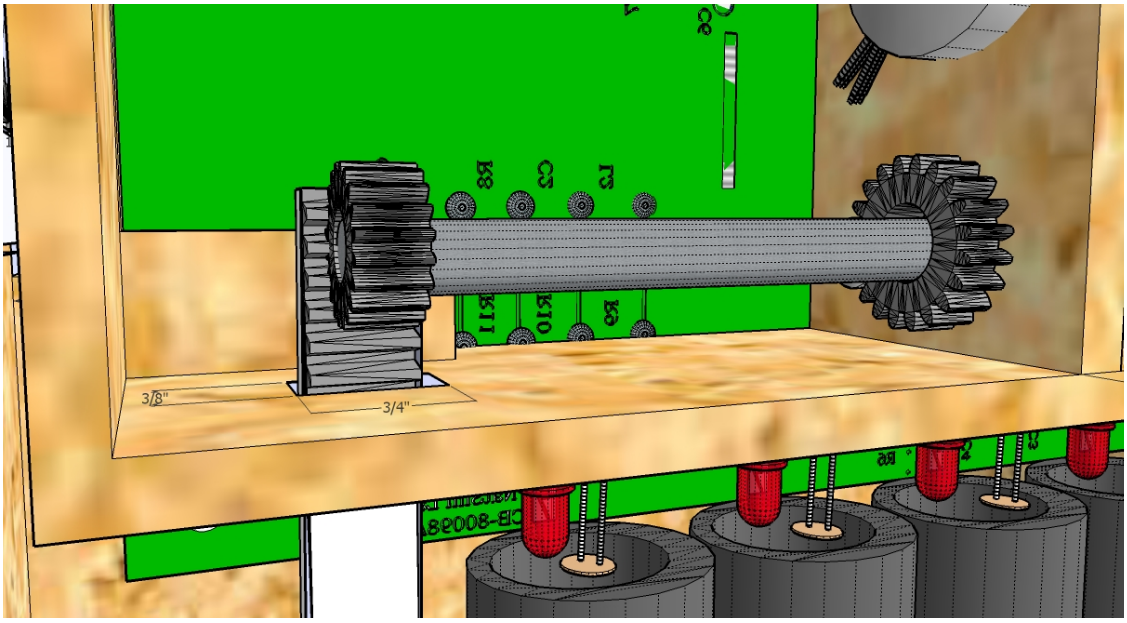
By integrating all three mechanical parts together, the robot will be able to first turn on the lights with the DC motor and then check one row of the egg tray and determine the position of the closet lights as well as if the lights are in good quality using the light sensors and LEDS. Afterwards, the robot will turn the lights off using the DC motor again with similar motion as turning the lights on. Next, the light sensors will be activated again to check if the lights are all turned off, in the process failing the lights that are still on. Finally the step motor will be activated so that the testing arm will continue to the next row. (7-5).



7-5 X-RAY VIEW OF TESTING ARM INTEGRATED IN THE MACHINE

7.3 SUGGESTIONS FOR IMPROVEMENT (ELECTROMECHANICAL PART)

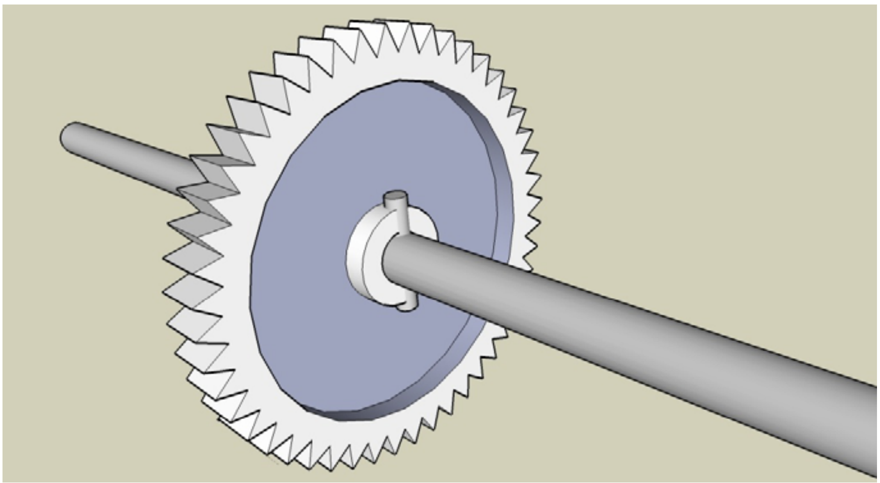
There are still many parts of the design that could be improved, however without properly testing these ideas, it would be difficult to conclude that these ideas would be compatible. First of all, a small full extension track (used for drawers) can be applied to the vertical motion[fig 1-7]. If the vertical motion rack is attached onto this track, there be little to no chance that the gear will still slide when the rack is powered to press the closet lights.



7-6 VERTICAL RACK AND PINION SYSTEM, RACK IS MOUNTED USING A SLIDE

7-7 FULL EXTENSION DRAWER SLIDE, USED TO RESIST FORCE IN ALL DIRECTIONS [13]

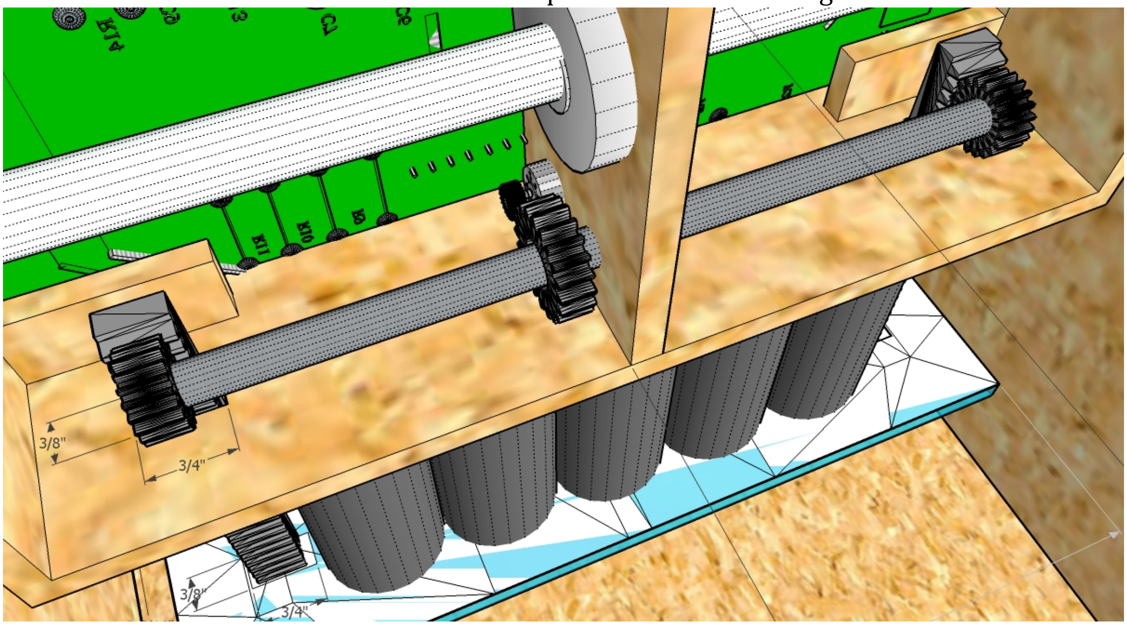
Also, the gears are mounted onto the shaft by punching a hole inside the shaft and placing a metal rod in there, then creating a slot and fixing the gear into place (7-8). This mechanism is unreliable since creating a slot on the original gear is very damaging to the gear. An alternative to this would be permanently attaching another smaller gear onto the original one and punching a hole in the smaller gear. In addition to this, we also place glue gun glue immediately beside the gear to help ensure that gear would not move out of place. However the main reason behind using glue gun is that it dries quickly and allows immediate testing. The glue gun was never intended to be a permanent solution to fixing the gear's position. A more reliable solution should be the Epoxy glue.



7-8 GEAR MOUNTED TO THE SHAFT, WILL BE FIXED USING GLUE GUN

Another weakness of our design is the idea that plastic gears are used instead of metal gears. In the end phase one of the key weakness that was observed was the gears breaking off the shaft. Due to the overwhelming amount of torque applied onto the gear during the repeated trials, the gears broke in half. One method to fix this problem was to use metal gears and rack system so that the gears won't break in long term use.

The next component that can require improvement would be the motors and gear ratios. Near the end of the integration process, we found out that the power output of the motor is high than predicted. When the power output of the DC motor is too high there is a good chance that the gears and rack will get severely damaged due to the high torque output. Since we did not have enough time to change the gear ratios of the DC motor, we decided to use a 100 spin/ min DC motor instead of the original 50 spin/min, also we were forced to add a 12 ohm resistor in front of the motor to further reduce the power. This is inefficient since a portion of the power is being wasted. An alternative solution to this problem would be reducing the gear ratio allowing the testing arm to move down faster. Doing this would allow a faster downward motion at same torque as our current design.



7-9 GEAR RATION OF THE VERTICAL MOTOR, DC TO SHAFT TO RACK

8 DATA CHARTS

Refer to appendix for information about the DC and stepper motor

9 SUPPORTING CALCULATIONS

Refer to appendix from power estimations of both stepper motor and DC motors. The calculations for torque to current ratio for both 100 rpm and 50 rpm DC motor will also be provided.

9.1 CURRENT TO TORQUE RATIOS OF DC MOTORS

By measurements, we determine Is, R and frequencies as follows for the 100rpm motor used.

$> I1 := 0.07;\ I2 := 0.30;\quad \omega1 := \frac{2\cdot\pi\cdot 100}{60};\ \omega2 := \frac{2\cdot\pi\cdot 25}{19};\ V := 11.25;\ R := 6;$

$$I1 := 0.07$$
$$I2 := 0.30$$
$$\omega 1 := \frac{10}{3}\pi$$
$$\omega 2 := \frac{50}{19}\pi$$
$$V := 11.25$$
$$R := 6 \qquad (1)$$

$> \text{Ratio1} := \frac{solve\left(I1\cdot V = I1^2\cdot R + \tau1\cdot\omega1,\ \tau1\right)}{I1};$

$$Ratio1 := 1.034188820 \qquad (2)$$

$> \text{Ratio2} := \frac{solve\left(I2\cdot V = I2^2\cdot R + \tau2\cdot\omega2,\ \tau2\right)}{I2};$

$$Ratio2 := 1.143050801 \qquad (3)$$

>

By measurements, we determine Is, R and frequencies as follows for the 50rpm motor used.

$> I1 := 0.0172;\ I2 := 0.22;\quad \omega1 := \frac{2\cdot\pi\cdot 50}{60};\ \omega2 := 2\cdot\pi\cdot 0.526;\ V := 11.25;\ R := 18;$

$$I1 := 0.0172$$
$$I2 := 0.22$$
$$\omega 1 := \frac{5}{3}\pi$$
$$\omega 2 := 1.052\,\pi$$
$$V := 11.25$$
$$R := 18 \qquad (4)$$

$> \text{Ratio1} := \frac{solve\left(I1\cdot V = I1^2\cdot R + \tau1\cdot\omega1,\ \right)}{I1};$

$$Ratio1 := 2.089462487 \qquad (5)$$

$> \text{Ratio2} := \frac{solve\left(I2\cdot V = I2^2\cdot R + \tau2\cdot\omega2,\ \tau2\right)}{I2};$

$$Ratio2 := 2.205778584 \qquad (6)$$

>

10 CIRCUIT SUBSYSTEM

10.1 ASSESSMENT OF THE PROBLEM

The circuit is the interface between the controller and the mechanical system. On one hand, it must convert signal given by the microcontroller to appropriate voltages that can drive the mechanical system and the sensors. On the other hand, it must collect sensory information from the sensors and feed it back to the PIC. In addition, the circuit system also includes the sensors themselves.

Therefore, there are really two main parts to the circuit system.

1) Interfacing between microcontroller and the rest of the machine
    a) Interface for stepper motor
    b) Interface for D.C. motor
    c) Interface for light position sensor
    d) Interface for light functionality sensor
    e) Interface for position feedback sensor

2) Sensor Design
    a) Reliable position sensor
    b) Reliable functionality sensor

Parts a) and b) of the first problem basically involves converting a 5V signal from the controller to a (possibly different) voltage that can supply a substantial amount of current (~0.1-1A). Parts c) d) and e) of problem 1 involves reliably feeding the signals to the microcontroller. Interference could be an issue here.

The second part involves reliable sensor designs.

Reliability is the key to both part 1 and part 2. The goal is to design and develop simple, reliable, flexible, and maintainable circuit system that get the job done. Modular design was in mind since the very beginning that every circuit component should be detachable from the machine without much effort so they can be repaired and tested separately. Every specialized circuit should be implemented on its own board and mutual dependencies should be minimized.

10.1.1 ASSESSMENT OF SENSOR DESIGN

The following decision tree nicely summarizes our decision (in green) vs. the alternatives (in white). The decision making process is discussed after the diagram.

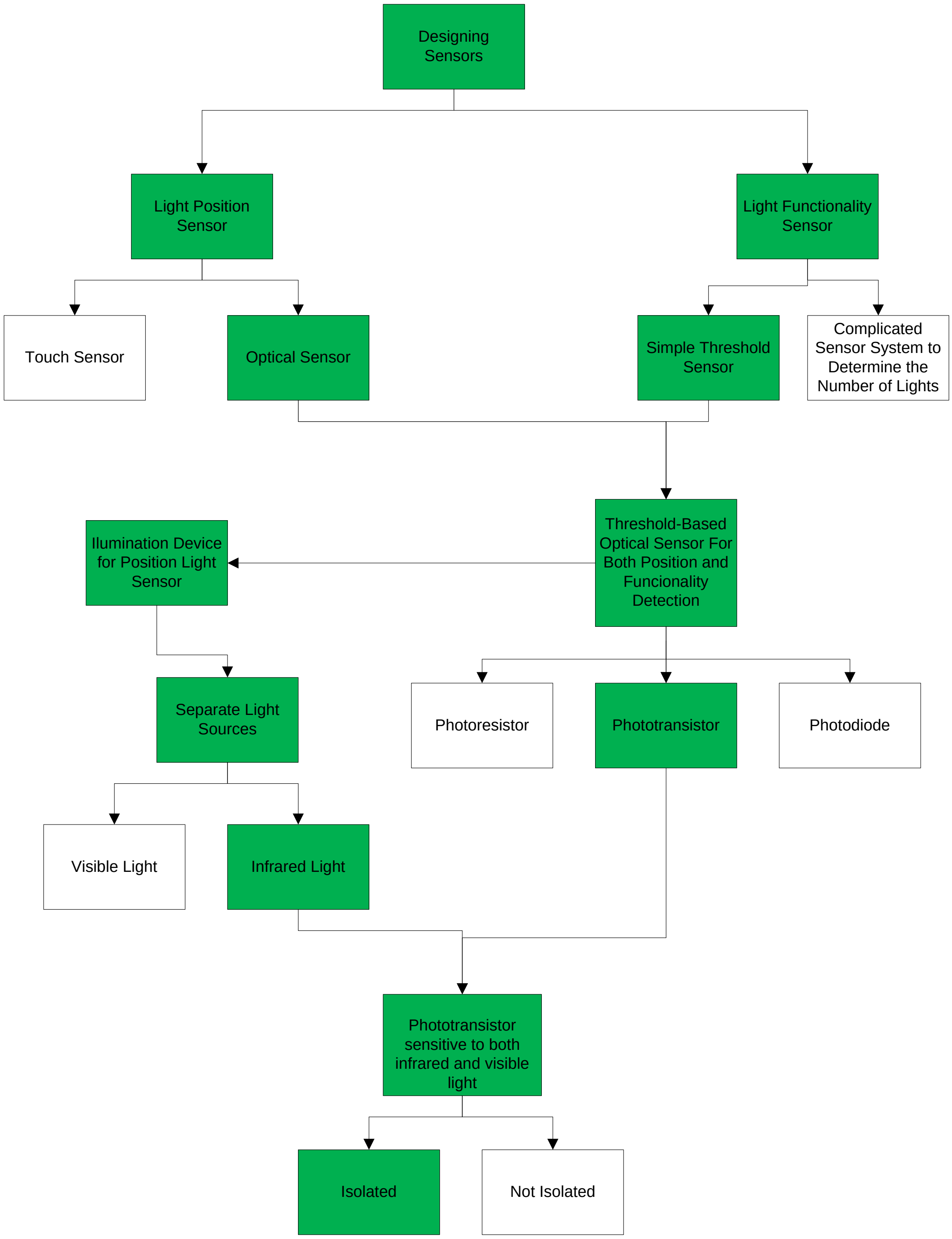


FIGURE 10-1, THE DECISION MAKING TREE FOR SENSORS

10.1.1.1 NUMBER OF DETECTORS

A) 1 Single detector
B) A row of 4 detectors

C) A row of 5 detectors
D) Two rows of 5 detectors
E) A grid of 20 detectors

More detectors imply a potentially shorter operating time but also more complicated circuits and high budget. Very serious consideration were given to the 20 detector arrangement but the budget would get above $150. It is decided that C (a single row of 5) works the best. In comparison to D and E, the circuits are much easier for C so that a pin-to-pin connection to the PIC is possible. In comparison to B, C involves one more sensor and improved operating time and operating time is fairly important according to major objectives.

10.1.1.2 DESIGN CONSIDERATIONS OF FUNCTIONALITY DETECTOR

A) Complicated system that can determine the number of lights that lit up
    a. Rotating sensor
    b. Sensor grid
B) An enhanced detector that more reliably determines the status of the closet light
    a. Lenses
    b. Fixing the relative position of the light and the sensors by a grip
C) A simple threshold-calibrated detector

Given that the product should cost below $150 complicated detection system is not feasible. A single threshold-calibrated detector will not work all the time because two bright LEDs are more intense than 3 dim LEDs (according to experiments). However, such situations should not occur often since the intensity between 2 and 3 LEDs are distinguishable most of the time and closets with exactly 2 LEDs on should only be a rare case. So in the spirit of keeping it simple and low cost, it is decided to use the simple threshold calibrated detector.

10.1.1.3 DESIGN CONSIDERATIONS OF POSITION DETECTORS

A) Position detection from the bottom of the egg-tray
B) Position detection by touch sensors
C) Position detection by light sensors

A relatively large force needs to be applied to turn the lights on. In addition, light needs to be transmitted from the closet light to the sensors above. So in order for choice B to work, the sensors should not obstruct any movements and it should not affect the transmission of light. A design satisfying those conditions may easily involve other tradeoffs.

Position detection by touch from the bottom would involve additional moving parts. Or at the very least, this method involves additional sensors.

Alternative C involves shining a light to the egg tray and measure the reflection (egg tray and the silvered closet light have very different reflections). It is the best alternative in our opinion. Light sensor is required to determine the functionality of closets and it can be used for position detection as well without major modification. According to experiments, this is quite reliable and only a few fairly cheap additional light sources (about $0.6 each) are needed. Moreover, when the machine inspects row-by-row, position detection by light opens the possibility of not pressing a row when that row is empty whereas each row must be pressed in the case of touch detector.

10.1.1.4 ILLUMINATION DEVICE FOR POSITION DETECTION

A) Single light source
B) LED grids
C) Infrared

This is a consideration unique to our position sensor since it is decided to use light sensors for position detection. There are several alternatives. A single light source for all position detectors is possibly cheaper but the relative position of each sensor with respect to the light would be different and therefore the same reading would mean different things at each sensor.

To keep things simple, a dedicated LED light source for each detector is better since there are only 5 detectors and LEDs are cheap.

The other consideration is the type of light to use. Visible light is easier to see and therefore easier to debug than infrared but infrared source has a larger signal-to-noise ratio. In addition, the acrylic glass that is used is very transparent to infrared. Experiments comparing two combinations: CdS photoresistor and white LED vs. phototransistor and infrared LED showed that the phototransistor and infrared LED combination works more reliably and consistently.

10.1.1.5 TYPE OF PHOTOSENSORS FOR POSITION AND FUNCTIONALITY DETECTORS

A) Photoresistor
B) Photodiode
C) Phototransistor

Phototransistor is most commonly used for photoelectric sensors while photoresistor is used when higher sensitivity for visible light is required. Photodiodes give faster response time and is quite linear over several orders of magnitudes [1]. Photoresistors are not sensitive to infrared light while phototransistors are sensitive over large wavelength ranges that include both visible and infrared light. Phototransistors often cost the same as photoresistors while photodiodes are much more expensive. According to experiments and as mentioned in the illumination device section there are several advantages for using infrared light for position sensors. In addition, because of a built-in focusing lens on the transistor, it is more directionally sensitive and thus less sensitive to ambient light. In consideration of all of the above, phototransistors was chosen.

10.1.1.6 SENSOR FIXATION

A) Design a mechanism to fix the relative position of a sensor with respect to the light in order to get a consistent reading.
B) Rely on the similarity of the egg tray (±1 cm)

Light intensity measurements are highly dependent on the relative position of the sensor with respect to the light. So this is an important consideration.

Alternative A involves gripping the closet light; this would likely involve additional moving parts. The positive positions of lights are fairly close to each other on the egg tray so that the gripping mechanism for one position would likely interfere with another light.

On the other hand, the egg trays can easily be stacked together so that they are fairly similar to each other and according to experiments a reliable and consistent readings still exists for the egg tray uncertainty of ±1 cm.

10.1.1.7 SENSOR ISOLATION

It is decided to isolate the sensors since it is very effective in improving the reliability of sensory readings at very little cost.

10.1.2 ASSESSMENT OF STEPPER MOTOR DRIVER

The stepper motor driver needs to convert 5V PIC signals to 12V signals that can deliver current of up to 12V/30Ohm = 0.4A. Within this framework there are several ways of doing it.

A) Use 2 PIC pins, so that one can indicate direction and the other one provides a clock.
B) Use 4 PIC pins, so that the signal directly corresponds to the motion of stepper motor.

Alternative A involves complicated circuit (a shift register) while saving pins is not an issue based on the sensor design. Alternative B gives PIC a lot of flexibility and takes advantage of its power and has the advantage of simple circuits.

In terms of implementation,

A) Build 4 switches using transistors
B) Use one of the stepper driver ICs

4 transistors (4x$1.6) that can handle current on the order of 0.5A typically cost more than the stepper driver ICs (~$4). The stepper driver ICs are also likely to be more reliable. Two types of ICs are available

A) TTL
B) CMOS

After some experimentation, the CMOS chip burns when a pin floats and the TTL chip is a lot more durable so a TTL chip L293 is chosen for this purpose.

10.1.3 ASSESSMENT OF DC MOTOR DRIVER

The goal here is still converting 5V signals to 12V that is capable of bi-directional drive with current of up to 1A. Because of this current requirement as well as a course requirement that forces the implementation, using transistors, of at least one actuator driver. The typical H bridge is chosen for this purpose.

10.2 SOLUTION

10.2.1 GENERAL

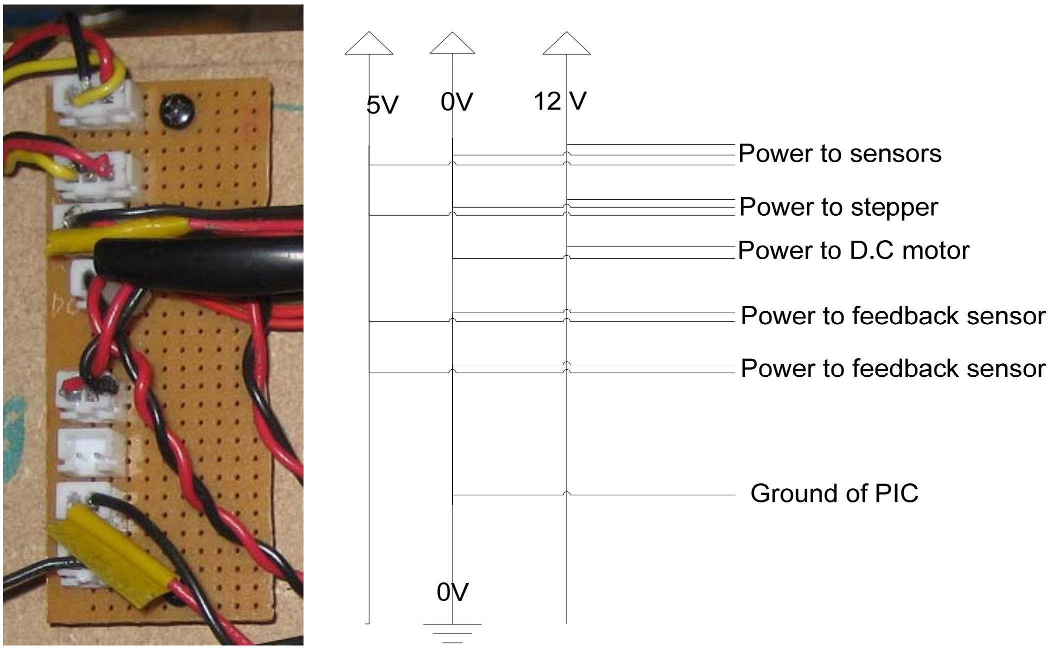


FIGURE 10-2, THE CENTRALIZED POWER HUB, ACTUAL PICTURE ON THE LEFT AND SCHEMATICS ON THE LEFT

Ample space is left on this power hub to make the system upgradable. The top plug connects to the power supply. The rest connects to different component as shown in the schematics.

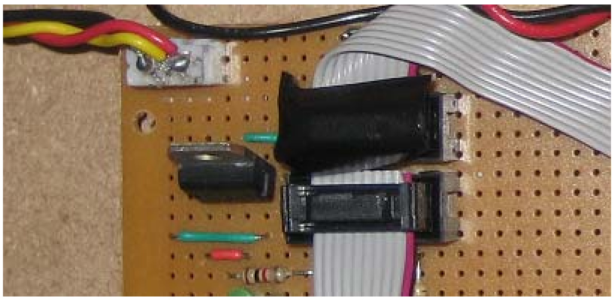


FIGURE 10-3, RIBBON CABLE AND CONNECTORS

It is the original intention to make the circuit as organized as possible. So ribbon cables and connectors are used extensively in the solution. They paid off in the debugging phase.

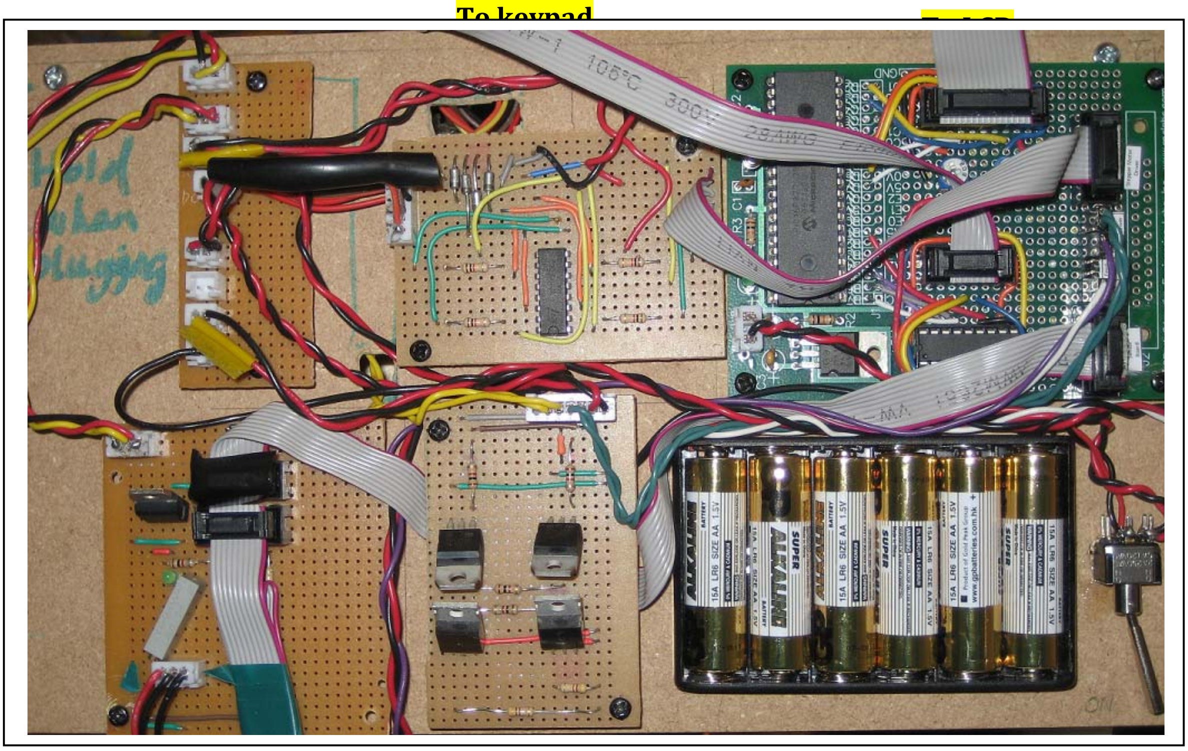


FIGURE 10-4, THE ENTIRE CIRCUIT AND CONTROL SYSTEM

This is one of the simplest and most maintainable circuits. Shown above are all the circuit boards! The actual sensors are not included in this picture but there is no complicated circuit element in the sensor board. The components are designed to minimize mutual dependence and allow for individual testing and maintenance.

Lines are colour coded, red represent high, mostly 12V, 5V for the PIC board, yellow represent 5V, and black represent 0V (or ground, this is unfortunately different from the computer power supply convention where 5V is the standard and is in red). Lines in other colours are reserved for signals. One possible exception is the D.C. motor driver where the yellow lines can be either 0 or 12V depending on the state of the D.C. motor.

10.2.2 SOLUTION FOR SENSOR

10.2.2.1 POSITION SENSOR

Infrared on

Closet light placed on egg-tray off

FIGURE 10-5, INFRARED LED AND TUBE FOR ISOLATION

The entire sensor system is placed inside a hollow cylinder. The angle of reflection will be much smaller than shown in the schematics (The infrared LED will be placed very close to the infrared sensitive phototransistor).

10.2.2.2 FUNCTIONALITY SENSOR

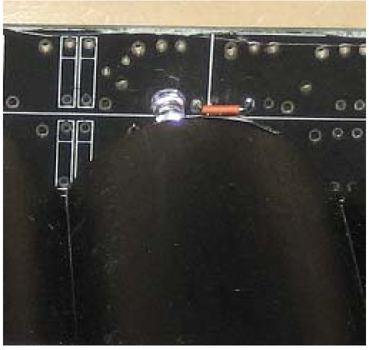


FIGURE 10-6, FUNCTIONALITY SENSOR

For the functionality sensor, a simple threshold would be set for the intensity of three lights so that the light passes its first test whenever 3 LEDs lights up. However, 2 LEDs occasionally can be as bright so 100% accuracy will not be guaranteed.

The design in the proposal were as follows,

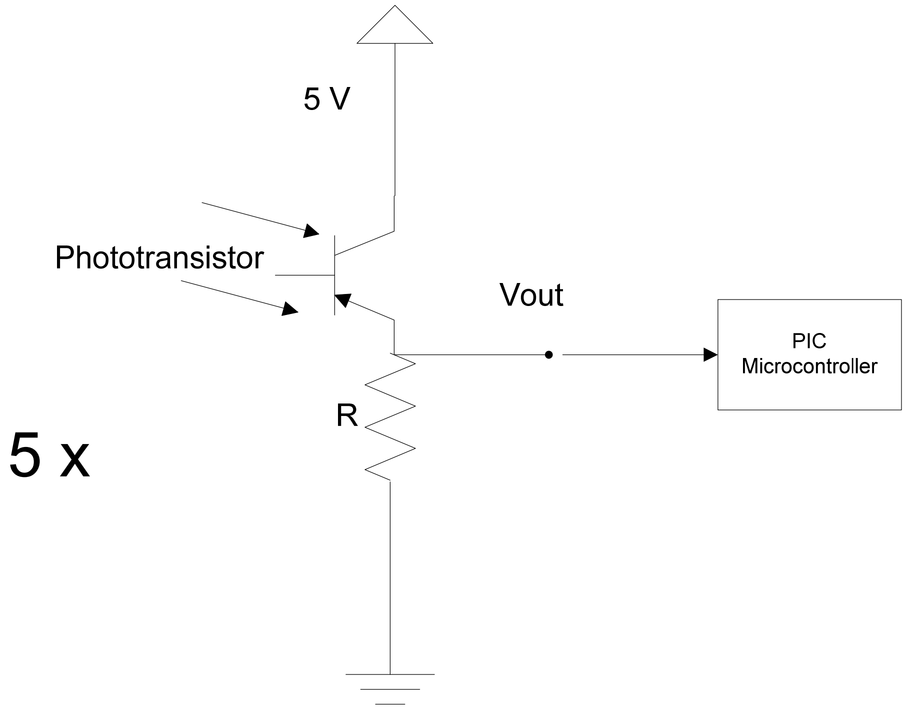


FIGURE 10-7, PROPOSED SENSOR DESIGN

Shown above is the circuit design for the functionality/position sensor LEDs. PT481 phototransistor was used. Its datasheet is attached in appendix C2. It was found that this setup with R=1 K ohm gives far superior sensitivity in the required range than any other configuration experimented with (in comparison to photoresistor and phototransistor with higher values). Key results were as follows:

With room light: Vout = **4.1V** ≈ 4.3V = Vhigh – Vsat = 5V – 0.7V = 4.3 V

With closet light shining directly on the transistor: Vout **= 0.7 to 1.3V**

With infrared LED reflecting off the egg tray: V out = **3.5 to 4 V**

With infrared LED reflecting off the closet light: Vout = **0.8 to 1.1 V**

The readings were highly stable and repeatable. Infrared reflection off egg tray and infrared reflection off closet light is very different and it thus enables reliable position detection. Also, the reading with closet light on is very different from the background light and this enables reliable functionality detection. This is reliable because the upper bound for positive results (1.3V) is much smaller than the lower bound of negative results (3.5V) so that setting the threshold in the middle has a Factor of Safety of more than 1.5.

However, it was later decided to use 0.5K Ohm instead of 1K Ohm for even more sensitivity in the given range. And the arrangement become slightly different

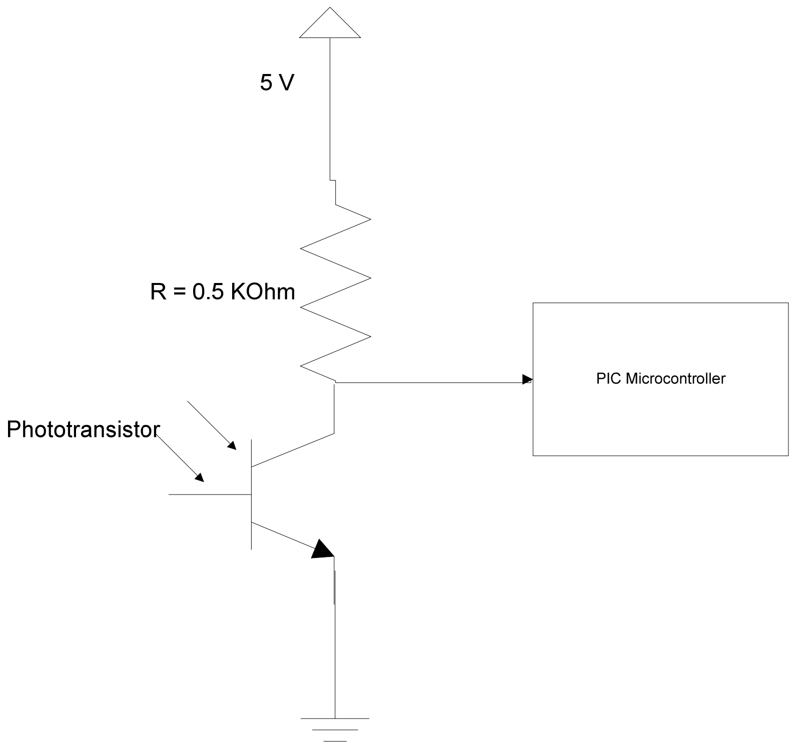


FIGURE 10-8, SENSOR DESIGN

The phototransistor is still the highly directionally sensitive PT481. The signal is feed directly into the PIC for position detection and for functionality detection of the closet lights.

The change from proposal is rather minor and is done just for convenience. Again, the readings are quite stable and repeatable.

For position sensing (thus with infrared on and light off), the results are as follows after feeding through the PIC's analog to digital.

	PIC High	PIC Low	Voltage Low	Voltage High
With Light	120	38	2.65	4.25
No Light	24	0	4.53	5.00

Here the PIC reading relates to (5V-voltage)/5x255. Therefore the PIC would display 0 if voltage is 5V and 255 if the voltage fed to it is 0V. The difference between 24 and 38 is not too significant, but due the high repeatability, position sensing was very reliable. In fact, after the orientations of the sensors were adjusted properly, position sensors never give any incorrect results provided that the threshold is set at 28 even after hundreds of readings were taken. The actual signal strength depends on the cleanness of the surface as well as the orientation of the light but the dependency is small enough so that there exist a clear threshold.

For functionality sensing, the results are as follows.

	PIC reading High	PIC reading Low	Voltage Low	Voltage High
3 Bright LEDs	204	85	1.00	3.00
2 Bright LED	204	70	1.00	3.22
1 Bright LED	170	45	1.67	4.12
No LED	0	0	5.00	5.00

Here, there is a clear overlap between 2 bright LEDs and 3 bright LEDs. Although in general, two bright LEDs still give a smaller reading than 3 LEDs, the reading actually depends more on the orientation of the light. Although a signal collection mechanism is designed for this purpose the lens on the phototransistor makes the sensor highly directional sensitive. As a result, there is no clear threshold. The threshold of 80 was chosen so that all light with 3 LEDs will pass but this threshold is highly flexible and can be adjusted just by changing one number in the code. This number is chosen since there should be very few light with 2 or 1 LEDs and the priority would be passing all lights with 3 LEDs. As a result, there is some chance to pass 2 or even 1 bright LEDs as well.

The reading is always 0 when there is no light. On the other hand, due to the intrinsic voltage drop across the transistor (0.6V), the reading is never 255. In fact, (255-204) x 5V / 255 = 1V is the reading at saturation.

10.2.2.3 INFRARED ILLUMINATION SYSTEM DESIGN

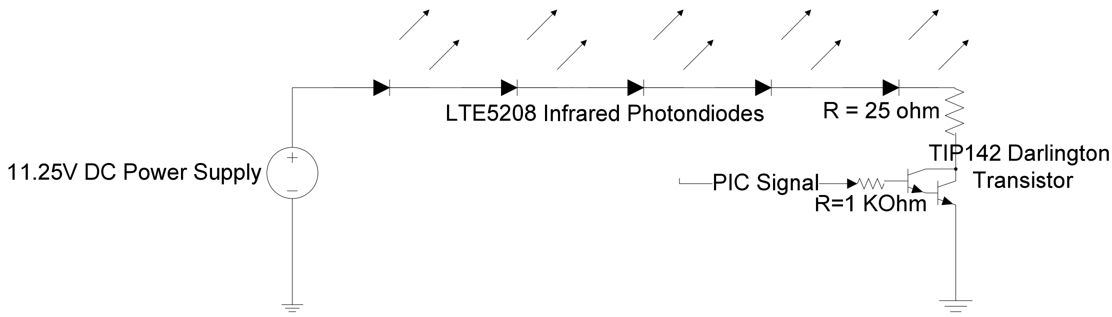


FIGURE 10-9, INFRARED ILLUMINATION SYSTEM

5 Infrared Light Emitting Diodes are connected in series. They are connected in series because the amount of light LEDs emit depends on the current going through them. LEDs connected in parallel have a huge inconsistence in intensity (~50%, Creatron Inc.).

The particular infrared LED is LTE5208. The device cost $0.65 and it emits very bright infrared light. The datasheet for this device is in the appendix C1. The value of R would be adjusted so the current going through the system is smaller than but about the same as that specified by the LED.

For example, the datasheet of LTE5208 states that I=100mA and V_{diode}=1.2V. Then if V=12V,

$$R = V/I = (12V-5x1.2V)/100mA=6V/0.1A=60\ Ohm$$

P = 12V x 100 mA = 1.2 Watt: this is feasible. Note that powering 1 LED costs just as much power as powering 6 in series with a 12V power supply.

The switch is controlled by the PIC and is implemented using a TIP142 Darlington transistor.

Using infrared LEDs could pose significant difficulties in aligning and debugging since they are invisible, the severity of this problem can be reduced by using near infrared viewers and infrared fluorescent cards. In reality though, sensors can be calibrated quite well just by twisting it while reading down the values. On the other hand, a cell phone camera conveniently picks up infrared due to its lack of infrared filter.

Each infrared LED is in reality isolated from each other and would feed into only one sensor. This is accomplished through plastic tubes installed for this purpose.

This system is exactly the same as proposed except a fuse was not installed since there was no risk for burning the LEDs (the current is actually 80mA and the LEDs would not burn even with 200mA going through it).

10.2.2.4 SENSOR DRIVER

The sensor driver is the interface between sensors and PICs. It is responsible for powering up infrared LEDs as well as the sensors. It is also responsible for sending the sensor signals back to the PIC.

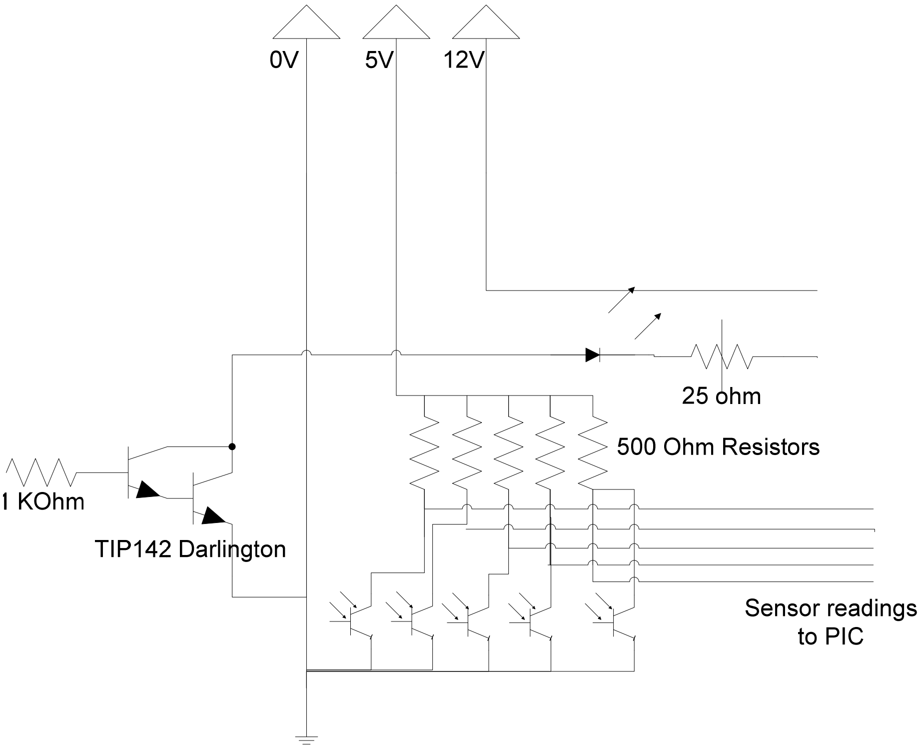


FIGURE 10-10, SENSOR DRIVER SCHEMATICS

FIGURE 10-11, THE IMPLEMENTATION OF THE SENSOR DRIVER

10.2.3 SOLUTION FOR STEPPER MOTOR DRIVER

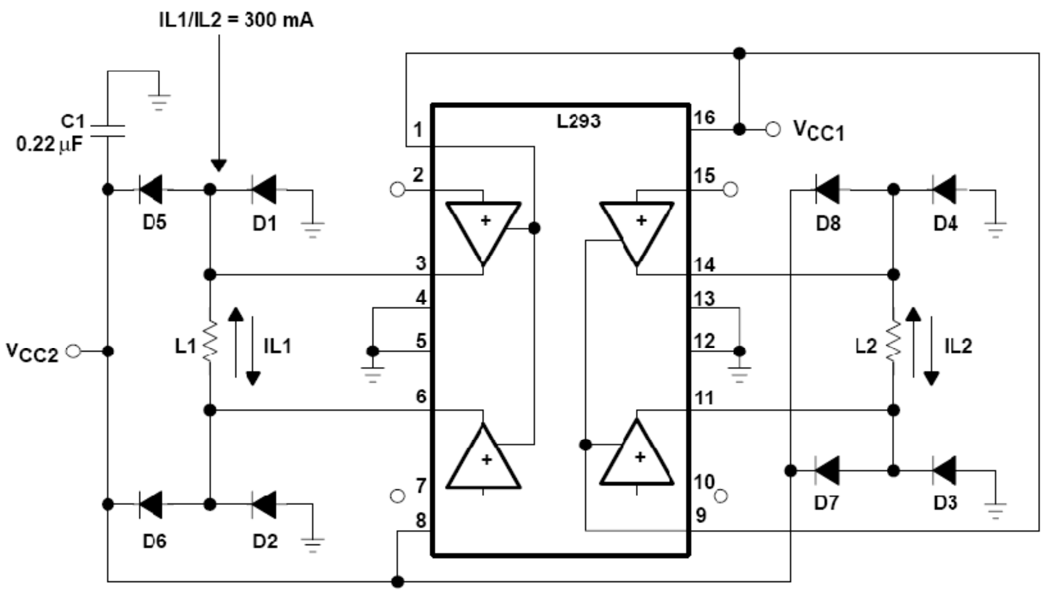


FIGURE 10-12, SCHEMATICS FOR MOTOR DRIVER (APPENDIX C3, L293D MOTOR DRIVER)

Pins 2, 15, 7 and 10 are the signal pins that are connected to the PIC with a 1K resistor in between. Two ends of L1 and L2 are the signal pins of the stepper motor.

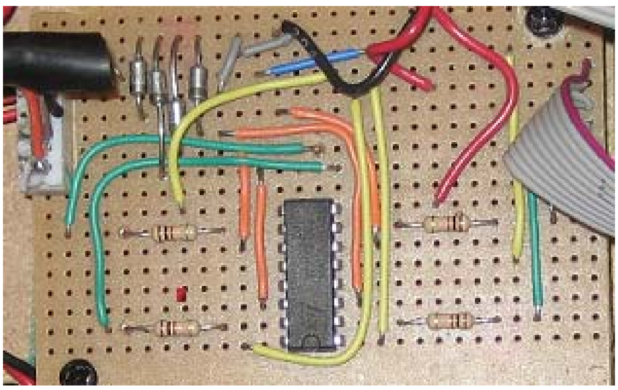
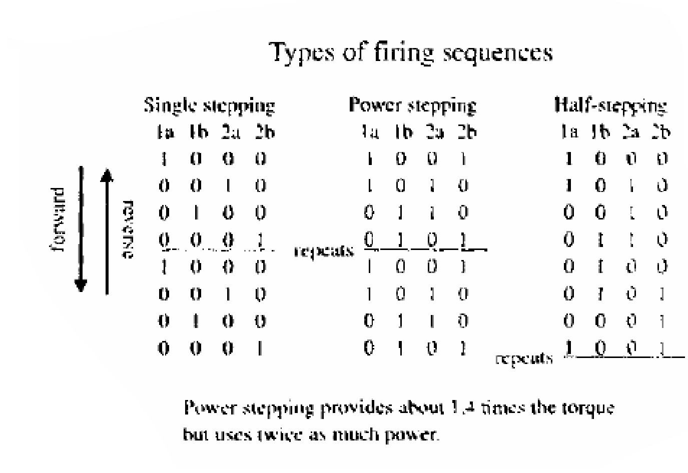


FIGURE 10-13, OUR IMPLEMENTATION

4 15V Zener diodes are used instead of 8 standard diodes for the same protection effect. The circuit is connected to the stepper motor on the left end through a 7-pins connector with 1 pin unused and is connected to the control board by a ribbon cable.



Types of firing sequences

Single stepping 1a	1b	2a	2b	Power stepping 1a	1b	2a	2b	Half-stepping 1a	1b	2a	2b
1	0	0	0	1	0	0	1	1	0	0	0
0	0	1	0	1	0	1	0	1	0	1	0
0	1	0	0	0	1	1	0	0	0	1	0
0	0	0	1	0	1	0	1	0	1	1	0
1	0	0	0	1	0	0	1	0	1	0	0
0	0	1	0	1	0	1	0	0	1	0	1
0	1	0	0	0	1	1	0	0	0	0	1
0	0	0	1	0	1	0	1	1	0	0	1

FIGURE 10-14, STEPPING SEQUENCES

Power stepping sequence was used in the final design.

10.2.4 SOLUTION FOR D.C. MOTOR DRIVER

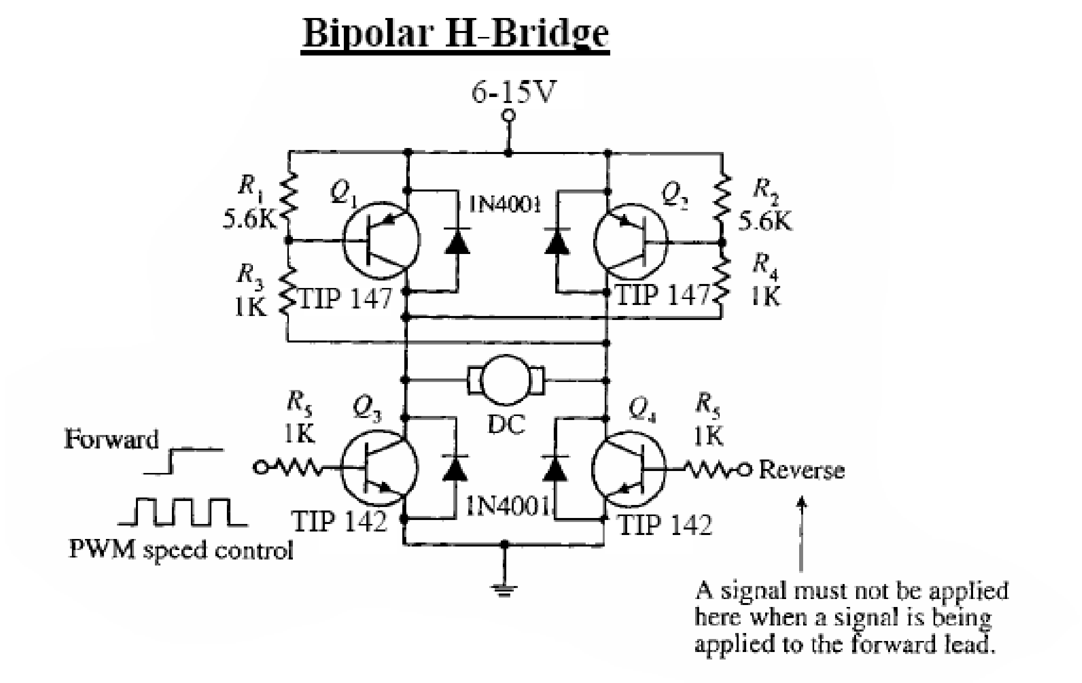


FIGURE 10-15 SCHEMATIC DC MOTOR DRIVER

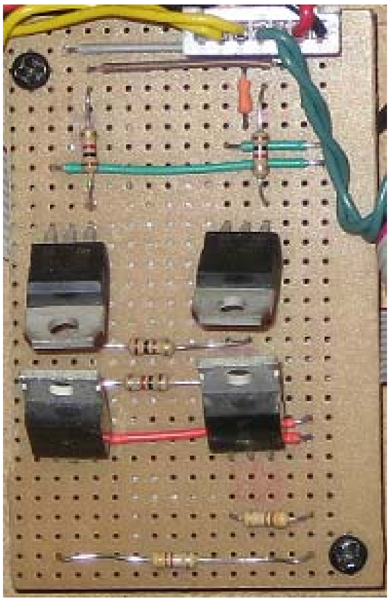


FIGURE 10-16, OUR SOLUTION TO DC MOTOR DRIVER

The top two Darlington transistors are TIP142 and the bottom ones are TIP147. This device should be able to handle current of up to 4A.

The motor turned out to be too powerful for the transmission so that a resistor of 12 Ohm was connected in series with the D.C. motor. See circuit appendix for sample measurements and calculations. Motor limited (as in the motor has just enough power to turn on the lights) is advantageous since it eliminates a feedback sensor and it protects gears. It also offers more flexibility to the machine in the case of a height change.

10.2.5 PIC PROTO64 BOARD SOLUTION

Refer to Appendix

10.2.6 FEEDBACK SENSORS

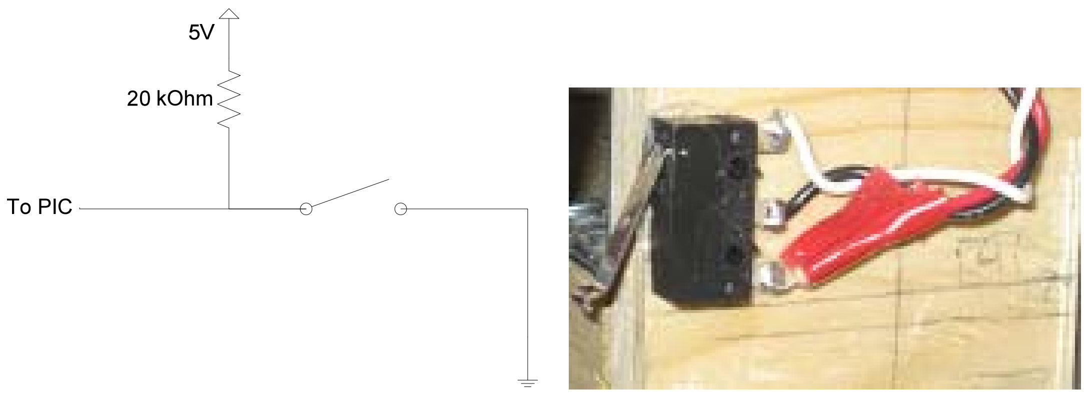


FIGURE 10-17, FEEDBACK SENSOR

10.3 SUGGESTIONS FOR IMPROVEMENT OF THE SUBSYSTEM

- Stranded wires: it is fine to use single-core wires on the boards but stranded wire should be used for inter-board connections. Single-core wires were initially used and they eventually broke apart after dozens of plugging/unplugging. By the time of the demo, almost all inter-board wires are replaced by stranded wires but the few that did not break are still single core wires. It would be better if stranded wires were used since the beginning.
- Instead of PT481, with its sensitivity dropping to 70% at a 10 degrees angle. A similar phototransistor with slightly less angle dependence would serve us well. Although the collection tube is fairly long it still cannot eliminate this angle dependence. We initially thought that this high angle dependence is to our advantage since it automatically filters out noise but it turned out to be more of a hindrance.

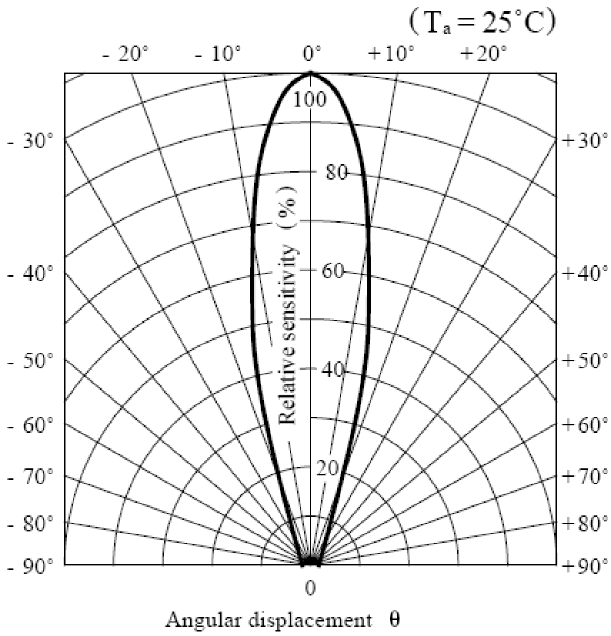


FIGURE 10-18, ANGULAR SENSITIVITY

- The L293d chip was soldered onto the board directly in the absence of a socket. Although it did not pose any problem, it should not be directly soldered to the board.
- More consistent colour coding. The colour coding in this design was generally good but it had some inconsistencies with the computer power supply convention. Some of the earlier boards were made based on availability of colours rather than a rigid convention. Wires of all colours should be obtained and a rigorous colour convention should be applied.
- It is better practice to add protection diodes on the D.C. motor driver as well. In this case, the maximum current is around 4A and only 0.5A is needed, so it was not necessary.

Overall, the circuit system is rather simple, reliable and maintainable. It did not pose any major problem ever since it was made. No circuit parts were replaced ever since it was installed to the backboard. As a result, not much need to be suggested for improvements of the circuit system.

11 MICROCONTROLLER SUBSYSTEM

11.1 OVERVIEW

Microcontroller subsystem was broken into X smaller divisions: machine interface, user interface, logic, timing, and log. Some general information such as stack usage, RAM usage… are also discussed in the following texts.

11.1.1 PROGRAM FLOW

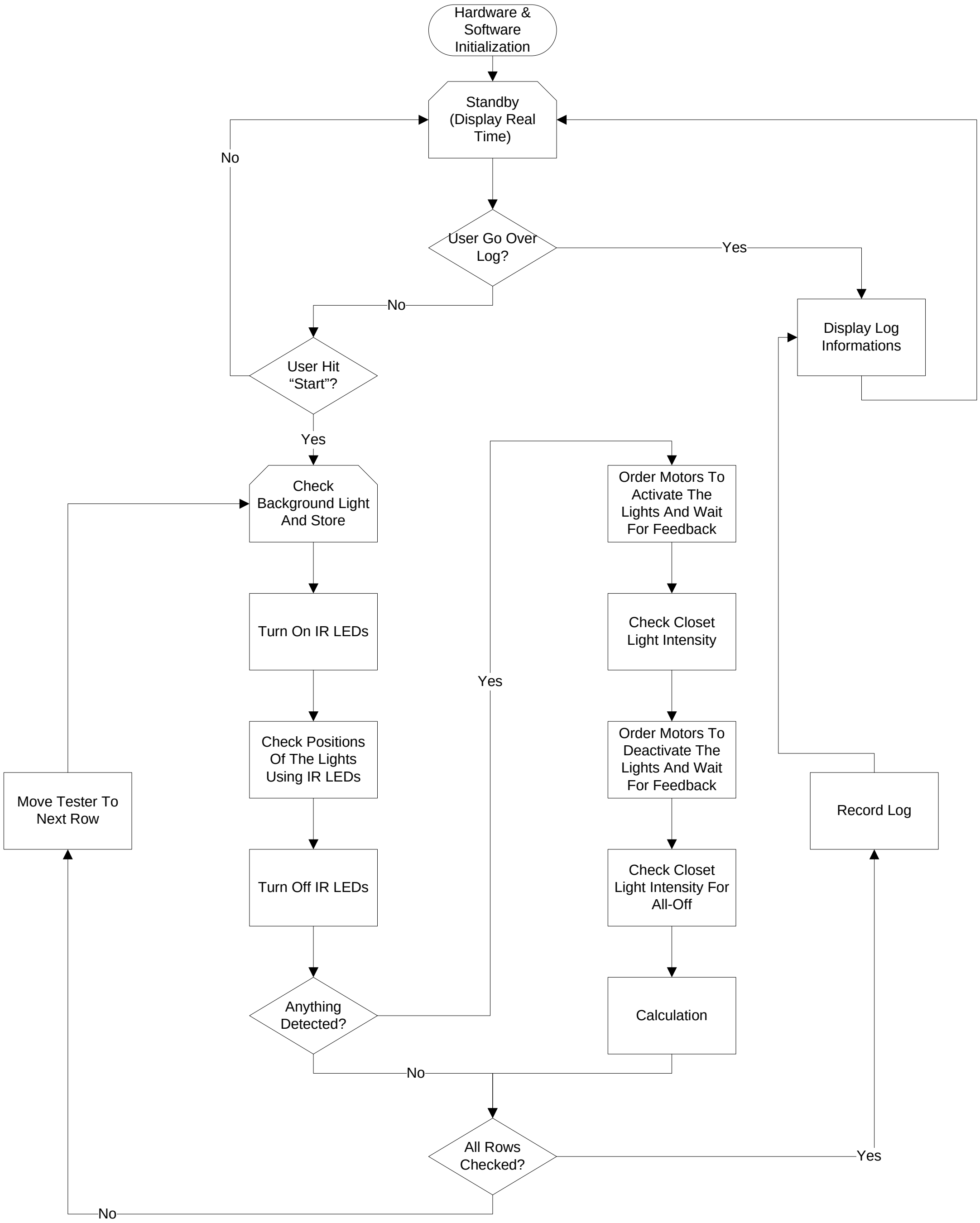


FIGURE 11-1 PIC PROGRAM FLOW CHART

The flow chart is almost same as that in the proposal, with the exception of "Sleep/Power Saving" feature, which is not implemented in the final design. Power saving feature of PIC program become less important because the main source of energy consumption of PIC board batteries is the LCD backlight, not the PIC.

The program has 7 phase of operation, defined in Table 11-1.

Label Name	Phase	Description
PHASE_HDINIT	0	Hardware Initialization/Reset
PHASE_RTCINIT	1	Real Time Clock Initialization/Reset
PHASE_REALTIME	2	Real Time Display
PHASE_NOREPORT	3	Report Review : No Report
PHASE_REPORT	4	Report Review
PHASE_RUN	5	Inspection Running
PHASE_FINISH	6	Inspection Finish

TABLE 11-1 PIC PHASE OF OPERATION

Some functions (Display, keypad Responds) behave differently in different phase. Refer to 11.3 User Interface for more detail.

11.2 MACHINE INTERFACE

11.2.1 PROBLEM DEFINITION

PIC program interact with the mechanical part of the machine by sending and receiving logic/analog signals from the controlling circuits. Following tables are the specification for microcontroller-circuit communication.

11.2.1.1 POSITION SENSOR

PIC Output/Machine Input	PIC Input/Machine Output	
LEDs	-	Machine Description
L		all infrared LEDs power OFF
H		all infrared LEDs power ON

TABLE 11-2 MACHINE INTERFACE: POSITION SENSOR

11.2.1.2 LIGHT SENSOR

PIC Output/ Machine Input	PIC Input/Machine Output					
-	AN0	AN1	AN2	AN3	AN4	Machine Description
	A[1]	A	A	A	A	analog signal should be sent to this 5 pins AT ALL TIME 5V: Lowest Intensity; 0V: Highest Intensity

TABLE 11-3 MACHINE INTERFACE: LIGHT SENSOR

[1] A=Analog

PIC will receive analog signal from the sensor. The program need to convert the signal to digital for logic part of the program.

11.2.1.3 VERTICAL DC MOTOR

PIC Output/Machine Input		PIC Input/Machine Output	
V.MOTOR C0	V.MOTOR C1	-	Machine Description
L	L		vertical DC motor power OFF
H	L		vertical DC motor power ON - moving UP
L	H		vertical DC motor power ON - moving DOWN
H	H		NEVER OCCUR (motor will burn)

TABLE 11-4 MACHINE INTERFACE: VERTICAL DC MOTOR

11.2.1.4 VERTICAL FEEDBACK

PIC Output/Machine Input	PIC Input/Machine Output		
-	TOP FB	BOTTOM FB	Machine Description
	L	L	both switches closed - IMPOSSIBLE, but PIC will turn OFF V.MOTOR
	H	L	TOP switch OPEN, BOTTOM switch CLOSED
	L	H	TOP switch CLOSED, BOTTOM switch OPEN
	H	H	BOTH switches OPEN

TABLE 11-5 MACHINE INTERFACE: VERTICAL FEEDBACK

11.2.1.5 STEPPER MOTOR

PIC Output/Machine Input					PIC Input/Machine Output	
EN	S0	S1	S2	S3	-	Machine Description
L	X	X	X	X		stepper motor power OFF
H	L	L	H	H		stepper motor power ON - moving FROM ROW 1 TO ROW 4
H	L	H	H	L		
H	H	H	L	L		
H	H	L	L	H		
H	H	H	L	L		stepper motor power ON - moving FROM ROW 4 TO ROW 1
H	L	H	H	L		
H	L	L	H	H		
H	H	L	L	H		

TABLE 11-6 MACHINE INTERFACE: STEPPER MOTOR

Stepper motor is controlled by signal pattern. The pattern repeats itself every 4 steps. The pattern listed here is the power stepping sequence. Refer to Figure 10-14 for reference.

11.2.1.6 HORIZONTAL FEEDBACK

PIC Output/Machine Input	PIC Input/Machine Output	Machine Description
-	ROW1 FB	
	L	switch CLOSED
	H	switch OPEN

TABLE 11-7 MACHINE INTERFACE: HORIZONTAL FEEDBACK

11.2.2 SOLUTIONS/ALGORITHMS

Generally (position sensor, DC motor, feedbacks), actuators are easy to communicate: just set/clear the corresponding pin when write, and test the pin when read. In the following sections, pin assignment is first introduced in general, and then stepper motor control is highlighted, initialization and analog to digital converting mechanism are introduced in the end.

11.2.2.1 PIC PIN ASSIGNMENT

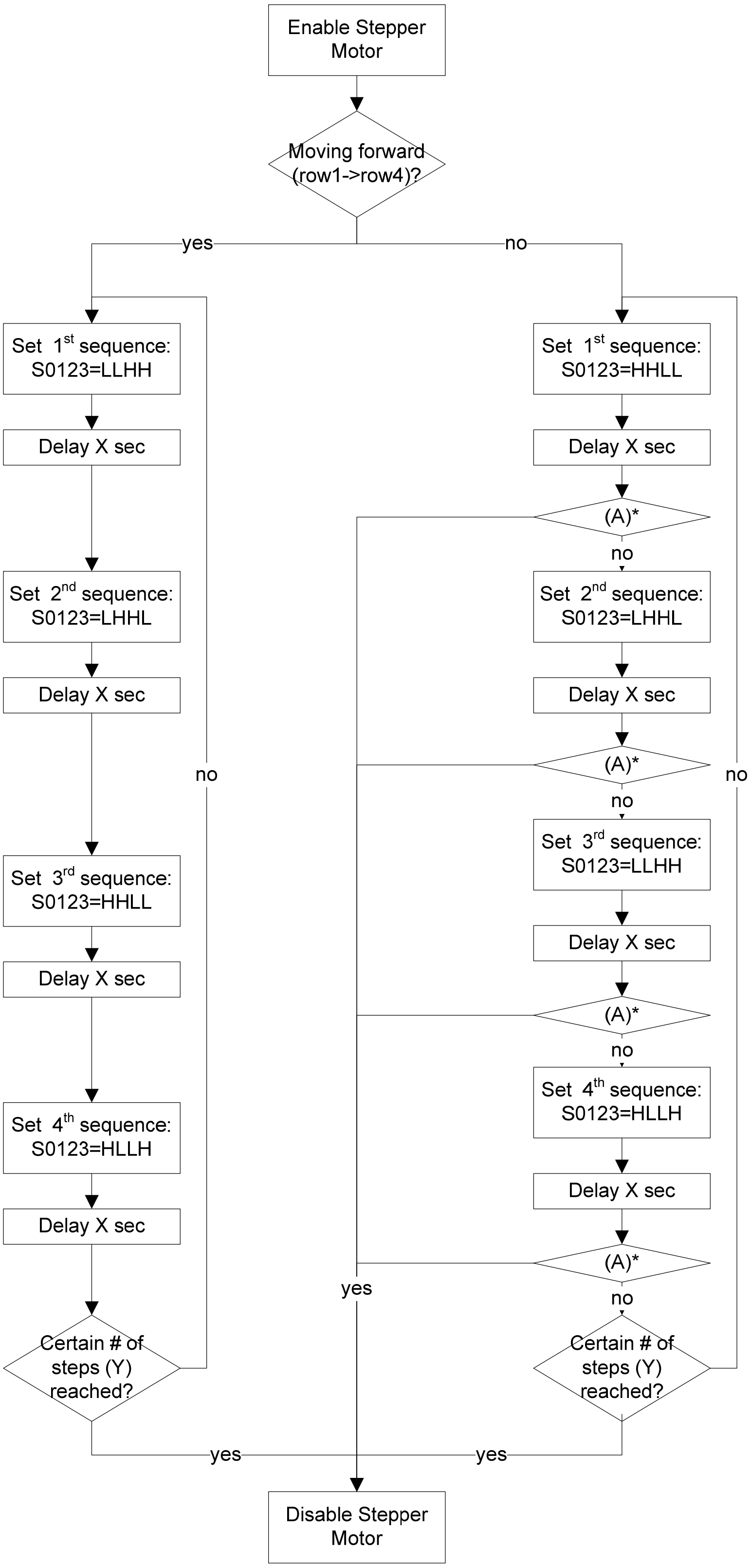
	RA		RB		RC		RD		RE	
0	AN0	I[2]	RV	O	V. MOTOR C0	O	S. MOTOR S0	O	S. MOTOR S2	O
1	AN1	I	KEYPAD DA[3]	I	V. MOTOR C1	O	S. MOTOR S1	O	S. MOTOR S3	O
2	AN2	I	RV	O	LEDs	O	LCD RS	O	S. MOTOR EN	O
3	AN3	I	RV	O	RV	O	LCD EN	O	-	-
4	RV	O	KEYPAD D0	I	RV	O	LCD DATA4	O	-	-
5	AN4	I	KEYPAD D1	I	TOP FB	I	LCD DATA5	O	-	-
6	-	-	KEYPAD D2	I	RV	O	LCD DATA6	O	-	-
7	-	-	KEYPAD D3	I	ROW1 FB	I	LCD DATA7	O	-	-

TABLE 11-8 PIC PIN ASSIGNMENT

PIC and the actuators achieved a one-to-one assignment without using multiplexer because this design only relies on a small number of actuators. For some actuator, like the stepper motor, a multiple-to-one pin assignment is used so that no additional interface circuit is needed to fully control the motor. After all, there are still 7 pins unused and can be used for future expansion in machine functions.

11.2.2.2 STEPPER MOTOR

[2] I = Input Pin, O = Output Pin, all relative to PIC.
[3] DA = Data Available



(A)* = "Horizontal Feedback Low(Pressed)?"

FIGURE 11-2 STEPPER MOTOR "ADVANCE ROW" ALGORITHMS

Figure 11-2 shows the basic algorithms of stepper motor control to advance the testing arm to next row. The program remembers which direction the testing arm should go: after reset the testing arm will be at row 1 so direction will be forward; after each inspection, the direction flips. As a result, the testing arm will move from row1 to row4 after reset, and then move from row4 back to row1 in the second run, and so on. This feature saves running time because the testing arm does not need to go back to row1 after every run.

Parameter Y decides the distance testing arm travels each time (one row), and X decides the speed the arm moves. From Appendix, stepper motor need to go 112 steps every time. So parameter SMOTOR_STEPS=Y $= \frac{112}{4} = 28$. Through repeated testing, X=30ms achieves a balance between speed and testing arm stability (too fast will cause the machine to shank).

11.2.3 INITIALIZATION

Initialization function consists of 4 parts (functions): PIC, hardware, software (variables), and real time clock.

PIC initialization function first disables the global interrupt, and then sets Timer0 Control bits, port states, and ADC Control one by one.

Hardware initialization reset the machine from whatever position it was left to the default position (acrylic plate at top position and testing arm at row1). The acrylic plate need to be lifted first so it does not hit the closet light while moving horizontally. Horizontal reset is done by call advance row function 4 times. The exact calling to move arm testing more than 3 rows to ensure it will travel back to initial position, where it will be stopped by the feedback.

Software initialization initialize important variables like report #, line #.

Real time clock initialization prompts the user to enter current date. The real time clock will start as soon as real time clock initialization function finish.

11.2.4 A/D CONVERTOR

PIC16F877 has an on-chip 10-bit Analog-to-Digital Convertor (ADC). It has up to 8 analog input pins and has resolution of 10 bits. According to the Mid-Range MCU Family Reference Manual, the absolute error is below 1 LSb (Least Significant bit) – only $\frac{1}{2^{10}} \times 5V \approx 0.005V$. In this design, the sensors are expected to distinguish value difference at 0.1V – far above error; therefore, PIC's built-in ADC is enough to serve the purpose of the project.

This design uses the sum of 4 consecutive A/D readings to reduce fluctuation in reading. Also for simplicity of one byte arithmetic, only the 6 most significant bit ($\frac{1}{2^6} \times 5V \approx 0.08V < 0.1V$) of the complement of digital result are used in calculation. The complement is used so that 0 corresponding to lowest intensity is more intuitive because the signal feed in has it reversed.

11.3 USER INTERFACE

11.3.1 PROBLEM DEFINITION

The PIC interacts with the user through LCD and keypad. LCD basically holds a multiline text that contains all information for a particular run. From keypad, the user can scroll up/down the multiline text to view the entire report. He/she can also go over logs for past runs and switch between real time and report.

11.3.2 SOLUTIONS/ALGORITHMS

11.3.2.1 LCD INTERFACE

Interfacing with LCD is a standard procedure, and there are numerous codes available online. This design used the sample code "Keypad_LCD", which came with the "PICusb" program loading application. The design modified the delay functions (now it send an instruction every 50μs instead of 200μs) in the sample code according to the responds time in TS1620-1 datasheet (Appendix) so it works faster.

11.3.2.2 KEYPAD

Keypad layout is defined in Table 11-9. From top left to bottom right, each key corresponds to a hexadecimal number. Pressing a particular key will pull up the Keypad_DA[4] pin and the corresponding key index will be send to Kyepad_D0~D3 pins.

0h 0000	Real Time Clock	1h 0001	Latest Report	2h 0010	(unused)	3h 0011	(unused)
4h 0100	(unused)	5h 0101	(unused)	6h 0110	(unused)	7h 0111	(unused)
8h 1000	Scroll Up	9h 1001	Last Log	Ah 1010	(unused)	Bh 1011	Stop
Ch 1100	Scroll Down	Dh 1101	Next Log	Eh 1110	(unused)	Fh 11101	Start

TABLE 11-9 KEYPAD LAYOUT, TEXT VERSION

[4] Data Available

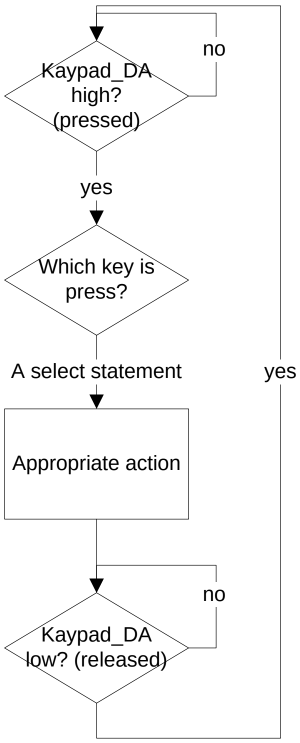


FIGURE 11-3 KEYPAD POLLING ALGORITHMS

Interfacing keypad used a polling algorithm shown in Figure 11-3. The program poll keypad_DA for low after action to prevent the action being executes multiple times if it runs too fast and the user did not release the key after it is done.

11.3.2.3 DISPLAY

Display function first makes up the content to be displayed in RAM, and then send it to LCD. The content to be displayed depends on the phase and line number and is shown in Figure 11-4.

Phase #	Phase	Line #	LCD Display															
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	Hardware Init	0	I	N	I	T	I	A	L	I	Z	A	T	I	O	N		
		1																
1	Real Time Clock Init	-																
2	Real Time Display	0	Y	Y	Y	Y	-	M	M	-	D	D						
		1	h	h	:	m	m	:	s	s								
3	Report Review : No Report	0	N	O		R	E	P	O	R	T							
		1																
4	Report Review	0	L	O	G		E	N	T	R	Y	:		8	8	/	8	8
		1	S	T	A	R	T	:			h	h	:	m	m	:	s	s
		2	F	I	N	I	S	H	:		h	h	:	m	m	:	s	s
		3	R	U	N	T	I	M	E	:					8	8	8	s
		4	T	O	T	A	L	:		8								
		5	P	A	S	S	:		8		F	A	I	L	:		8	
		6	L	A	Y	O	U	T	:									
		7										X	X	X	P	X		
		8										X	F	X	X	X		
		9										X	X	X	X	P		
		10										P	X	P	X	X		
5	Inspection Running	-																
6	Inspection Finish	0	I	N	S	P	E	C	T	I	O	N		F	I	N		
		1																

FIGURE 11-4 DISPLAY LAYOUT

Real time clock phase and inspection running phase do not use general display function since they have their own message to display.

11.4 LOGIC

11.4.1 PROBLEM DEFINITION

The machine must be able to decide which position has closet light and whether it is functional based on the sensor readings it collect.

11.4.2 SOLUTIONS/ALGORITHMS

Machine collect 4 groups of sensor readings for each row: background, reading after IR LED is on, reading after 1st press, reading after 2nd press. 3 meaningful tests can be done to the data:

Symbol	Expression	Comment

C0	Reading after IR LED is on >= THD_IRLED	True=there is closet light
C1	Reading after 1st press >= THD_3LED	True=the closet light has 3 LEDs
C2	Reading after 2nd press >= background +THD_BG	True=the closet light does not turn off after 2nd press

TABLE 11-10 LOGIC CONDITIONS OF SENSOR READINGS

Input			Output				
IRLED test (>=THD)	CL 3LED test (>=THD)	Off-BG test (>=THD)	Error 0=normal 1=error	Functionality 0=fail 1=pass	Existence 0=no 1=yes	Display	Note
C0	C1	C2	R2	R1	R0		
0	0	0	0	0	0	X	Nominal Response
0	0	1	1	0	1	F	Error (very unlikely: position sensing broken & first activation failed)
0	1	0	1	1	1	P	Error (position sensing system is likely broken)
0	1	1	1	0	1	F	Error (position sensing system is likely broken)
1	0	0	0	0	1	F	Nominal Response
1	0	1	1	1	1	P	Error (Mechanical Problem – First Activation Failed, Possible only two LEDs)
1	1	0	0	1	1	P	Nominal Response
1	1	1	0	0	1	F	Nominal Response (possible deactivation fail but light pass)

TABLE 11-11 POSITION/FUNCTIONALITY DETERMINATION

Table 11-11 listed all possible outcomes of logic test result and their corresponding displayed information. Decision making on error cases is based on the assumption that:

- There are far more functional closet light (3LEDs) than broken (2LEDs or less) ones
- Mechanical failure occur more often than a closet light cannot be turned off (because it then should be on at all time)

11.5 TIMING

11.5.1 PROBLEM DEFINITION

One requirement in the RFP was that the machine must be able to know its running time, to the precision that less than 5% error. In addition to that, because this design implemented a "Log System", it was required that the machine should keep a real time clock and write the real time to every entry in the log so the operator could know when did every inspection in the log happened. Many functions in the program also require delay sequence. Their specifications should follow:

Running time (Requirement)

- Less than 5% error

Real time clock (Bonus Points)

- Proper incrimination in minute, hour, day
- Can be calibrated
- Run without external power
- Less than 5% error

Being able to properly increment, run without external power, and be calibrated are the basics for any clock. The clock error should be less than 5% so running time calculation can use the clock.

11.5.2 SOLUTIONS/ALGORITHMS

11.5.2.1 RUNNING TIME

It is nature to use the real time clock to calculate running time. The program records the starting time when the user press "Start" and then records the end time when operation finished.

Minute difference = ((end minute – start minute) + 60) MOD 60

"+ 60) MOD 60" is to solve the hour-borrowing problem when end minute < start minute, so the minute difference will always between 0 and 59. This assumption is justified because RFP require the machine never run more than 3 minute.

Running time = (end second – start second) + minute difference * 60

Borrowing minute to calculate second is already included in minute difference.

11.5.2.2 REAL TIME CLOCK

11.5.2.2.1 POSSIBLE SOLUTIONS, ANALYSIS & DECISION MAKING

There are two possible solutions:

1. Use external real-time-clock chip DS1307
2. Use PIC internal clock and interrupt service to count second

A comparison was draw between these options:

Criteria	External Chip	Internal Clock
Proper incrimination in time	(+++) the chip take care of it	(+) doable, but need to write own functions
Can be calibrated	(+) easy to do (write some values to the chip)	(++) easy to do (change some values in RAM)
Run without external power	(+++) a external button battery will supply the power	(+) it can hold the value as long as PIC has power (powered by battery sets)
Less than 5% error	(+++) precision depends on the crystal, but generally very precise: frequency tolerance 20ppm [14]	(+) precision depends on the PIC crystal and interrupt frequency: 64ppm is the theoretical low limit for 10MHz, practically error is even bigger
Low cost	(---) additional $5 is required for the chip, more if a customize PIC board is made	(+++) no additional cost
Little material need to learn	(-) I^2C module	(-) interrupt service, Timer0 module

TABLE 11-12 COMPARISON BETWEEN EXTERNAL CHIP AND INTERNAL CLOCK FOR REAL TIME CLOCK

Both solution can achieve all criteria/requirement set in previous section, so the decision making dominated by the budget. Internal Clock solution is eventually chosen for its low (no) cost (thus more flexible for other part of this design).

11.5.2.2.2 SOLUTION: USE PIC INTERNAL CLOCK AND INTERRUPT SERVICE TO COUNT SECOND

Basic idea is shown in Figure 11-5.

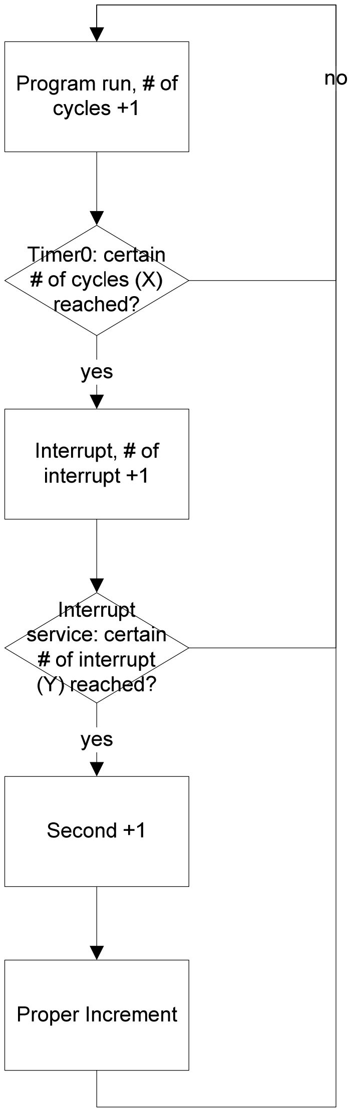


FIGURE 11-5 REAL TIME CLOCK ALGORITHMS

To decide X and Y, first need to determine number of cycles in one second:

$$1\text{s} = \frac{1{,}000{,}000\mu\text{s}}{0.4\mu\text{s/cycle}} = 2{,}500{,}000 \text{ cycles} = 26\ 25\ \text{A}0\ \text{h cycles}$$

PIC Timer0 Module interrupts every $X = 2^n \times 100h$ cycles by design, where "100h" is the max value TMR0 register can hold and "2^n" depends on the prescaler (a shift register) setting, n has value between 0 and 8.

To be most accurate (n=0):

$$Y = \frac{2625A0h}{100h} \approx 2626h = 100110\ 00100110\ b$$

To simplify the program, so that no 2-byte addition and conditional testing happens:

$$100110\ 00100110\ b \approx 10011000\ 100000\ b = 98h \times 40h$$

So 1:40h = 1:64 = 1:2^6 prescaler rate should be used (X=4000h) and Y=98h.

Accuracy is

$$98h \times 40h \times 100h \times 0.4\mu s = 260000h \times 0.4\mu s = 996{,}147.2\mu s \rightarrow -0.4\% \text{ error}$$
$$\rightarrow -1s/260s \rightarrow -13.8s/h$$

The clock will run 0.4% faster than real time, far below the 5% tolerance. Running fast is not bad because sometimes other interrupts may cause PIC to omit some Timer0 interrupts.

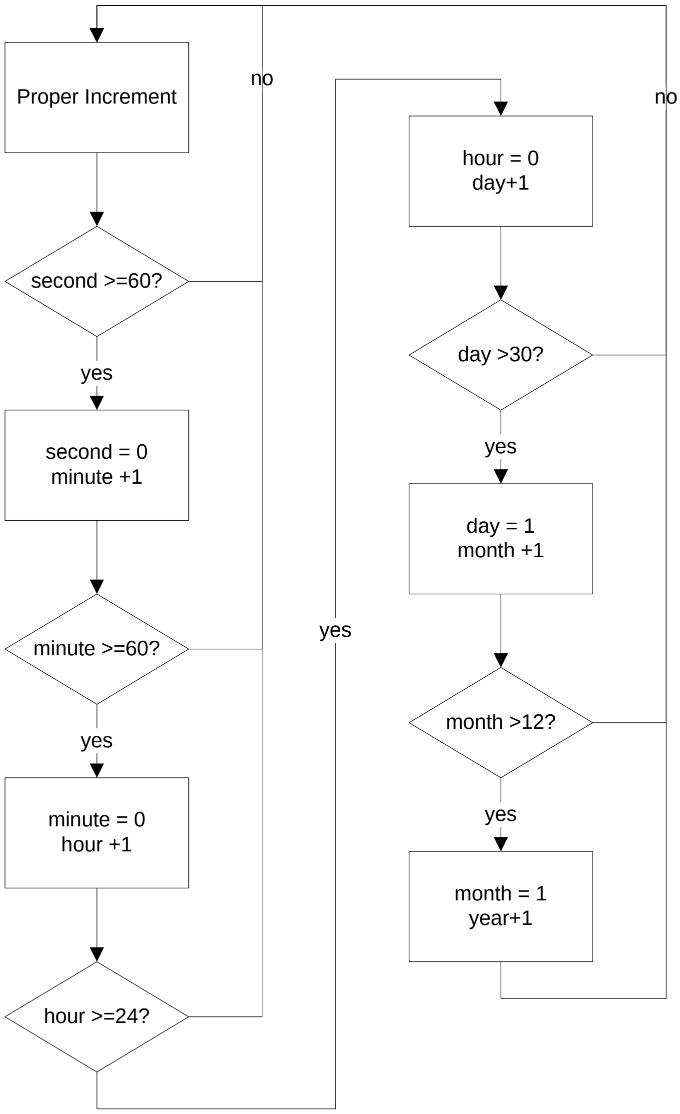


FIGURE 11-6 REAL TIME CLOCK INCREMENT ALGORITHM

The algorithm for proper increment is straight forward and shown in Figure 11-6. This program treats all month as 30 days for simplicity. This is justified because the batteries for the PIC cannot last for one month.

Calibration is simple: every time when PIC initialize, the program will prompt user to enter current date and time in "yymmdd" and "hhmmss" format, and store the number directly into memory. No further adjustment can be made other than restart the PIC for simplicity.

11.6 LOG

11.6.1 PROBLEM DEFINITION

The machine should hold as many entries as possible (Bonus Points).

Each entry must contain all information needed for a report.

11.6.2 SOLUTION/ALGORITHMS

A report contains following information:

- Log entry #
- Starting time
- End time
- Running time
- Total # of light
- # of light passed
- # of light failed
- Closet light layout

Among them, log entry # is related to the position in RAM of the log; end time can be calculated using start time + running time; Total # of light, # of light passed, and # of light failed can be calculated as long as closet light layout is known. As a result, only the following information needs to be stored:

- Starting time
- Running time
- Closet light layout

A log entry design is shown in Table 11-13.

Addr	+0	+1	+2	+3	+4	+5
bit	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
+000	runtime:8	year:4 | month:4	day:5	hour:5	min:6	sec:6
Addr	+6	+7	+8	+9	+A	+B
bit	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
+000	layout0 | layout1	layout2 | layout3	layout4 | layout5	layout6 | layout7	layout8 | layout9	layout10 | layout11
Addr	+C	+D	+E	+F		
bit	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0		
+000	layout12 | layout13	layout14 | layout15	layout16 | layout17	layout18 | layout19		

TABLE 11-13 LOG ENTRY DESIGN

The log entry is not compressed due to programming difficulty. The log can hold 255s for runtime (enough, since it never greater than 3min=180s), and up to year 2015, enough for the RFP.

Each entry occupies 16 Byte RAM. The log uses all bytes in Bank2 and 3, which gives $\frac{96\text{ Byte/bank} \times 2\text{ bank}}{16\text{ Byte/entry}} = 12\text{entries}$.

11.7 MISCELLANEOUS

11.7.1 PROGRAM ORIGINATIONS

This program code use multi-file program, comments, constants, and MACROs extensively to increase the readability of the code and thus reduce program error.

11.7.2 RAM LAYOUT

Refer to Appendix for a complete table. 300 Bytes out of 368 Bytes RAM are used in real time, corresponds to 82%.

11.7.3 TEST PROGRAM

A testing program is written for debugging the machine. It take sensor reading every 0.5s (faster than this will make the numbers changing too often to read) and display it onto the LCD. Keypad layout is shown in . Each button directly calls the corresponding machine interface functions.

Testing arm forward (Stepper)	Testing arm backward (Stepper)		
Testing arm down (DC)	Testing arm up (DC)		
Turn on IR LED	Turn off IR LED		

TABLE 11-14KEYPAD LAYOUT FOR TEST PROGRAM

11.8 OVERALL SUBSYSTEM IMPROVEMENT

11.8.1 SAVE CALIBRATION CONSTANTS IN EEPROM

One problem with the calibration of this design is that every time some calibration constants changes (i.e. sensor threshold, stepper speed), it is necessary to modify the code and reload the program. Since the PIC board circuit do not have a programmer module, this task is even more annoying.

To solve this program is to write all the constants in EEPROM, and every time the program initialize, it reads these constants from EEPROM. To expand this idea further, it is also make the calibration process convenient if a function that prompt user to enter new constant and write them into EEPROM exists.

It is also convenient to have a key designated to switch between the "real" program and the test program.

11.8.2 PROGRAM SHOULD BE WRITTEN IN C

Although writing the entire program in assembly language is a good learning experience for the programmer for better understanding of the microcontroller, there are limitations of using assembly:

- one has to keep track on every aspects of the PIC like stack and memory paging, which consumes a lot of time and easy to make mistakes
- one has to write his/her own helper functions as simple as one byte multiplication
- one has to learn all the instructions for the assembler

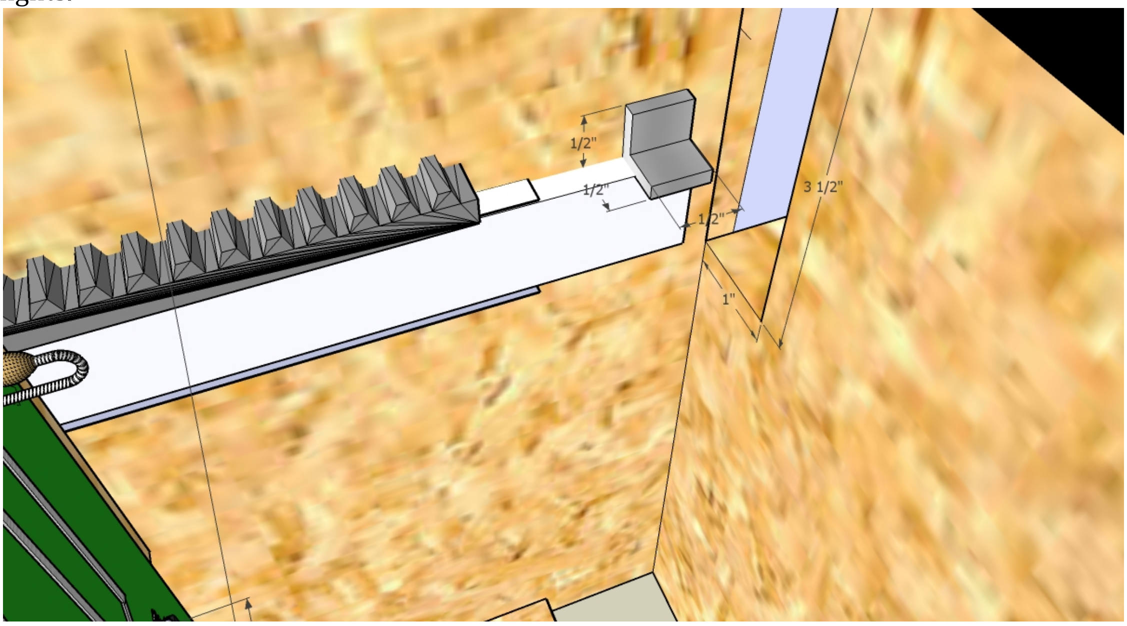
Using a C compiler can solve all the above problems. Although the compiled program written in C might be less efficient and larger than that in assembly, it is not so much a problem because this design uses only ¼ of the programming memory and efficiency is not a primary concern.

12 INTEGRATION

After all of the key components of the robots are complete, the integration process begins. For this particular project the integration process took approximately half of the time in our project timeline. In the integration process all the components of the robot is put together, and there will be no clear line drawn between the duties of the three members anymore. A lot of incompatibility was observed in this process. Thus most of the time was spent on debugging.

12.1 HORIZONTAL/ VERTICAL MOVING ARM

Since the horizontal and vertical arm is seen as the most complicated component of the robot, it was the first thing that was integrated onto the frame. The good thing is that by design the horizontal and vertical moving arm was built in one piece thus, there is no need to integrate these two components together any further. Half of the track was placed onto the frame while the other half was placed onto the moving arm. Fitting these two tracks together and the moving arm was successfully integrated onto the frame. Advantage to this integration process was that the moving arm can always be easily taken out of the frame for repairs. However, the track we used was not designed to withstand vertical force so a L was placed in the end of the track to keep the moving arm from lifting up when it's pressing the lights.



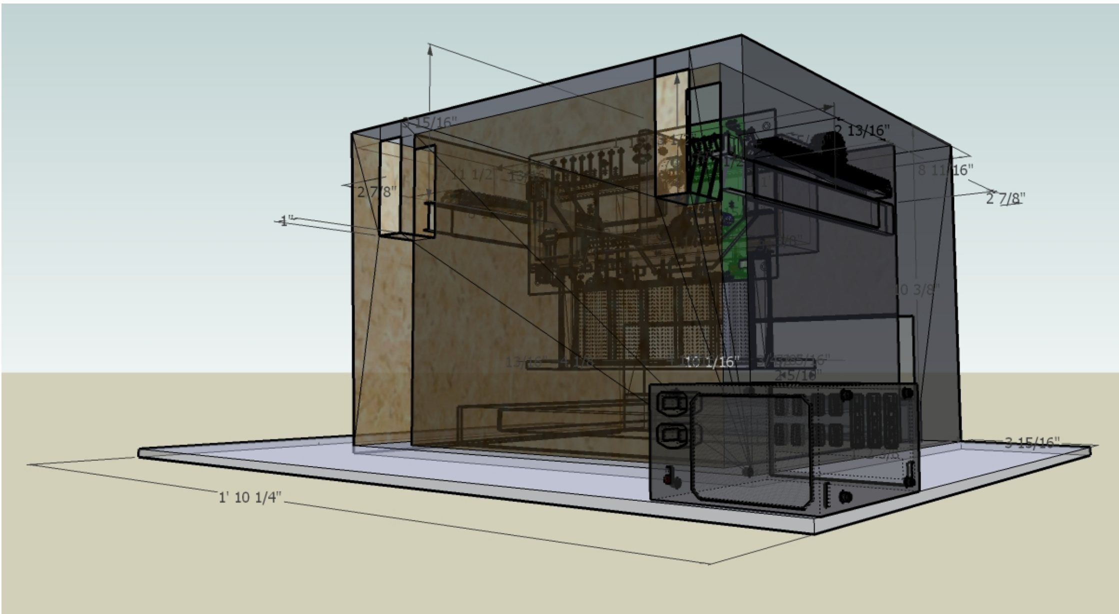
12-1 PIECE OF METAL WAS USED TO KEEP THE MACHINE FROM MOVING UP DURING THE LIGHT PRESSING PROCESS

12.2 PIC BOARD AND CIRCUIT BOARD

The PIC and circuit board was fitted perfectly onto the back of the moving arm, we did this cause the majority of the wires that need to be connected are from the circuit board to the motors and light sensors. Thus by doing so, the amount of wires floating around inside the machine is greatly reduced. However, this also made the moving arm much more complicated. When the gears or the motors require fixing, it is often necessary to take the circuit board off the moving arm. This caused a lot of trouble during the debugging process.

12.3 POWER SUPPLY

Unfortunately there was insufficient space inside the machine for another power supply, as a result we decided to place the power supply outside the robot which extended the size of the robot by quite a bit. Wires are connected from the power supply to the circuit board through a hole in the frame, and some portion of the wires was left floating.

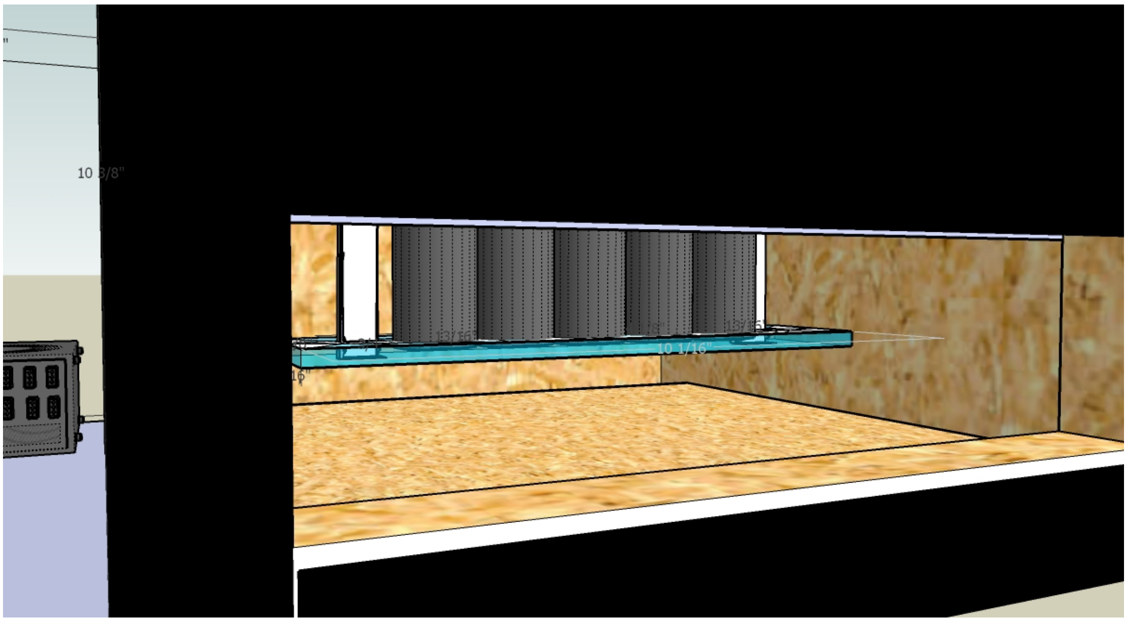


12-2 POWER SUPPLY LOCATED OUTSIDE THE ROBOT

(Refer to Appendix for larger version)

12.4 DRAWER

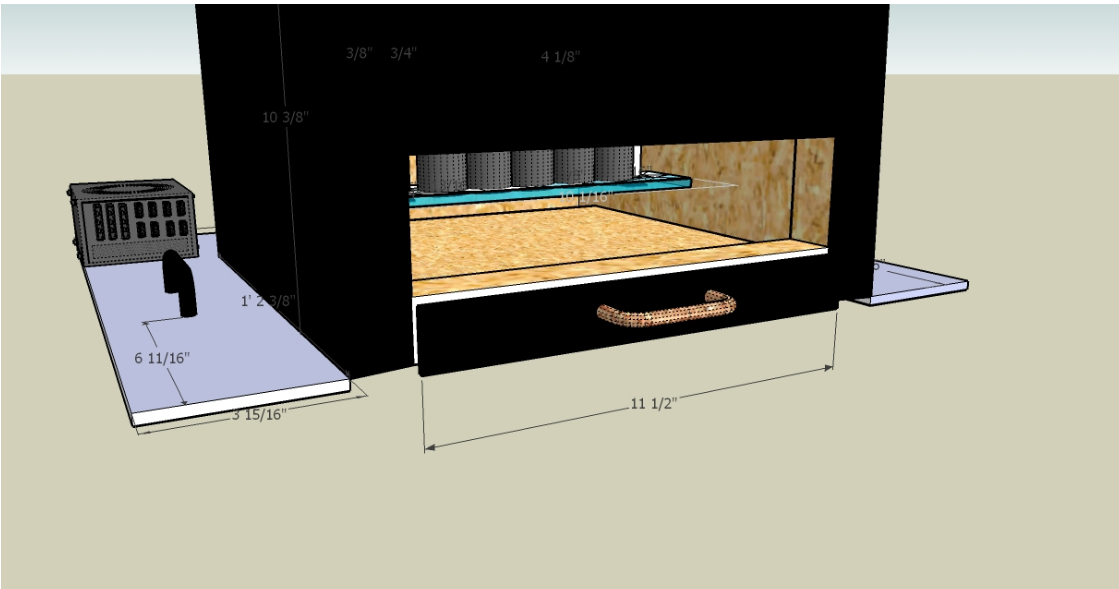
The drawer was initially just simply placed into the frame without any locking/ sliding mechanism. However due to the fact that the drawer was wobbling a lot which caused the position of the lights to vary, a decision was made to use a slide for the drawer as well. When the slide was placed onto the drawer, it has caused the drawer to become a lot tighter, and seemed to work quite well for locking the drawer in place.



12-3 DRAWER PUSHED INTO THE TESTING MACHINE

12.5 ADDITIONAL ITEMS FOR AESTHETIC PURPOSES

This is obviously the last thing that was added onto the machine. First, in order to keep the machine looking professional, we decided to cover the outside of the machine with a black cardboard. The top of the frame was locked into place with two hinges so that it still can be opened for debugging purposes. Then handles were added onto the drawer and the base of the frame, so that it is user friendly and easy to carry around.



12-4 HANDLES MOUNTED ONTO THE MACHINE

13 ACCOMPLISHED SCHEDULE

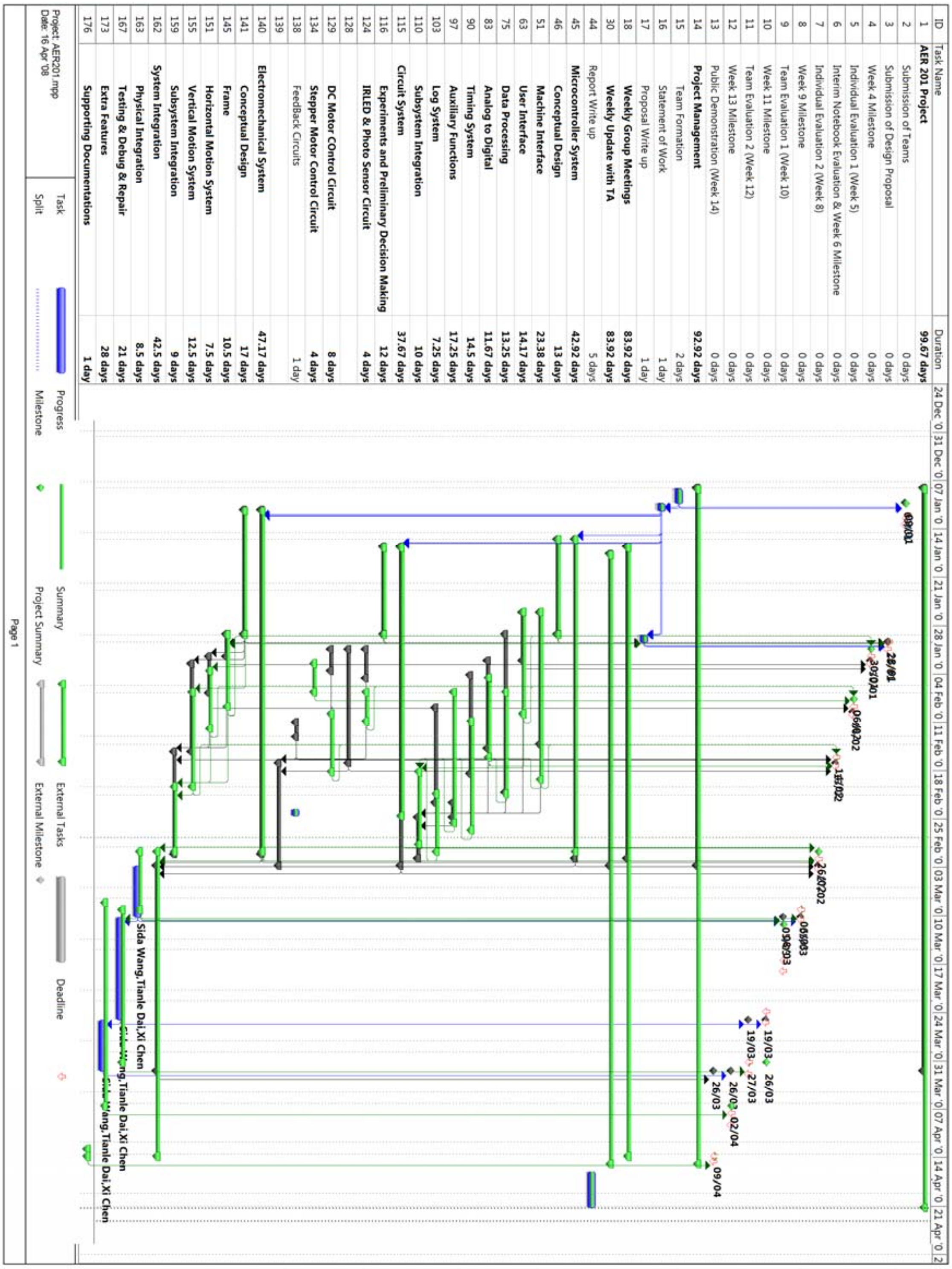


FIGURE 13-1 ACCOMPLISHED SCHEDULE

In order to finish the project (deliver the prototype) on time, this design used advanced project management software – Microsoft Project 2007. A Gantt chart generated by the software is shown in Figure 13-1.The green bar is the actual progress while the black is what is planned. The subsystem is generally consisting with the plan, while integration part varies a lot due to the inaccurateness in the planning. Some entry initially planned are discard later and new tasks are added in.

We used "divide and conquer" method to consecutively divide the project to smaller pieces until many simplest indivisible tasks. The project is first divided into five main categories (level 2. Level 1 is the project itself):

Category	Definition
Project Management	Including all administrative tasks, such as team formation, meetings and reviews.
Microcontroller System	First half of the course. Each member is designated to his corresponding subsystem.
Circuit System	
Electromechanical System	
System Integration	Second half of the course. After all subsystems completed, the integrated machine will be

TABLE 13-2 SCHEDULE OVERVIEW

13.1 DIVISION OF WORK:

According to the RFP, the Project implementation can be broken into 3 subsystems: microcontroller system, circuit system, and electromechanical system, each accomplishment by one member of the team. This proposal honours the RFP subsystem division but unite all three members in system integration. The work of each member is defined as following:

13.1.1 MICROCONTROLLER MEMBER (XI CHEN)

Microcontroller member shall finish microcontroller programming on his own in the first half of the course. (The course can be divided into 2 parts: first half Week 4 – 8, second half Week 9 – 13) He is also responsible for providing Analog to Digital Converter (ADC) subroutine to circuit member for test. (As described in previous section, this design uses on-PIC ADC for light sensors. After microcontroller member finished the ADC subroutine and circuit member finish circuit prototype, they will join together to do test and debug.) In second half of the course, he is responsible for building power supply for PIC Board, amending and calibrating the Light Sensors, per required by the RFP to help circuit member.

13.1.2 CIRCUIT MEMBER (SIDA WANG)

Circuit member is responsible for construct all sensors and actuators and their corresponding machine interface. Motor drivers and feedback circuits are expected to be finish before reading

TABLE 13-1 GANTT CHART SCHEDULE, OUTLINE LEVEL 3 week for electromechanical member testing his motors. In the second half of the course, microcontroller member will take over the work to calibrate and repair PIC Board Power and Light Sensors, the rest is still the responsibility of circuit member.

13.1.3 ELECTROMECHANICAL MEMBER (TIANLE DAI)

Electromechanical member is responsible for constructing the structure and assigning locations for sensors and circuits, as well as building driving and transmission system, in the first half of the course. In the second half, he is responsible for assembling sensors to the frame and repair/rebuild any structural parts damaged in the integration stage.

13.2 REVIEW MECHANISM

Module	Definition
Team Formation	Finding members and form the team. Already done.
Statement of Work	Each member chooses subsystem. Already done.
Proposal Write up	The time designated to finish the writing part of this proposal.
Weekly Group Meetings	A weekly group meeting on every Tuesday Afternoon. The team suppose to update the progress, prepare for the coming lab and review the task assignment for following two weeks.
Weekly Update with TA	During the lab session on every Wednesday, the team will meet with TA, update progress, ask questions encountered and discuss any changes in plan. It also includes a milestone to confirm the schedule for the second half of the course.

TABLE 13-3 SCHEDULE REVIEW

We are aware of the precision limitation (further the time away from now, less likely we are going to plan the tasks precisely) of our project plan, due to less experience and conflicts with other subjects; therefore, the schedule must be constantly under review. According to [2], we decide our planning limitation would be around 2 weeks. As a result, every week we can going to have a group meeting (Tuesday) first and then meet TA (Wednesday) to update progress and review the plans for the following two weeks. For the same reason, predicting the specific tasks (level 4) of system integration stage would be unrealistic. A milestone for detailing out the specific tasks is set 2 weeks before system integration starts.

13.3 TASK ASSIGNMENT

13.3.1 MICROCONTROLLER SYSTEM CATEGORY:

Module	Definition
Conceptual Design	Learning the language, Flowchart, Module Division, and General Pseudo Code.
Machine Interface	The communication between PIC and circuit, such as sensor

	readings, control signals and feedbacks.
User Interface	The communication between PIC and operator, through Keypad and LCD.
Data Processing	The subroutines for storing sensor readings and use them to determine closet light functionalities. RAM structure is also part of this module.
Analog to Digital	The subroutines to convert analog sensor input to digital values.
Timing System	Real time clock and timing mechanism.
Auxiliary Functions	Any helper functions required for major tasks, such as multiplication and division functions.
Log System	The subroutines for storing and retrieving data, including data compression formatting.
Subsystem Integration	Writing initialization and "main" program, then putting all subroutines together and debug.

TABLE 13-4 SCHEDULE PIC

For every module (Level 3), such as Machine Interface, we follow the Waterfall Design Model (Very commonly used in software engineering): Definition – Design – Implementation – Test. The following diagram is an example:

ID	ℹ	Task Name	Duration
40		**Machine Interface**	**18 days**
41	✓	Pin Assignment	0.5 days
42	✓	Device Addressing	0.5 days
43		Pseudo Code	1 day
44		**Implementation**	**3.25 days**
51		Debug	1.5 days

TABLE 13-5PIC SCHEDULE CYCLE

Similar development ideas also apply to circuit system and electromechanical system.

13.3.2 CIRCUIT SYSTEM CATEGORY:

Module	Definition
Experiments and Preliminary Decision Making	Sensitivity and reliability test for sensors, tradeoffs analysis, choosing sensors, and conceptual design for circuits.
LED Illumination Circuit	The circuit (on prototype board) that translate digital control to the on/off of the Infrared LED.
Photo Sensor Circuit	The circuit that regulate light sensor output voltage (which feed into PIC).
Motor Control Circuit	The circuit that control stepped motor, as well as DC Vertical Motor and its feedback.
Other Foreseeable Tasks	Other circuit that does not fall into above modules, such as power regulation and protection mechanism.

Subsystem Integration	Production of above circuits on circuit board with soldering.

TABLE 13-6 SCHEDULE CIRCUIT

13.3.3 ELECTROMECHANICALLY SYSTEM CATEGORY:

Module	Definition
Conceptual Design	Equipment/Requirement Learning, machine Shop Tutorial, choosing motors and pressing mechanism, preliminary drawings.
Frame	Machine Structure, including drawers.
Horizontal Motion System	The driving and transmission system that moves sensors row by row.
Vertical Motion System	The driving and transmission system that activates/deactivates closet light by moving up and down.
Subsystem Integration	Assembling motion systems onto the structure and testing.

TABLE 13-7 SCHEDULE ELECTROMECHANICAL

13.3.4 SYSTEM INTEGRATION CATEGORY:

Module	Definition
Physical Integration	Assembling all three subsystems together.
Testing & Debug & Repair	Test the machine as a whole for functionality, make any adjustment and repair ware-out components.

TABLE 13-8 SCHEDULE INTEGRATION

No breaking further to level 4 tasks because of the plan limitations described previously. A milestone is set on week 6 to guarantee level 4 tasks to be reviewed with TA.

13.4 MILESTONES (LEVEL2)

12 level 2 milestones are set to keep track of the progress: Weekly milestones for TA to check the progress, including all major marking events. Week 7 is missing because it is the reading week.

Milestones	Definition
Submission of Teams	Already done.
Submission of Design Proposal	This proposal.
Week 4 Milestone	User interface (microcontroller) and frame element (electromechanical) are expected to be finished by the end of the lab.
Individual Evaluation 1 (Week 5)	As specified in course notes: microcontroller: user interface, all pseudo code; circuit: sensor circuits, detailed voltage and current analysis; electromechanical: frame, one actuation mechanism.
Interim Notebook Evaluation & Week 6 Milestone	Microcontroller: machine interface, analog to digital module; circuit: all circuits on prototype board with power regulation and protection mechanism; electromechanical: vertical motion

	system.
Individual Evaluation 2 (Week 8)	All subsystems are expected to be finished at this point.
Week 9 Milestone	System is expected to be physically integrated.
Team Evaluation 1 (Week 10)	System is expected to be "more or less" functional.
Week 11 Milestone	System is expected to be functional, except some further calibration.
Team Evaluation 2 (Week 12)	System is expected to be fully functional.
Week 13 Milestone	All supporting materials like documents are expected to be ready for demonstration.
Public Demonstration (Week 14)	Show the machine to the public.

TABLE 13-9 SCHEDULE MILESTONES

13.5 CRITICAL PATH ANALYSIS (PERT)

A PERT chart (Network Diagram in Microsoft Project) is included in the appendix. Due to the fact that the project has so many tasks, the PERT chart is too large to print on one single page and the font is too small to see. The following diagram has only the critical indicated and it should serve our purpose equally well as a PERT chart:

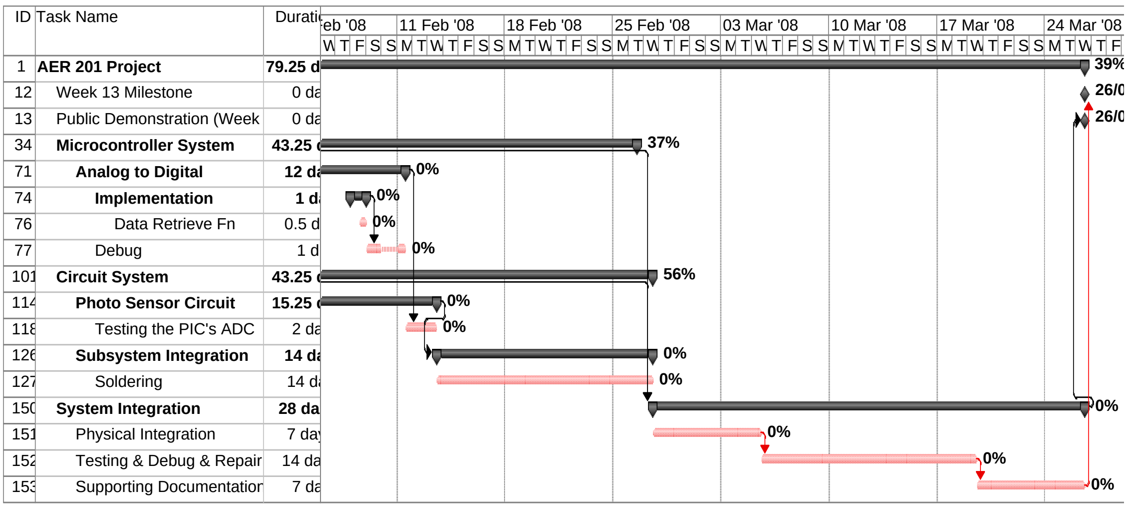


FIGURE 13-2 CRITICAL PATH GANTT CHART

From the critical path Gantt chart, analog-to-digital module appears to be the most critical task of the project. It can be understood that because both microcontroller member and circuit member need to cooperate to finish the task. After ADC module, circuit soldering is the critical path. Be aware of the 14 day duration estimation since it may indicate some prediction limitation as discussed previously. As a result, these critical tasks need to be monitored closely during reviews; moreover, taking as counter measurement, the "division of work" do state that after reading week, the circuit member have some extra time to finish the soldering and microcontroller member takes care of the ADC module, as part of physical integration.

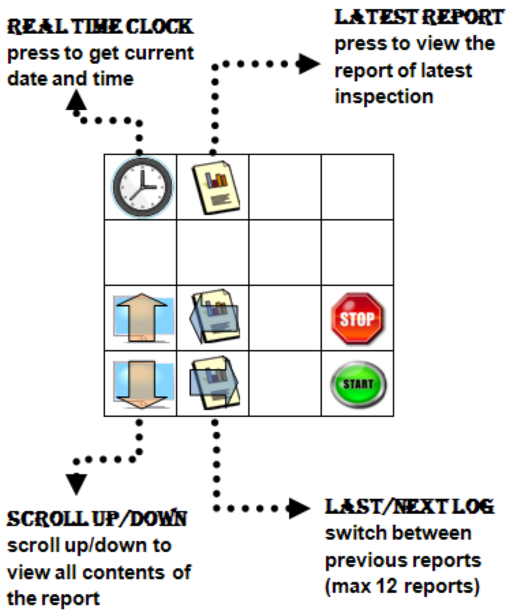
14 STANDARD OPERATING PROCEDURE

14.1 TEST PROCEDURE

Plug external power source if previously not plugged
Open the drawer
Load egg tray on the fixture on the drawer
Make sure all closet lights are more or less facing up
Close the drawer slowly
Press "Start" button
Wait for the machine to finish test
Press page up/down to retrieve test results after test is complete

14.2 CHANGE PIC BOARD BATTERY

Unplug external power source
Turn off internal battery
Replace the used batteries by 6 new AA batteries
Plug external power source
Turn on internal battery
Wait for hardware to reset
Input current time
Turn off external power source if needed



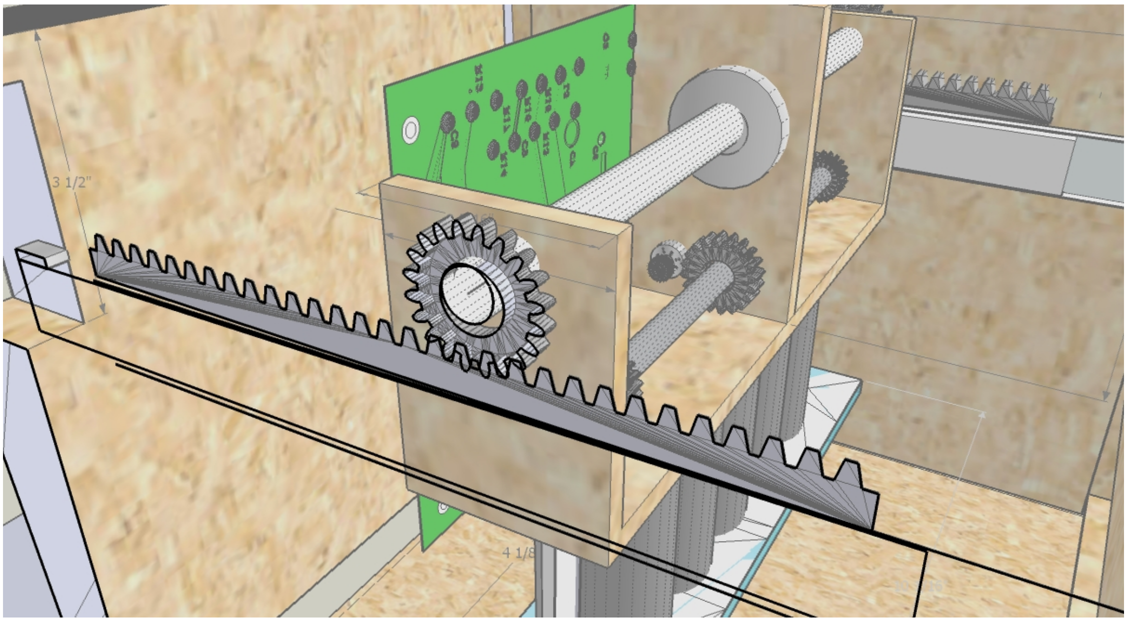
14-1, INTERFACE

15 SYSTEM IMPROVEMENT SUGGESTIONS

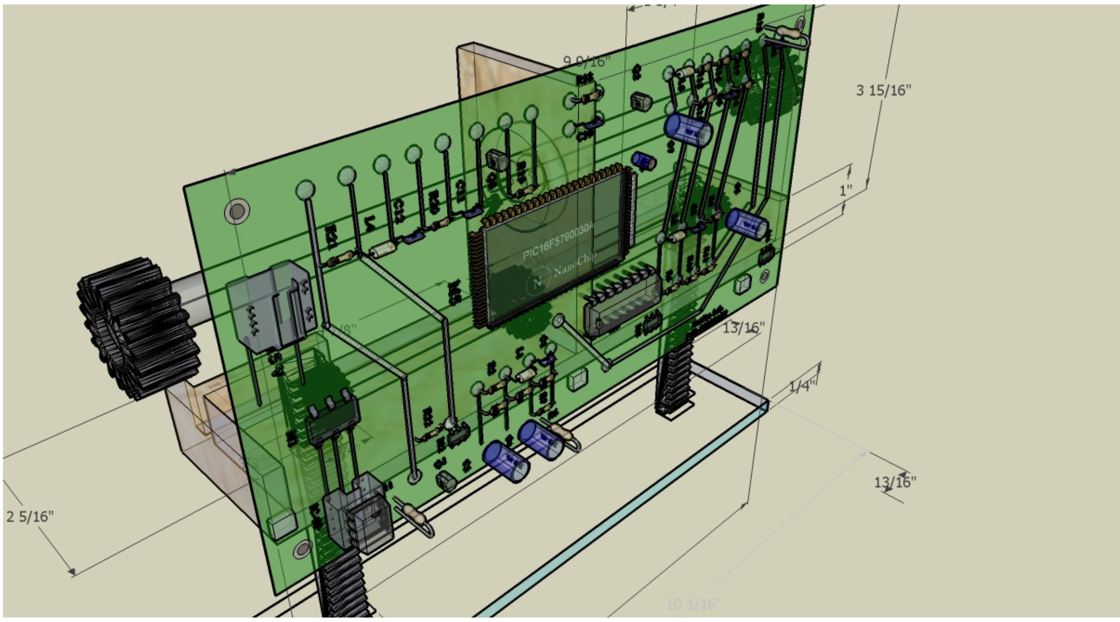
15.1 ISSUES WITH THE CURRENT DESIGN

The wood that was used to build the frame of the robot is very thin and will buckle at very low compression force. Throughout the testing process, it was observed that there are a lot of vibrations that causes the robot to be unstable. A possible to the original frame is to use a heavier and thicker wood. This will provide more support to the moving arm and damp the vibrations cause from it. However if we were to change the frame to thicker wood, this would mean more money spent on the wood and also more weight will be added onto the robot.

In our design both the horizontal and vertical motion is being put together into one moving arm. This is later on found to be very complex and difficult to debug. A possible alternative to this would be allocating the horizontal movement on to the frame. So instead of the arm moving, we will have the egg tray moving into the machine. This will also reduce the complexity of the moving arm, and allow the debugging process to go more smoothly. However, the idea that the egg tray will move into the robot automatically also means that the egg tray needs to be moved out automatically. Thus this will cause the overall run time to increase by 10 seconds.



Currently in our design, the circuit board is being attached onto the back of the moving arm, and this has caused some problems when the moving arm is being repaired. Also in our design, the complexity of the moving arm is far greater than the complexity of the frame and all other parts of the machine. In order to offset this imbalance, it might be a good idea to put the circuit board on the frame instead. However this will cause more wires to be floating around inside the machine connecting the circuit board to the motors which can be hazardous.



In the current design during the testing process, it was found that our machine still has trouble determining the difference between lights with 1 LED on and lights with 3 LED on. This shows that the calibration on the threshold of the light sensors is insufficient. The machine might be too sensitive to the change in environment and requires more calibration.

Another possible design that was considered was to have 20 light sensors to press all the lights at once. This idea will significantly simplify the electromechanical portion of the project in exchange for the circuit portion. Also the run time of the checking process will also be reduced significantly, since the machine will only be required to press up and down. However the downside of doing this would be the complexity of the circuit part. Considering how there will be 20 light sensors that will require calibration, there is a good chance that something will go wrong in the circuits. Also, this would cause the reliability of the machine to decrease significantly.

15.2 IMPROVEMENT SUGGESTIONS

Many things could be improved, the following is identified.

1. The huge base could have been eliminated if we used thicker boards
2. Power supply could be hidden
3. Metal gear should be used for heavily loaded transmissions
4. Stranded wires should be used for ALL inter-board components
5. Very consistent and uniform colour coding should be applied
6. Sensors can be calibrated to a more optimized fashion

7. The wood used to build the frame was too thin, should be replaced by more durable wood.

8. Areas of improvement of the PIC

Please refer to the respective sections of subsystems for more detail since system improvements is mostly dependent on its subsystems.

16 CONCLUSION

Simplex is a robotic prototype that was designed and built over the past 4 months to solve the closet light inspection problem. The closet light inspection problem calls for a machine that can quickly and reliably determine and display the functionality and position of closet lights placed on an egg tray.

Simplex takes a tray of closet lights. It inspects the tray row-by-row. It turns the lights on and off using an arm that cover all possible positions. Both functionality and position sensors are implemented using the same 5 phototransistors. It takes about 20 seconds to load the tray and it has a user-friendly interface. The microcontroller (PIC16F877), the circuits, the sensors and two motors are all placed on a testing arm that moves row-by-row. The horizontal motion of the testing arm is powered by a stepper motor for accuracy and the vertical motion of the testing arm is powered by a DC motor for power.

Simplex has the following distinctive features,

- Simple circuit system, no logic elements used, no op-amps used
- Customized PIC proto64 board
- Single phototransistor for both position by infrared reflection and functionality detection by intensity threshold
- Continuous operation despite row-by-row design (i.e. the starting position of the testing arm for the second run is the end position of the first run)
- Hold up to 12 logs with date and time accurate to seconds
- PIC is powered by 6 x 1.5V battery
- Isolated testing environment, black box. Operates under any external lighting condition
- Intuitive display of results in the form of a matrix,

  XXXXP
XXXXX
XXFXX
XXXPX

  where P stands for pass, X for empty, F for fail.

The finished machine performed fairly well during the demo. The machine correctly determined the positions of 5 out of 5 lights and correctly determined the functionalities of 4 out of 5 lights in each of the two trials in 40 seconds and 30 seconds. The machine is fairly reliable since it went through about 15 successful full runs and over 100 test runs before the demo.

However, Simplex is not perfect; it often passes lights that have only one or two bright LED(s). This is due the value of sensor threshold, a balance between the risk of failing closet lights with 3 LEDs and the risk of passing lights with less than 3 LED(s) need to be carefully studied in order to improve the results. Important ways by which the machine can be improved was discussed in suggestions for improvement sections.

Simplex expresses our vision of creating a simple machine that solves a complex problem. However, even though we went for the simplest solution we can conceive, various complications still arise in the design process and that's why we chose a complicated font for it.

All in all, Simplex is quite a success in our opinion despite the many ups and downs we went through with it and the learning experience should benefit our future careers in engineering.

17 REFERENCE

[1] M.R. Emami, Engineering Design. Toronto: University of Toronto
[2] P. E. Harris, Planning and Scheduling Using Microsoft Office Project 2007: Including Microsoft Project 2000 to 2003. Eastwood Harris Pty Ltd, 2007.
[3] M.R. Emami, PIC DevBugger Manual. 2007
[4] Microchip Technology Inc, PIC16F87X Data Sheet. 2001
[5] Microchip Technology Inc, PICmicro™ Mid-Range MCU Family Reference Manual. 1997
[6] Wikipedia, The Free Encyclopedia, "Quality Control" [online document], 27 January 2008, http://en.wikipedia.org/w/index.php?title=Special:Cite&page=Quality_control&id=187415113
[7] Wikipedia, The Free Encyclopedia, "Waterfall model" [online document], 27 January 2008, http://en.wikipedia.org/wiki/Waterfall_model
[8] Wikipedia, The Free Encyclopedia, "Intel Core 2" [online document], 27 January 2008, http://en.wikipedia.org/wiki/Core_2_duo
[9] "Lifemax Limited – About Us" [online document] http://www.lifemaxuk.co.uk/cgi-bin/showpage.pl
[10]Some types of Gear" [online document], 13 April 2008, http://www.fi.edu/time/Journey/Time/Escapements/geartypes.html
[11]"MECHANICAL MOVEMENTS POWERS AND DEVICES " [online document], 13 April 2008, http://knowledgepublications.com/history/mechanical_movements_detail.htm
[12]"Press screw assembly ACME screw shafts in sizes 1" - 1 1/8" - 1 1/4"" [online document], 13 April 2008, http://www.applejournal.com/correll/const.htm
[13]"Hardware online" [online document], 13 April 2008, http://elraco.com.au/
[14]APPLICATION NOTE 58: Crystal Considerations with Dallas Real-Time Clocks (RTCs) [online document], 13 April 2008, http://www.maxim-ic.com/appnotes.cfm/an_pk/58

18 TABLE OF FIGURES AND TABLES

18.1 TABLE OF FIGURES

18.2 TABLE OF TABLES

APPENDIX CONTENTS

ELECTTROMECH APPENDIX

CIRCUIT APPENDIX

PIC APPENDIX

ZHENG D.C MOTOR SPECIFICATIONS

Shaft Speed		50 RPM
Rated Voltage		12.0 V
Operating Voltage Range		4.5 – 18 V
Operating Current	No load	70 mA
	Full load	1380 mA
Stall Torque		28 kg/inch
Average Power Draw		8 W

NMB-MAT

Minebea Motor Manufacturing Corporation

eMINEBEA.COM

PM55L-048

PM Motor

PM Type

Reference Characteristics

Motor Size	PM55L-048	
Number of Steps per Rotation	48(7.5°/Step)	
Drive Method	2-2 PHASE	
Drive Circuit	UNIPOLAR CONST. VOLT.	BIPOLAR CHOPPER
Drive Voltage	24[V]	24[V]
Current/Phase		*800[mA]*
Coil Resistance/Phase	30[Ω]	5.5[Ω]
Drive IC	2SC3346	UDN2916B-V
Magnet Material	Ferrite plastic magnet (MSPL) Polar anisotropy ferrite sintered magnet (MS50) Nd-Fe-B bonded magnet (MS70)	
Insulation Resistance	100M[Ω] MIN	
Dielectric Strength	AC 500[V] 1[min]	
Class of Insulation	CLASS E	
Operating Temp.	-10[°C] ~ 50[°C]	
Storage Temp.	-30[°C] ~ 80[°C]	
Operating Hum.	20[%] RH ~ 90[%] RH	

Applications

OA Equipment : Printers / Scanners

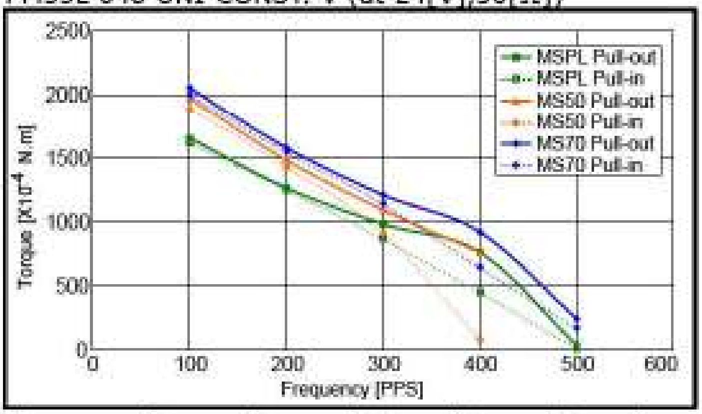
Industrial equipment : Flow control valves

Toys : Slot machines

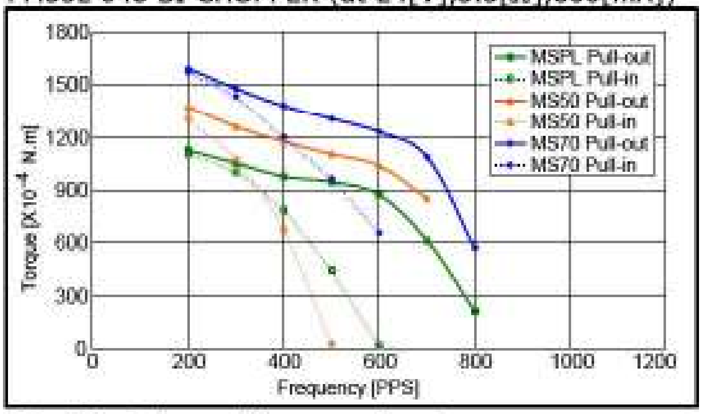
Home automation appliances : Sewing machines

Torque Characteristics

PM55L-048 UNI-CONST. V (at 24[V],30[Ω])

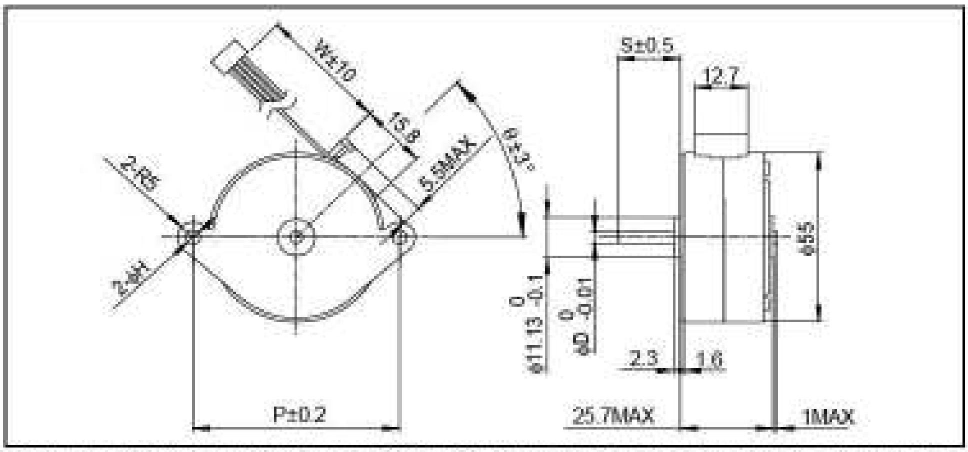


PM55L-048 BI-CHOPPER (at 24[V],5.5[Ω],800[mA])



These torque values are reference only. Heat radiation conditions and temperature rise effect by duty are different on each equipment, therefore please select motors after considering the heat conditions in the actual equipment.

Dimensions



If you would like to know this Dimensions(D,S,W,θ), Please see Standerd Dimensions in our Home Page.

By measurements, we determine Is, R and frequencies as follows for the 100rpm motor used.

$> I1 := 0.07;\ I2 := 0.30;\quad \omega 1 := \frac{2 \cdot \pi \cdot 100}{60};\ \omega 2 := \frac{2 \cdot \pi \cdot 25}{19};\ V := 11.25;\ R := 6;$

$$I1 := 0.07$$
$$I2 := 0.30$$
$$\omega 1 := \frac{10}{3}\pi$$
$$\omega 2 := \frac{50}{19}\pi$$
$$V := 11.25$$
$$R := 6 \tag{1}$$

$> \text{Ratio1} := \frac{solve(I1 \cdot V = I1^2 \cdot R + \tau 1 \cdot \omega 1,\ \tau 1)}{I1};$

$$Ratio1 := 1.034188820 \tag{2}$$

$> \text{Ratio2} := \frac{solve(\text{I2} \cdot V = I2^2 \cdot R + \tau 2 \cdot \omega 2,\ \tau 2)}{I2};$

$$Ratio2 := 1.143050801 \tag{3}$$

>

By measurements, we determine Is, R and frequencies as follows for the 50rpm motor used.

$> I1 := 0.0172;\ I2 := 0.22;\quad \omega 1 := \frac{2 \cdot \pi \cdot 50}{60};\ \omega 2 := 2 \cdot \pi \cdot 0.526;\ V := 11.25;\ R := 18;$

$$I1 := 0.0172$$
$$I2 := 0.22$$
$$\omega 1 := \frac{5}{3}\pi$$
$$\omega 2 := 1.052\,\pi$$
$$V := 11.25$$
$$R := 18 \tag{4}$$

$> \text{Ratio1} := \frac{solve(I1 \cdot V = I1^2 \cdot R + \tau 1 \cdot \omega 1,\ )}{I1};$

$$Ratio1 := 2.089462487 \tag{5}$$

$> \text{Ratio2} := \frac{solve(\text{I2} \cdot V = I2^2 \cdot R + \tau 2 \cdot \omega 2,\ \tau 2)}{I2};$

$$Ratio2 := 2.205778584 \tag{6}$$

>

> $P = \frac{V^2}{R}$; $Pmax = \frac{V^2}{Rmax}$;

$$P = \frac{V^2}{R}$$
$$Pmax = \frac{V^2}{Rmax} \quad (1)$$

> $StepperPower := \frac{2 \cdot 11.25^2}{30}$;

$$StepperPower := 8.437500000 \quad (2)$$

> $DCPower := \frac{11.25^2}{(6+12)}$;

$$DCPower := 7.031250000 \quad (3)$$

> $SensorPower := 11.25 \cdot 0.1$

$$SensorPower := 1.125 \quad (4)$$

> $TotalPower := StepperPower + DCPower + SensorPower$;

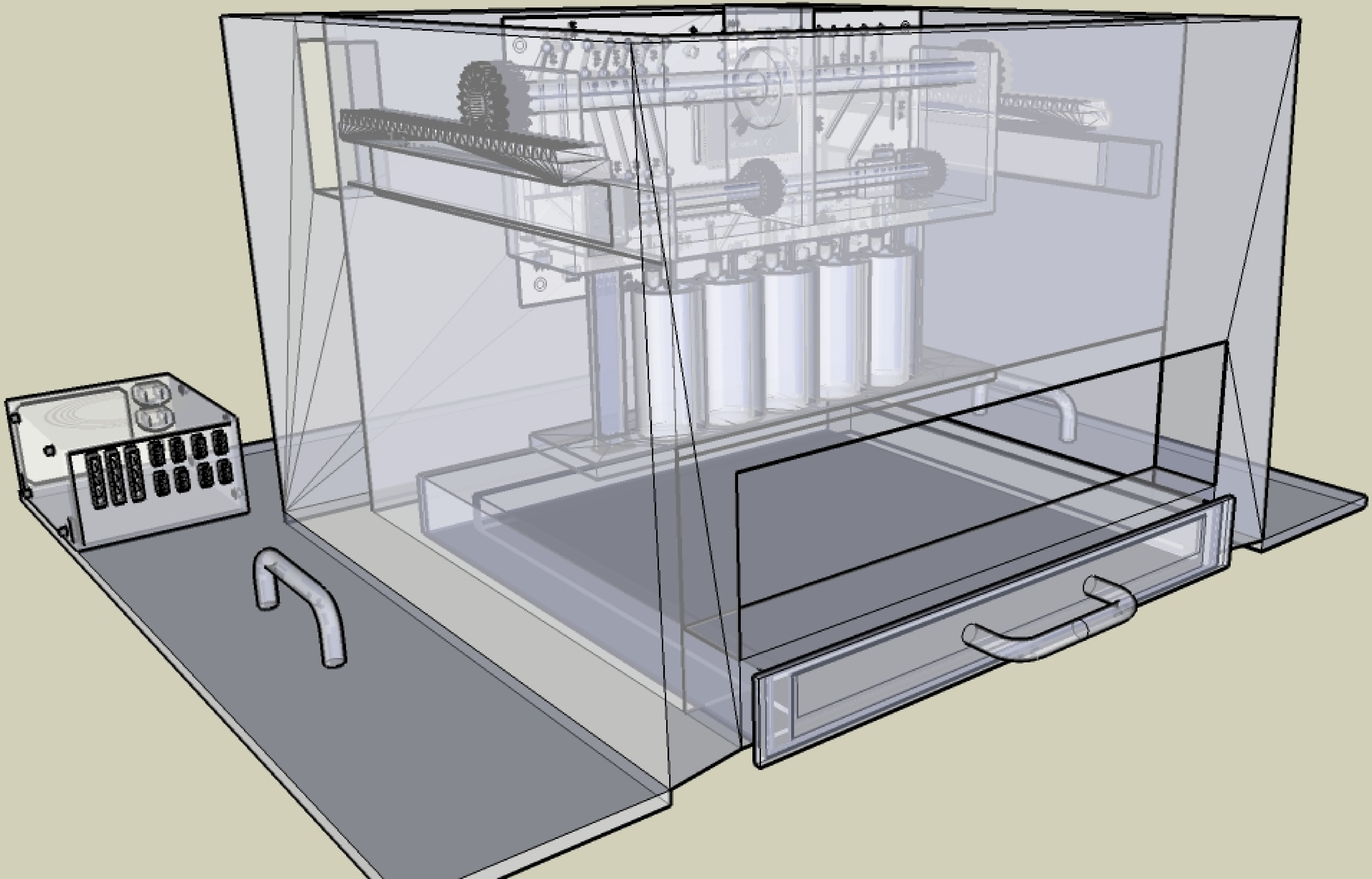
$$TotalPower := 16.59375000 \quad (5)$$

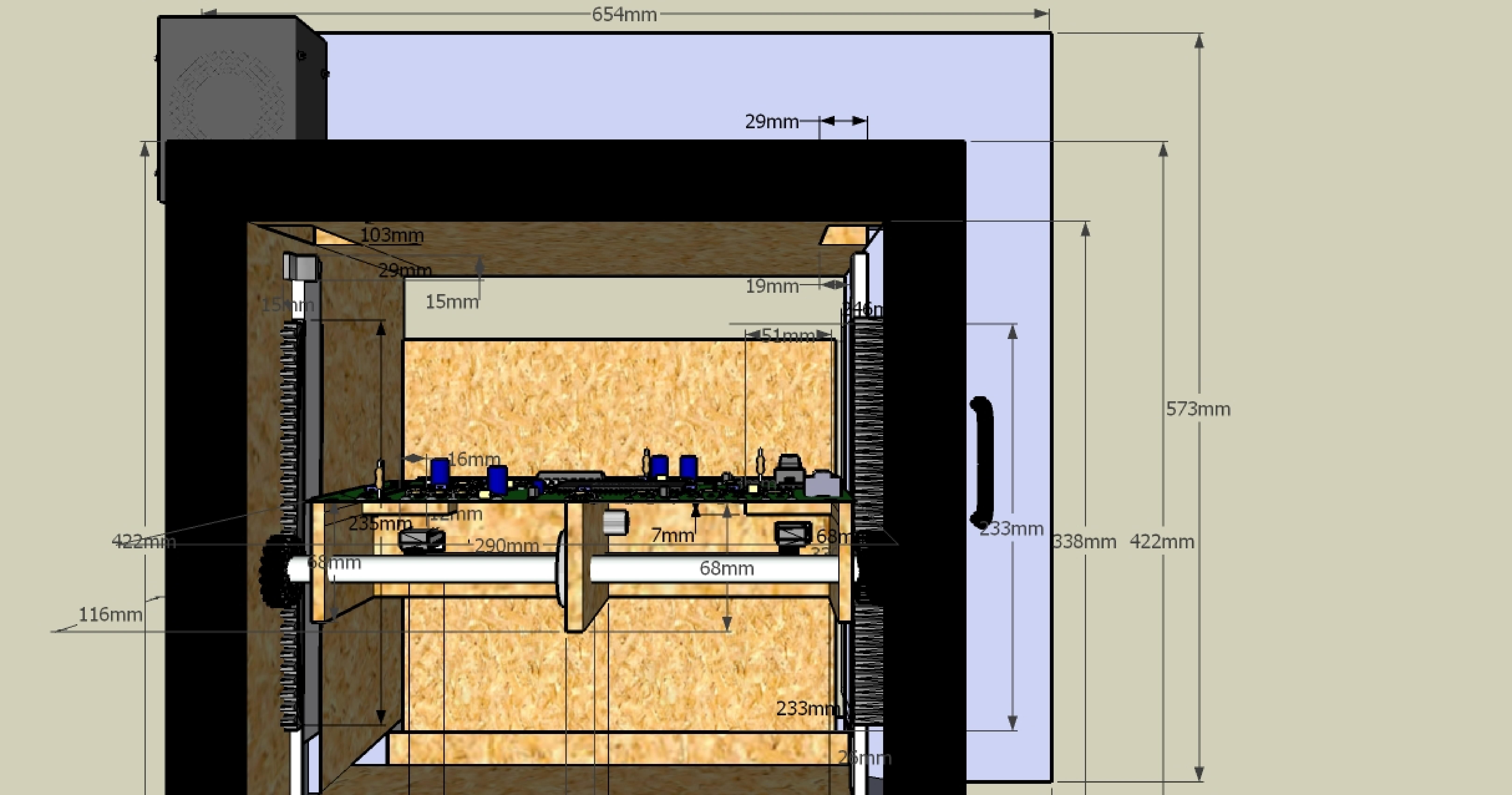
> $TotalCurrent := \frac{11.25}{30} + \frac{11.25}{6+12} + 0.1$;

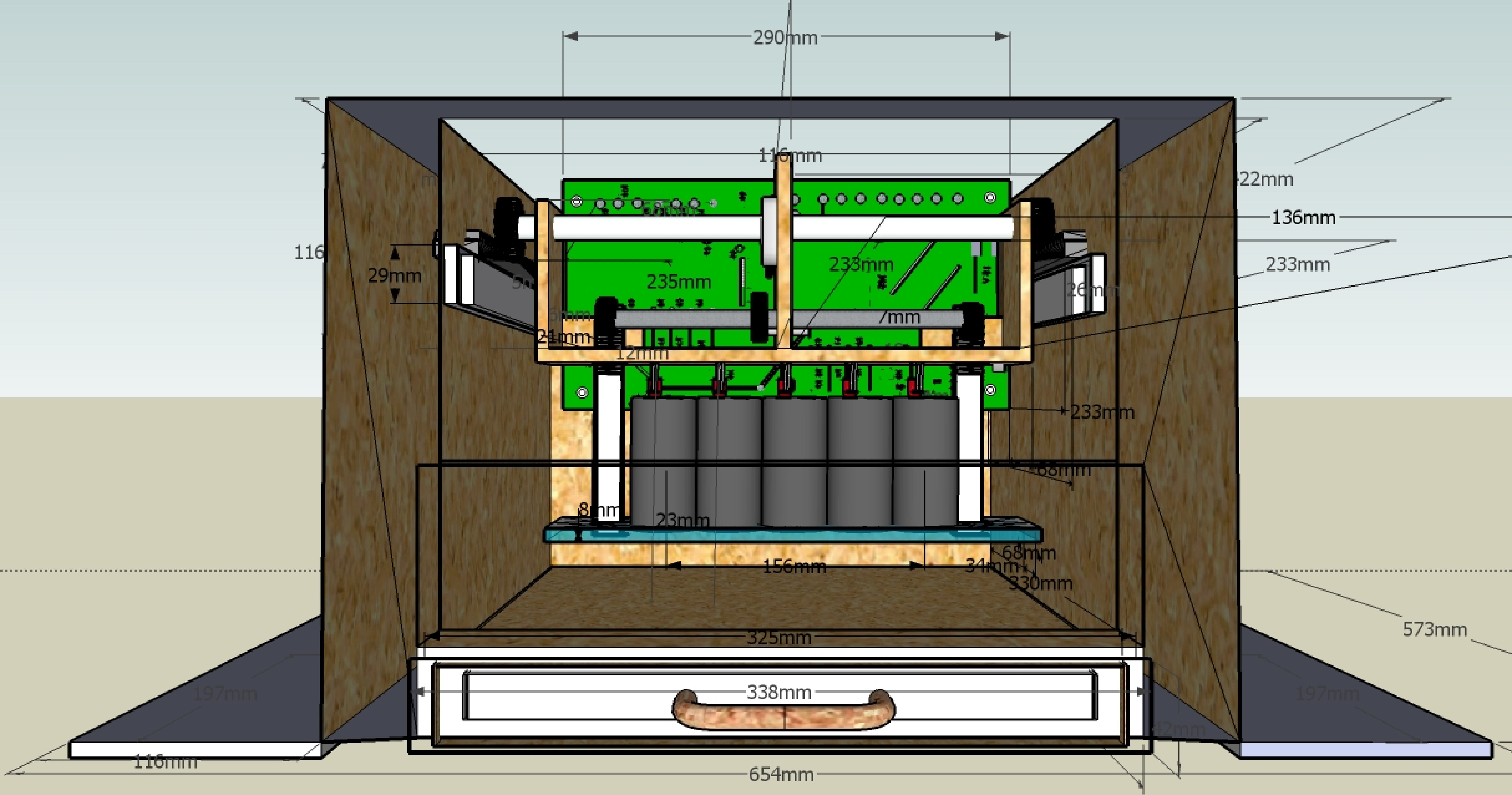
$$TotalCurrent := 1.100000000 \quad (6)$$

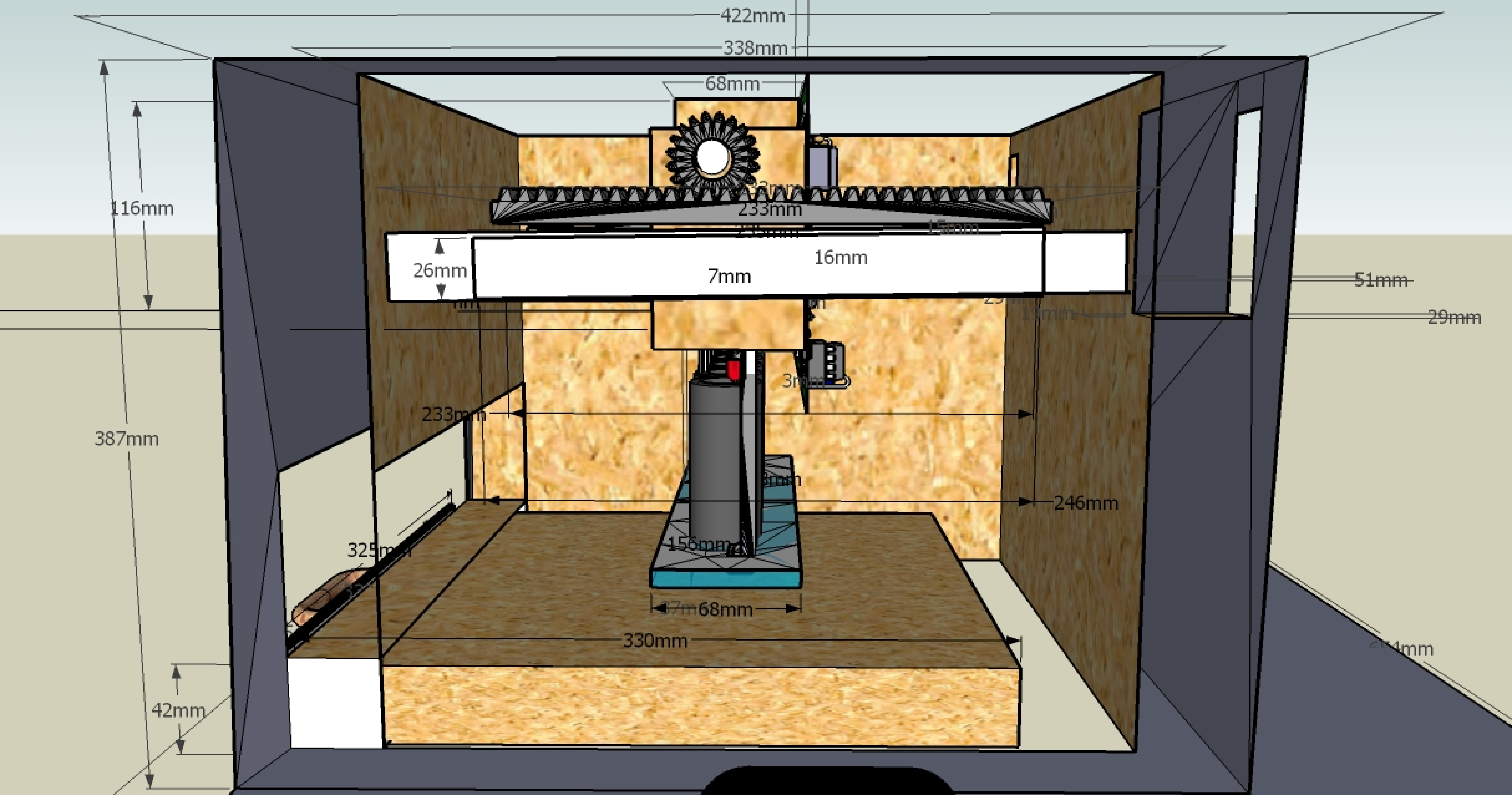
>

Note that this is much smaller than the 180W limit of the power supply.
The peak current drawn is about 1A, this should be considered safe.




654mm
29mm
103mm
29mm
15mm
15mm
19mm
51mm
16mm
235mm
7mm
290mm
68mm
68mm
68mm
422mm
116mm
233mm
233mm
338mm
422mm
573mm
26mm


290mm
116mm
136mm
233mm
235mm
233mm
29mm
21mm
12mm
233mm
68mm
8mm
23mm
68mm
156mm
34mm
330mm
325mm
338mm
573mm
197mm
197mm
116mm
654mm



422mm
338mm
68mm
116mm
233mm
16mm
26mm
7mm
51mm
29mm
3mm
233mm
387mm
246mm
325mm
156mm
68mm
330mm
42mm

CIRCUIT APPENDIX

1, LTE5208 Infrared emitter

2, PT481 Phototransistor

3, L293d quad H-bridge driver

4, TIP142

5, TIP147

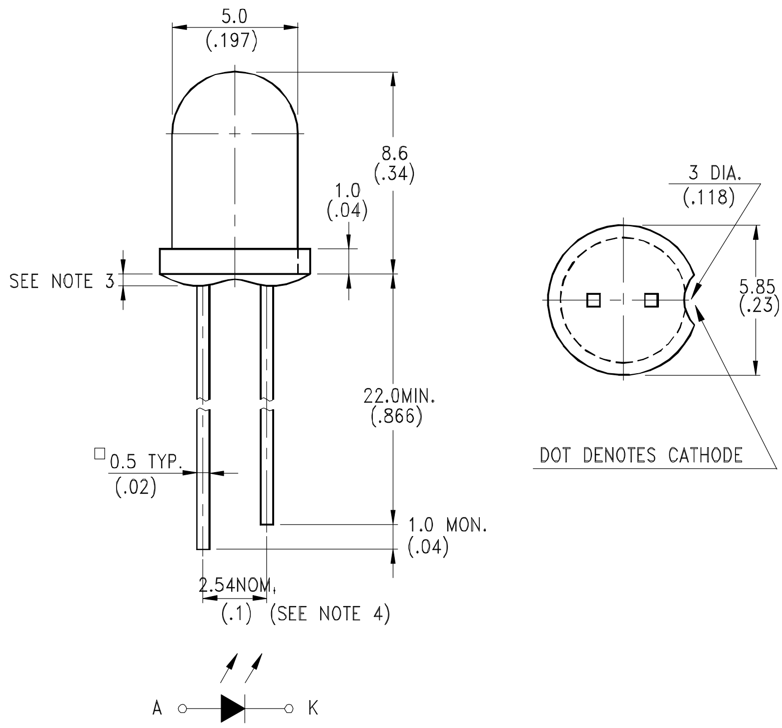
6, IPS-1806 180W Power Supply

7, Determining Resistor Value

FEATURES

* SELECTED TO SPECIFIC ON-LINE INTENSITY AND RADIANT INTENSITY RANGES
* LOW COST MINIATURE PLASTIC END LOOKING PACKAGE
* MECHANICALLY AND SPECTRALLY MATCHED TO THE LTR-3208 SERIES OF PHOTOTRANSISTOR
* CLEAR TRANSPARENT COLOR PACKAGE

PACKAGE DIMENSIONS



NOTES:
1. All dimensions are in millimeters (inches).
2. Tolerance is ± 0.25mm(.010") unless otherwise noted.
3. Protruded resin under flange is 1.5mm(.059") max.
4. Lead spacing is measured where the leads emerge from the package.
5. Specifications are subject to change without notice.

ABSOLUTE MAXIMUM RATINGS AT TA=25℃

PARAMETER	MAXIMUM RATING	UNIT
Power Dissipation	150	mW
Peak Forward Current (300pps, 10 μs pulse)	2	A
Continuous Forward Current	100	mA
Reverse Voltage	5	V
Operating Temperature Range	-40℃ to + 85℃	
Storage Temperature Range	-55℃ to + 100℃	
Lead Soldering Temperature [1.6mm(.063") From Body]	260℃ for 5 Seconds	

ELECTRICAL OPTICAL CHARACTERISTICS AT TA=25℃

PARAMETER	SYMBOL	MIN.	TYP.	MAX.	UNIT	TEST CONDITION	BIN NO.
Aperture Radiant Incidence	Ee	0.44		0.96	mW/cm²	I_F = 20mA	BIN A
		0.64		1.20			BIN B
		0.80		1.68			BIN C
		1.12					BIN D
Radiant Intensity	I_E	3.31		7.22	mW/sr	I_F = 20mA	BIN A
		4.81		9.02			BIN B
		6.02		12.63			BIN C
		8.42					BIN D
Peak Emission Wavelength	λ_{Peak}		940		nm	I_F = 20mA	
Spectral Line Half-Width	$\Delta\lambda$		50		nm	I_F = 20mA	
Forward Voltage	V_F		1.2	1.6	V	I_F = 20mA	
Reverse Current	I_R			100	μA	V_R = 5V	
Viewing Angle (See FIG.6)	$2\theta_{1/2}$		40		deg.		

TYPICAL ELECTRICAL / OPTICAL CHARACTERISTICS CURVES

(25℃ Ambient Temperature Unless Otherwise Noted)

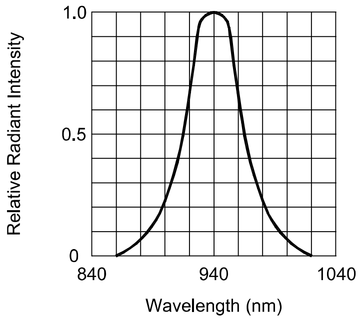


FIG.1 SPECTRAL DISTRIBUTION

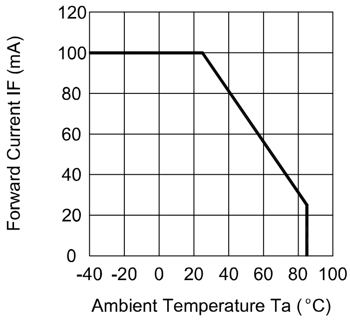


FIG.2 FORWARD CURRENT VS. AMBIENT TEMPERATURE

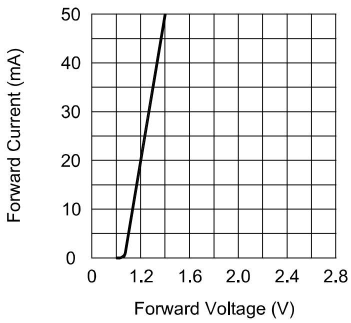


FIG.3 FORWARD CURRENT VS. FORWARD VOLTAGE

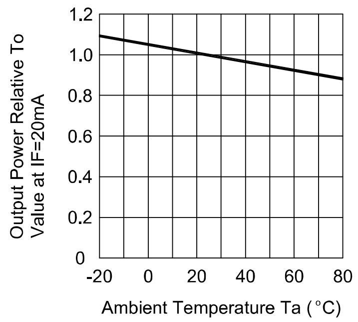


FIG.4 RELATIVE RADIANT INTENSITY VS. AMBIENT TEMPERATURE

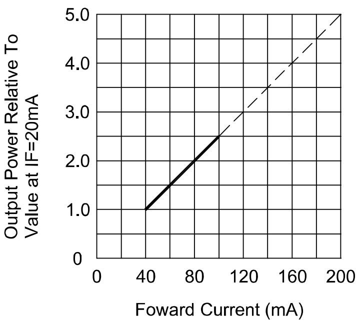


FIG.5 RELATIVE RADIANT INTENSITY VS. FORWARD CURRENT

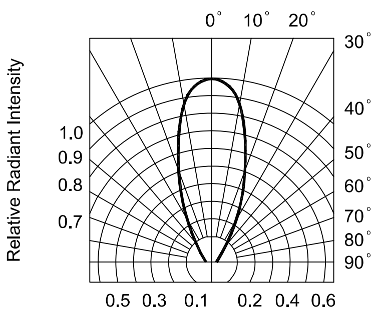


FIG.6 RADIATION DIAGRAM

PT481/PT481F/ PT483F1

Narrow Acceptance High Sensitivity Phototransistor

■ Features

1. Epoxy resin package
2. Narrow acceptance ($\Delta\theta$: Typ. ± 13˚)
3. High sensitivity
   (I_C : MIN. 1.5mA at E_e = 0.1mW/cm^2) : **PT481/PT483F1**
   (I_C : MIN. 0.9mA at E_e = 0.1mW/cm^2) : **PT481F**
4. Visible light cut-off type : **PT481F/PT483F1**
5. Long lead pin type : **PT483F1**

■ Applications

1. VCRs, cassette tape recorders
2. Floppy disk drives
3. Optoelectronic switches
4. Automatic stroboscopes

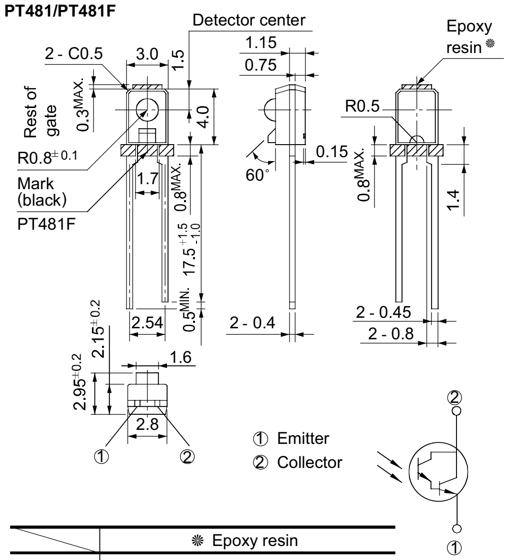
■ Absolute Maximum Ratings (Ta = 25˚C)

Parameter	Symbol	Rating	Unit
Collector-emitter voltage	V_{CEO}	35	V
Emitter-collector voltage	V_{ECO}	6	V
Collector current	I_C	50	mA
Collector power dissipation	P_C	75	mW
Operating temperature	T_{opr}	- 25 to +85	˚C
Storage temperature	T_{stg}	- 40 to +85	˚C
*1Soldering temperature	T_{sol}	260	˚C

*1 For 3 seconds at the position of 1.4mm from the bottom face of resin package

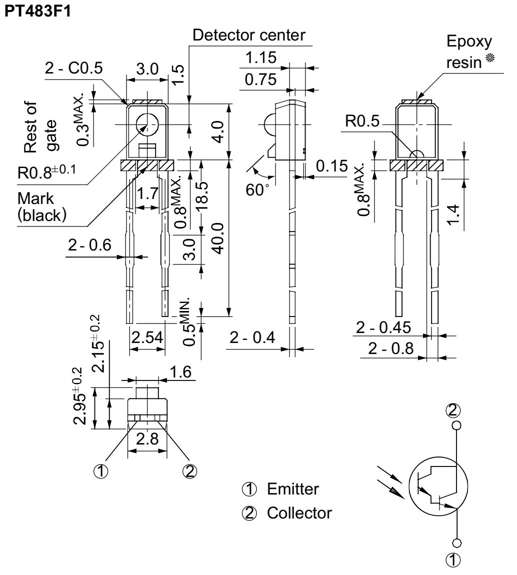
■ Outline Dimensions (Unit : mm)

PT481/PT481F



	❋ Epoxy resin
PT481	Light blue transparent resin
PT481F	Visible light cut-off resin (black)

PT483F1



❋ Visible light cut-off resin (black)

■ Electro-optical Characteristics

(Ta = 25°C)

Parameter		Symbol	Conditions	MIN.	TYP.	MAX.	Unit
*2 Collector current	**PT481**	I_C	V_{CE} = 2V, E_e = 0.1mW/cm^2	1.5	10	25	mA
	PT481F			0.9	-	27	mA
	PT483F1			1.5	-	4.0	mA
Collector dark current		I_{CEO}	V_{CE} = 10V, E_e = 0	-	-	10^{-6}	A
*2 Collector-emitter saturation voltage		$V_{CE(sat)}$	I_c = 2.5mA, E_e = 1mW/cm^2	-	0.7	1.0	V
Peak emission wavelength	**PT481**	λ_p	-	-	800	-	nm
	PT481F/PT483F1			-	860	-	nm
Response time	Rise time	t_r	V_{CE} = 2V, I_C = 10mA, R_L = 100Ω	-	80	-	μs
	Fall time	t_f		-	70	-	μs

*2 E_e : Irradiance by CIE standard light source A (tungsten lamp)

Fig. 1 Collector Power Dissipation vs. Ambient Temperature

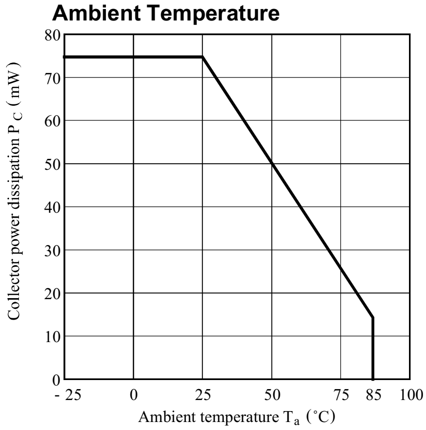


Fig. 2 Collector Dark Current vs. Ambient Temperature

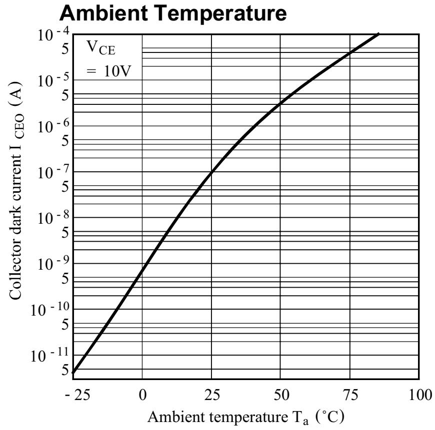


Fig. 3 Relative Collector Current vs. Ambient Temperature

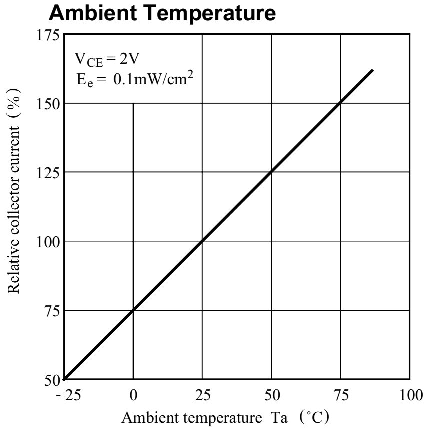


Fig.4-a Collector Current vs. Irradiance (PT481)

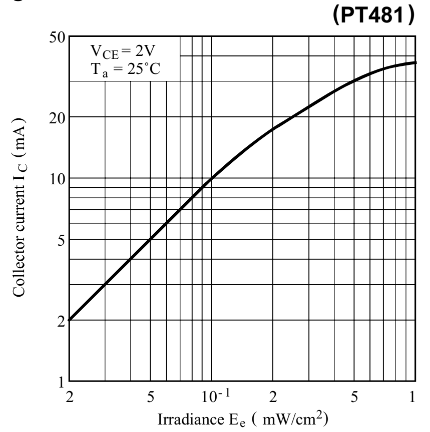


Fig.4-b Collector Current vs. Irradiance (PT481F/PT483F1)

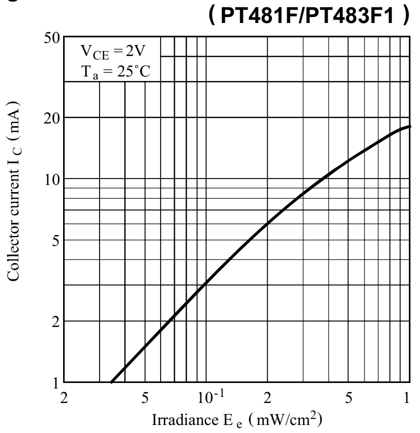


Fig.5-a Collector Current vs. Collector-emitter Voltage (PT481)

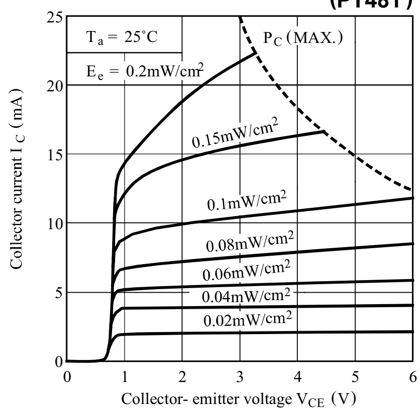


Fig.5-b Collector Current vs. Collector-emitter Voltage (PT481F/PT483F1)

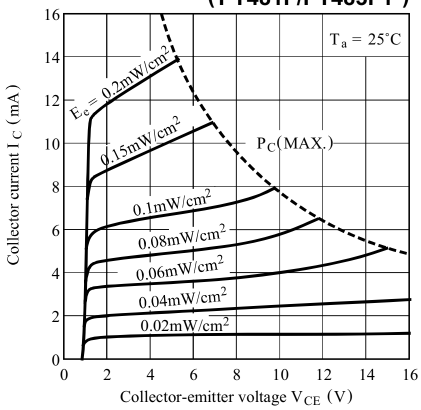


Fig. 6 Spectral Sensitivity

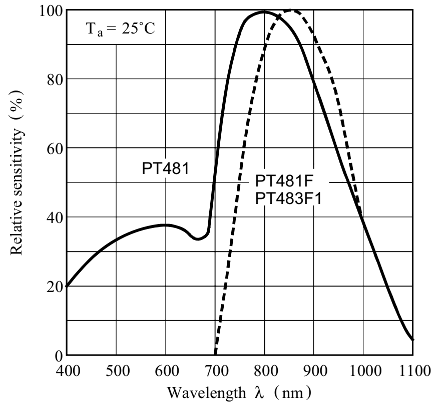
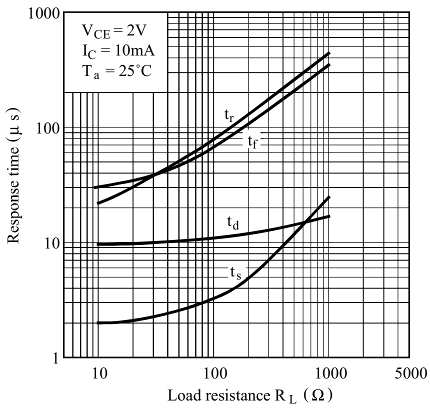


Fig. 7 Response Time vs. Load Resistance



Test Circuit for Response Time

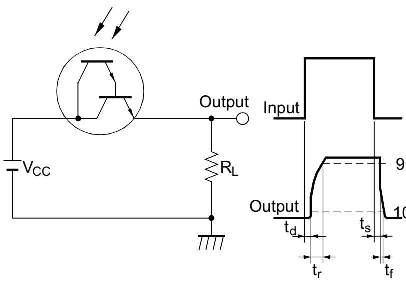


Fig. 8 Sensitivity Diagram

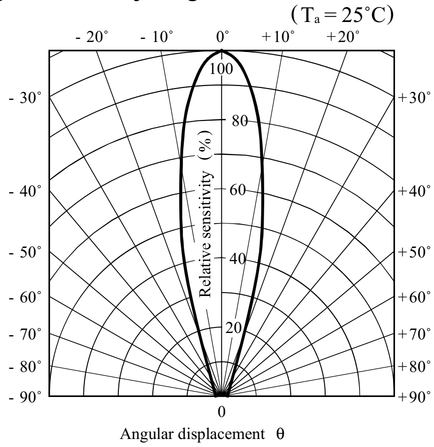


Fig.9-a Collector-emitter Saturation Voltage vs. Irradiance (PT481)

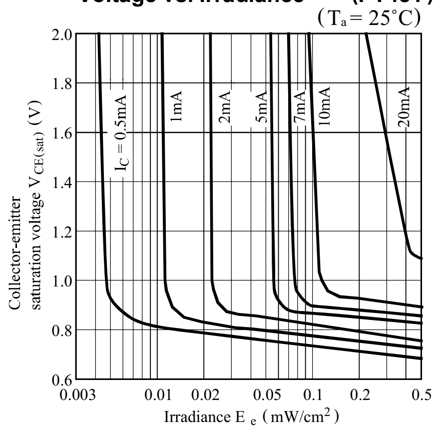


Fig.9-b Collector-emitter Saturation Voltage vs. Irradiance (PT481F/PT483F1)

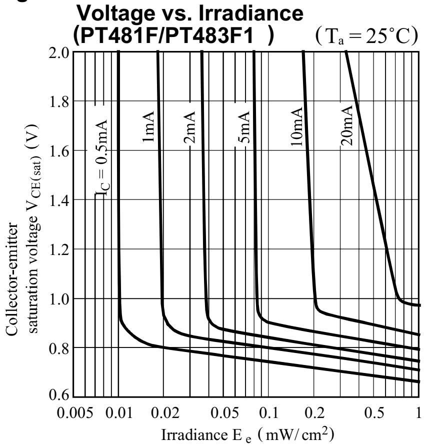
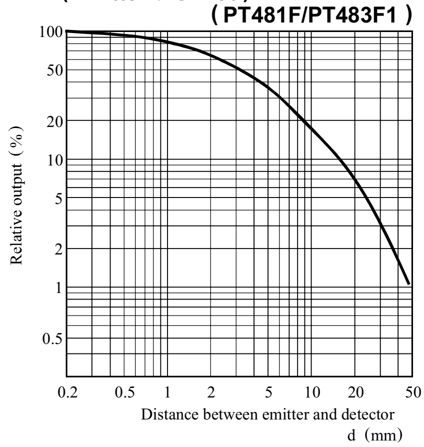


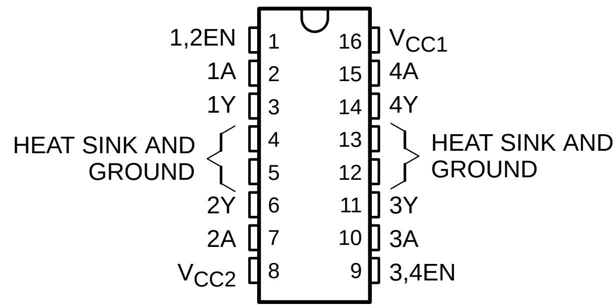
Fig.10 Relative Output vs. Distance (Emitter : GL480) (PT481F/PT483F1)



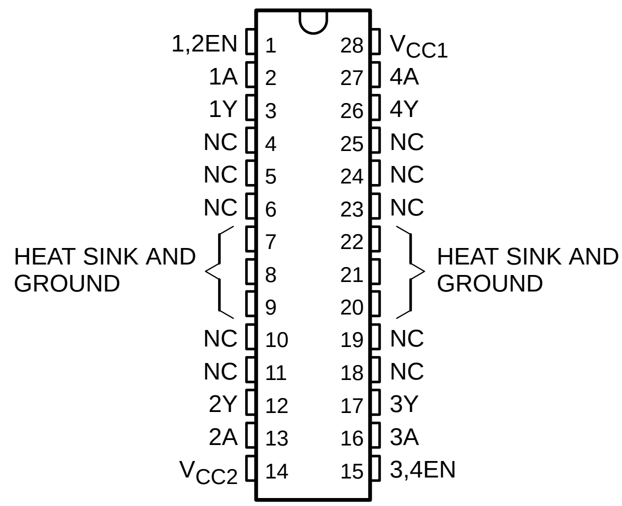
● Please refer to the chapter " Precautions for Use."

- **Featuring Unitrode L293 and L293D Products Now From Texas Instruments**
- **Wide Supply-Voltage Range: 4.5 V to 36 V**
- **Separate Input-Logic Supply**
- **Internal ESD Protection**
- **Thermal Shutdown**
- **High-Noise-Immunity Inputs**
- **Functional Replacements for SGS L293 and SGS L293D**
- **Output Current 1 A Per Channel (600 mA for L293D)**
- **Peak Output Current 2 A Per Channel (1.2 A for L293D)**
- **Output Clamp Diodes for Inductive Transient Suppression (L293D)**

N, NE PACKAGE
(TOP VIEW)



DWP PACKAGE
(TOP VIEW)



description

The L293 and L293D are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are 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.

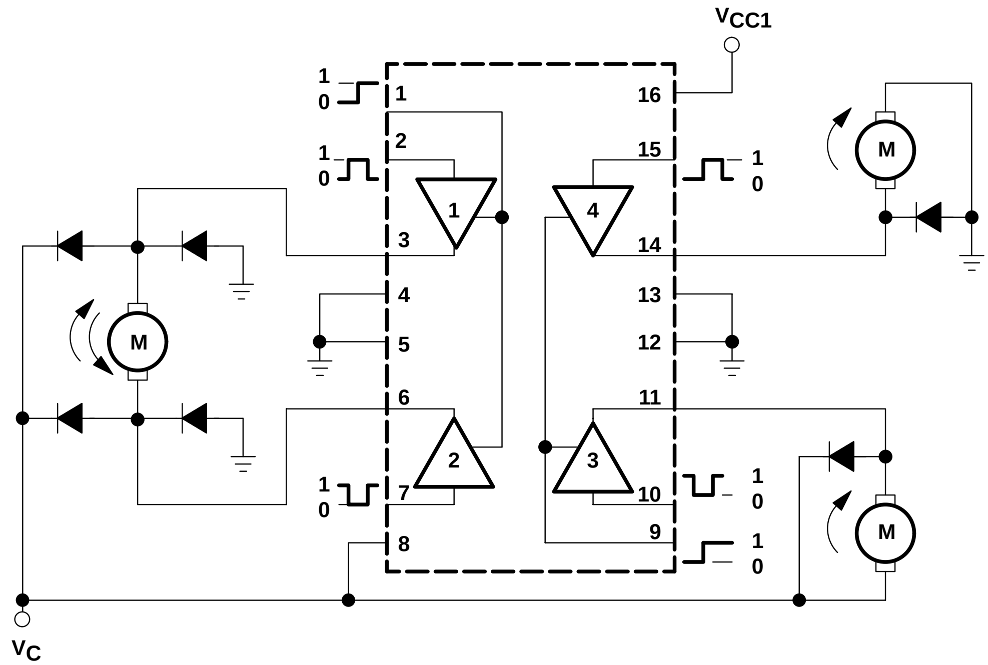
All inputs are TTL compatible. Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers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

On the L293, external high-speed output clamp diodes should be used for inductive transient suppression.

A V_{CC1} terminal, separate from V_{CC2}, is provided for the logic inputs to minimize device power dissipation.

The L293and L293D are characterized for operation from 0°C to 70°C.

block diagram



NOTE: Output diodes are internal in L293D.

TEXAS INSTRUMENTS
AVAILABLE OPTIONS

T_A	PACKAGE PLASTIC DIP (NE)
0°C to 70°C	L293NE L293DNE

Unitrode Products from Texas Instruments

AVAILABLE OPTIONS

T_A	PACKAGED DEVICES	
	SMALL OUTLINE (DWP)	PLASTIC DIP (N)
0°C to 70°C	L293DWP L293DDWP	L293N L293DN

The DWP package is available taped and reeled. Add the suffix TR to device type (e.g., L293DWPTR).

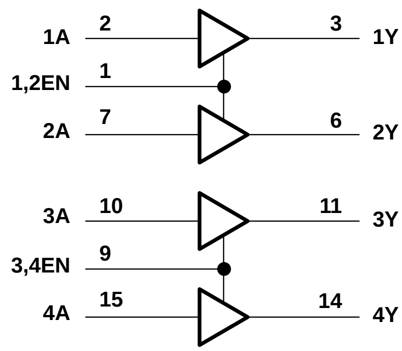
FUNCTION TABLE
(each driver)

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

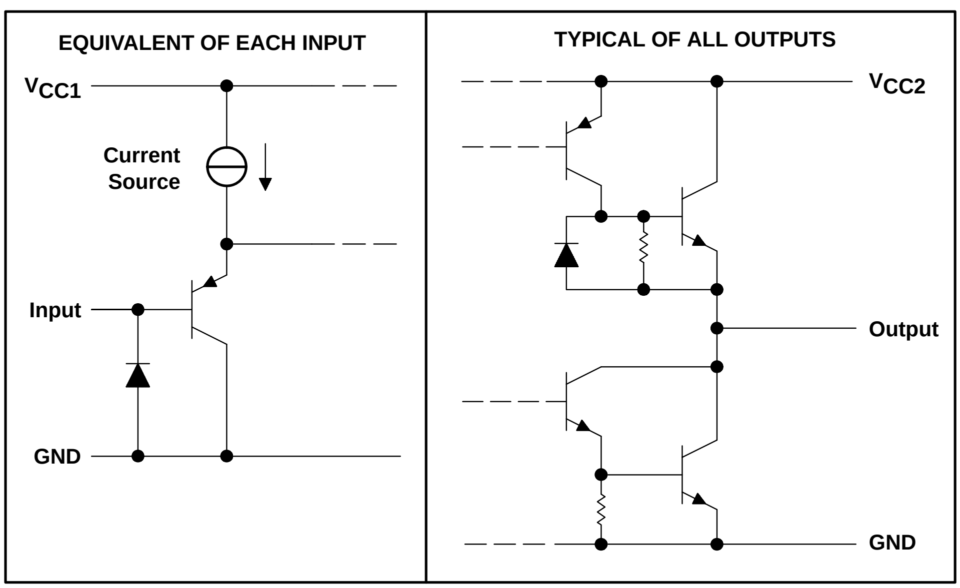
H = high level, L = low level, X = irrelevant, Z = high impedance (off)

† In the thermal shutdown mode, the output is in the high-impedance state, regardless of the input levels.

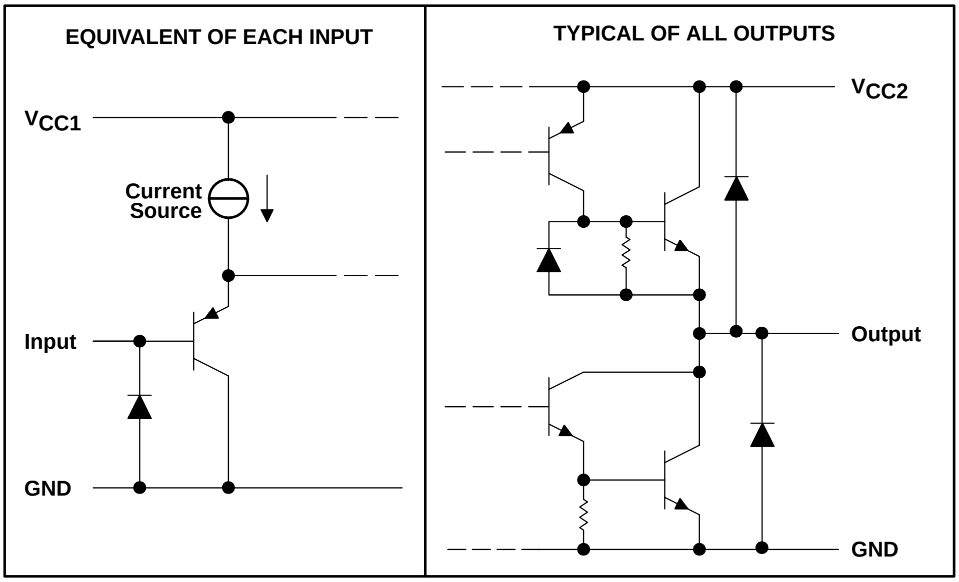
logic diagram



schematics of inputs and outputs (L293)



schematics of inputs and outputs (L293D)



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

Supply voltage, V_{CC1} (see Note 1)	36 V
Output supply voltage, V_{CC2}	36 V
Input voltage, V_I	7 V
Output voltage range, V_O	–3 V to V_{CC2} + 3 V
Peak output current, I_O (nonrepetitive, $t \le 5$ ms): L293	±2 A
Peak output current, I_O (nonrepetitive, $t \le 100$ µs): L293D	±1.2 A
Continuous output current, I_O: L293	±1 A
Continuous output current, I_O: L293D	±600 mA
Continuous total dissipation at (or below) 25°C free-air temperature (see Notes 2 and 3)	2075 mW
Continuous total dissipation at 80°C case temperature (see Note 3)	5000 mW
Maximum junction temperature, T_J	150°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds	260°C
Storage temperature range, T_{stg}	–65°C to 150°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 the network ground terminal.
2. For operation above 25°C free-air temperature, derate linearly at the rate of 16.6 mW/°C.
3. For operation above 25°C case temperature, derate linearly at the rate of 71.4 mW/°C. Due to variations in individual device electrical characteristics and thermal resistance, the built-in thermal overload protection may be activated at power levels slightly above or below the rated dissipation.

recommended operating conditions

			MIN	MAX	UNIT
	Supply voltage	V_{CC1}	4.5	7	V
		V_{CC2}	V_{CC1}	36	
V_{IH}	High-level input voltage	$V_{CC1} \le 7$ V	2.3	V_{CC1}	V
		$V_{CC1} \ge 7$ V	2.3	7	V
V_{IL}	Low-level output voltage		–0.3†	1.5	V
T_A	Operating free-air temperature		0	70	°C

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

electrical characteristics, V_{CC1} = 5 V, V_{CC2} = 24 V, T_A = 25°C

PARAMETER			TEST CONDITIONS		MIN	TYP	MAX	UNIT
V_{OH}	High-level output voltage		L293: I_{OH} = –1 A L293D: I_{OH} = –0.6 A		V_{CC2}–1.8	V_{CC2}–1.4		V
V_{OL}	Low-level output voltage		L293: I_{OL} = 1 A L293D: I_{OL} = 0.6 A			1.2	1.8	V
V_{OKH}	High-level output clamp voltage		L293D: I_{OK} = –0.6 A			V_{CC2} + 1.3		V
V_{OKL}	Low-level output clamp voltage		L293D: I_{OK} = 0.6 A			1.3		V
I_{IH}	High-level input current	A	V_I = 7 V			0.2	100	µA
		EN				0.2	10	
I_{IL}	Low-level input current	A	V_I = 0			–3	–10	µA
		EN				–2	–100	
I_{CC1}	Logic supply current		I_O = 0	All outputs at high level		13	22	mA
				All outputs at low level		35	60	
				All outputs at high impedance		8	24	
I_{CC2}	Output supply current		I_O = 0	All outputs at high level		14	24	mA
				All outputs at low level		2	6	
				All outputs at high impedance		2	4	

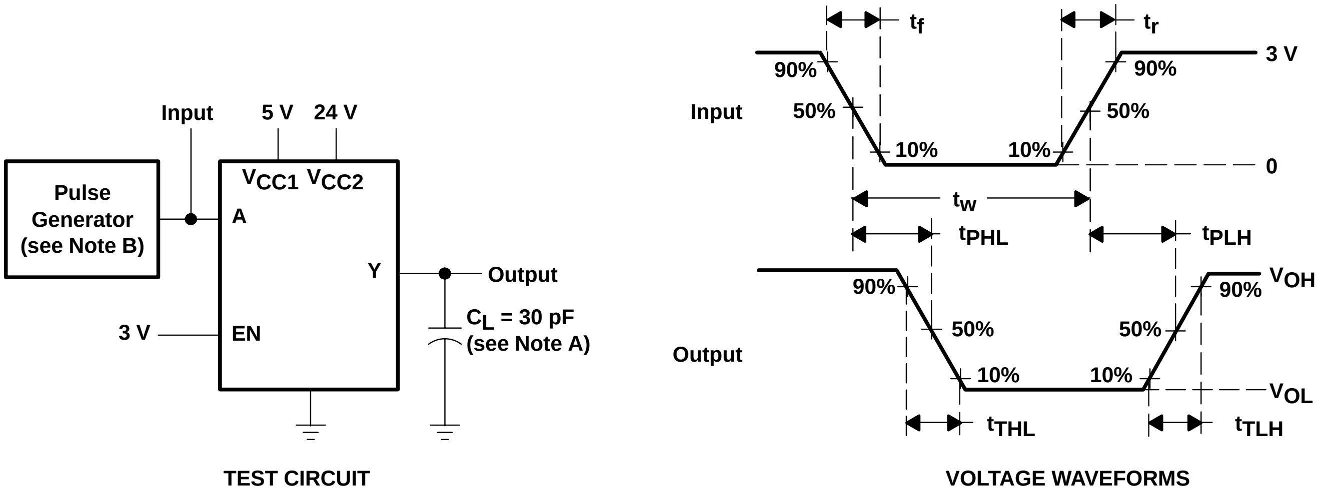
switching characteristics, V_{CC1} = 5 V, V_{CC2} = 24 V, T_A = 25°C

PARAMETER		TEST CONDITIONS	L293NE, L293DNE			UNIT
			MIN	TYP	MAX	
t_{PLH}	Propagation delay time, low-to-high-level output from A input	C_L = 30 pF, See Figure 1		800		ns
t_{PHL}	Propagation delay time, high-to-low-level output from A input			400		ns
t_{TLH}	Transition time, low-to-high-level output			300		ns
t_{THL}	Transition time, high-to-low-level output			300		ns

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

PARAMETER		TEST CONDITIONS	L293DWP, L293N L293DDWP, L293DN			UNIT
			MIN	TYP	MAX	
t_{PLH}	Propagation delay time, low-to-high-level output from A input	C_L = 30 pF, See Figure 1		750		ns
t_{PHL}	Propagation delay time, high-to-low-level output from A input			200		ns
t_{TLH}	Transition time, low-to-high-level output			100		ns
t_{THL}	Transition time, high-to-low-level output			350		ns

PARAMETER MEASUREMENT INFORMATION



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

Figure 1. Test Circuit and Voltage Waveforms

APPLICATION INFORMATION

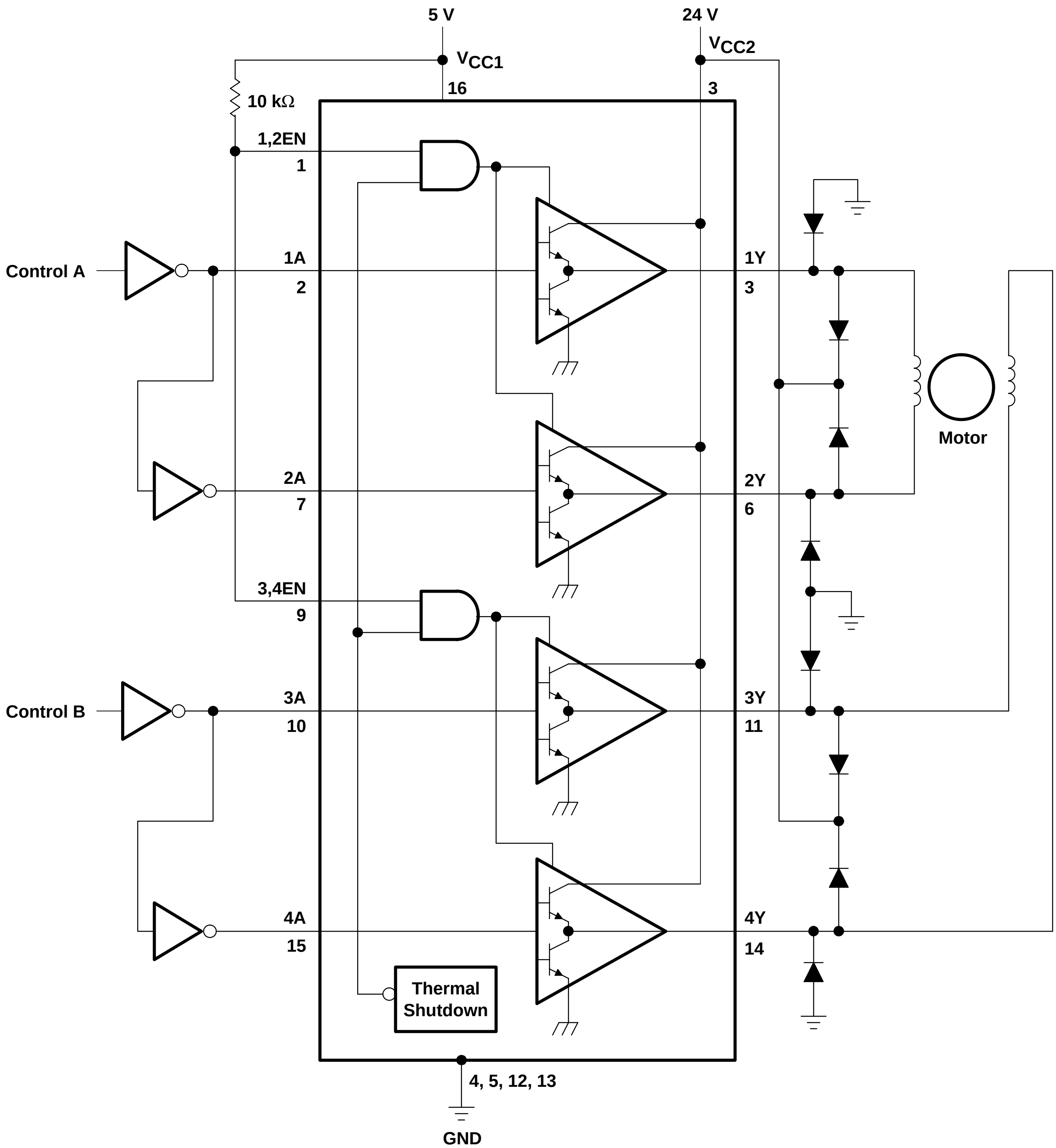


Figure 2. Two-Phase Motor Driver (L293)

APPLICATION INFORMATION

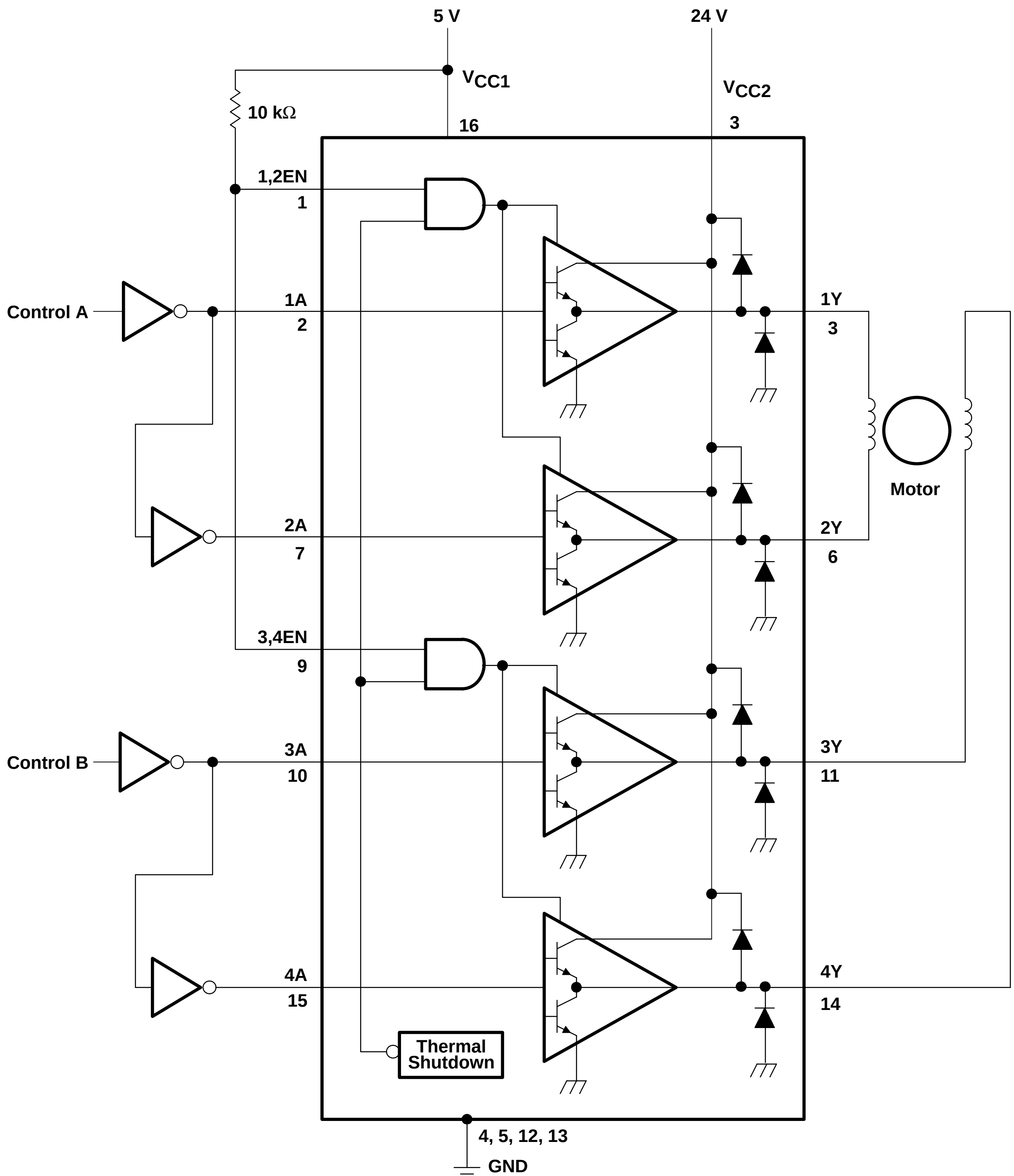


Figure 3. Two-Phase Motor Driver (L293D)

APPLICATION INFORMATION

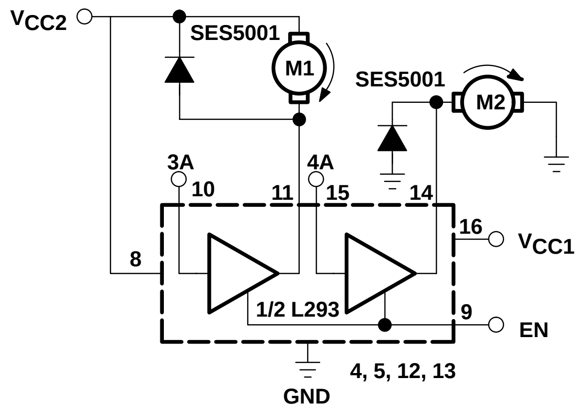


Figure 4. DC Motor Controls (connections to ground and to supply voltage)

EN	3A	M1	4A	M2
H	H	Fast motor stop	H	Run
H	L	Run	L	Fast motor stop
L	X	Free-running motor stop	X	Free-running motor stop

L = low, H = high, X = don't care

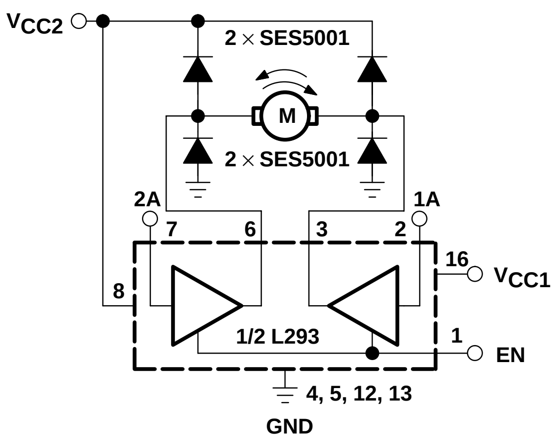


Figure 5. Bidirectional DC Motor Control

EN	1A	2A	FUNCTION
H	L	H	Turn right
H	H	L	Turn left
H	L	L	Fast motor stop
H	H	H	Fast motor stop
L	X	X	Fast motor stop

L = low, H = high, X = don't care

APPLICATION INFORMATION

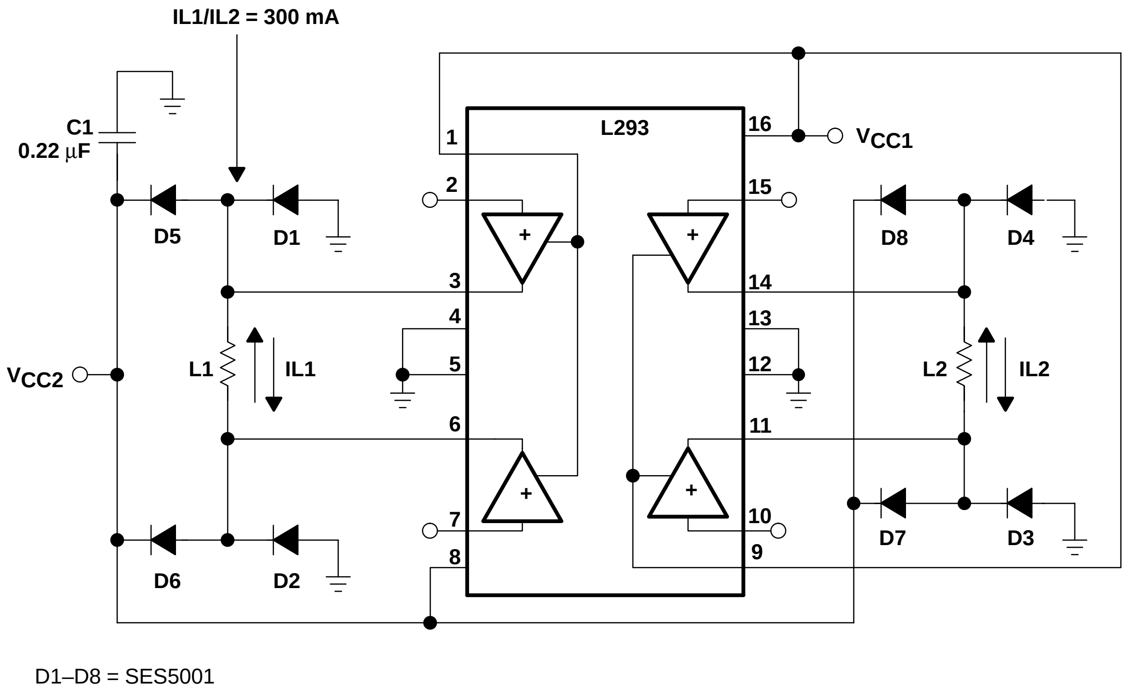


Figure 6. Bipolar Stepping-Motor Control

mounting instructions

The Rthj-amp of the L293 can be reduced by soldering the GND pins to a suitable copper area of the printed circuit board or to an external heatsink.

Figure 9 shows the maximum package power P_{TOT} and the θ_{JA} as a function of the side ℓ of two equal square copper areas having a thickness of 35 µm (see Figure 7). In addition, an external heat sink can be used (see Figure 8).

During soldering, the pin temperature must not exceed 260°C, and the soldering time must not be longer than 12 seconds.

The external heatsink or printed circuit copper area must be connected to electrical ground.

APPLICATION INFORMATION

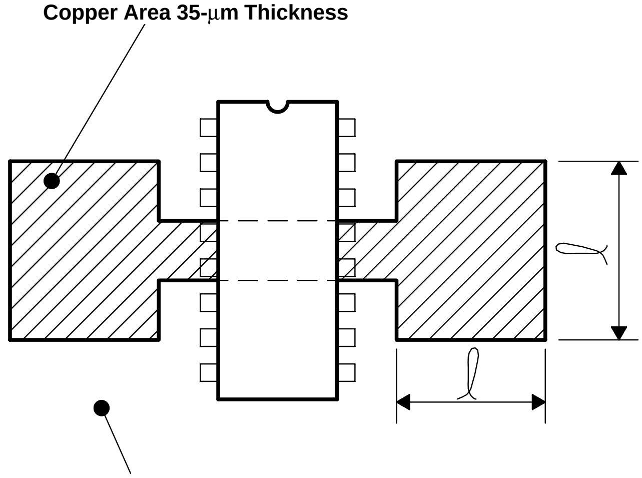


Figure 7. Example of Printed Circuit Board Copper Area (used as heat sink)

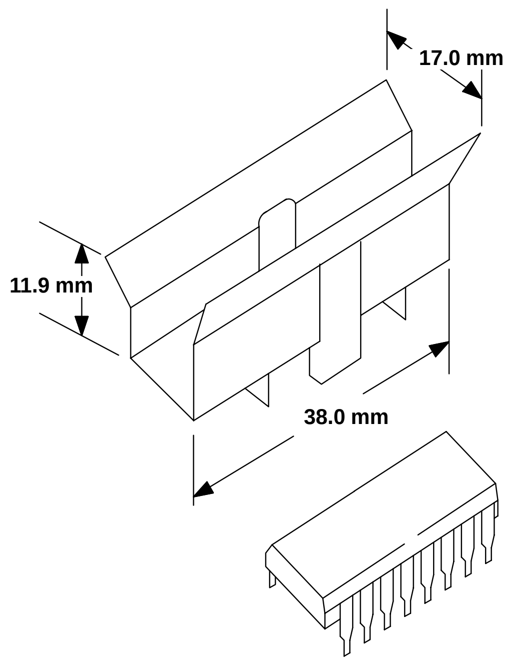


Figure 8. External Heat Sink Mounting Example (θ_{JA} = 25°C/W)

APPLICATION INFORMATION

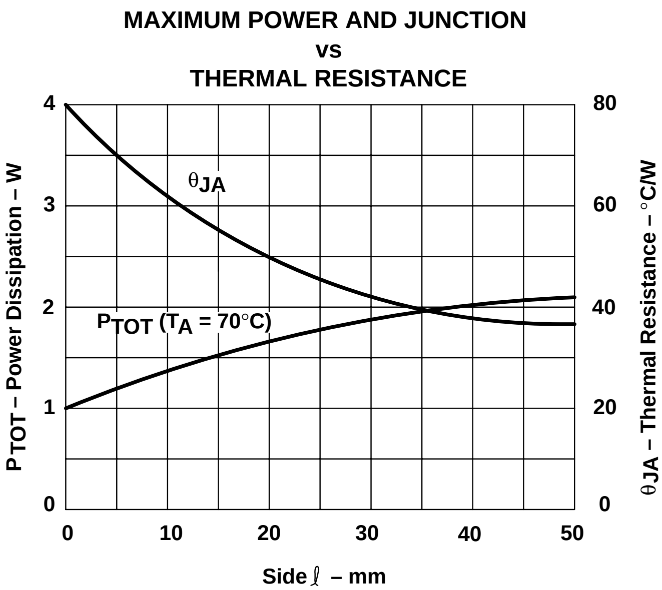


Figure 9

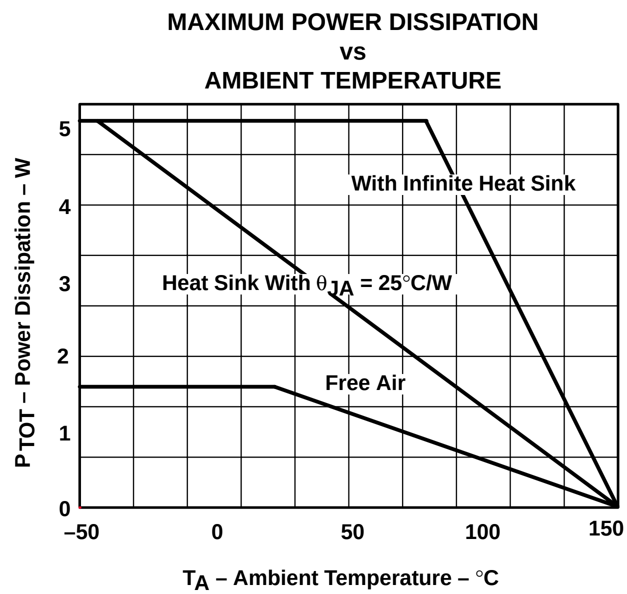


Figure 10

Mailing Address:

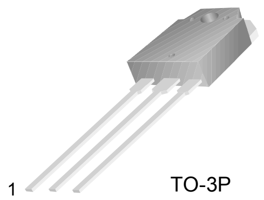
Texas Instruments
Post Office Box 655303
Dallas, Texas 75265

FAIRCHILD SEMICONDUCTOR ™

TIP140/141/142

Monolithic Construction With Built In Base-Emitter Shunt Resistors

- High DC Current Gain : h_{FE} = 1000 @ V_{CE} = 4V, I_C = 5A (Min.)
- Industrial Use
- Complement to TIP145/146/147

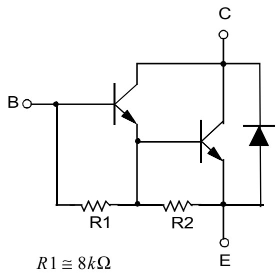


TO-3P

1.Base 2.Collector 3.Emitter

NPN Epitaxial Silicon Darlington Transistor

Equivalent Circuit



$R1 \cong 8k\Omega$
$R2 \cong 0.12k\Omega$

Absolute Maximum Ratings T_C=25°C unless otherwise noted

Symbol	Parameter	Value	Units
V_{CBO}	Collector-Base Voltage : TIP140	60	V
	: TIP141	80	V
	: TIP142	100	V
V_{CEO}	Collector-Emitter Voltage : TIP140	60	V
	: TIP141	80	V
	: TIP142	100	V
V_{EBO}	Emitter-Base Voltage	5	V
I_C	Collector Current (DC)	10	A
I_{CP}	Collector Current (Pulse)	15	A
I_B	Base Current (DC)	0.5	A
P_C	Collector Dissipation (T_C=25°C)	125	W
T_J	Junction Temperature	150	°C
T_{STG}	Storage Temperature	- 65 ~ 150	°C

Electrical Characteristics T_C=25°C unless otherwise noted

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Units
V_{CEO}(sus)	Collector-Emitter Sustaining Voltage					
	: TIP140	I_C = 30mA, I_B = 0	60			V
	: TIP141		80			V
	: TIP142		100			V
I_{CEO}	Collector Cut-off Current					
	: TIP140	V_{CE} = 30V, I_B = 0			2	mA
	: TIP141	V_{CE} = 40V, I_B = 0			2	mA
	: TIP142	V_{CE} = 50V, I_B = 0			2	mA
I_{CBO}	Collector Cut-off Current					
	: TIP140	V_{CB} = 60V, I_E = 0			1	mA
	: TIP141	V_{CB} = 80V, I_E = 0			1	mA
	: TIP142	V_{CB} = 100V, I_E = 0			1	mA
I_{EBO}	Emitter Cut-off Current	V_{BE} = 5V, I_C = 0			2	mA
h_{FE}	DC Current Gain	V_{CE} = 4V, I_C = 5A	1000			
		V_{CE} = 4V, I_C = 10A	500			
V_{CE}(sat)	Collector-Emitter Saturation Voltage	I_C = 5A, I_B = 10mA			2	V
		I_C = 10A, I_B = 40mA			3	V
V_{BE}(sat)	Base-Emitter Saturation Voltage	I_C = 10A, I_B = 40mA			3.5	V
V_{BE}(on)	Base-Emitter ON Voltage	V_{CE} = 4V, I_C = 10A			3	V
t_D	Delay Time	V_{CC} = 30V, I_C = 5A		0.15		µs
t_R	Rise Time	I_{B1} = 20mA, I_{B2} = -20mA		0.55		µs
t_{STG}	Storage Time	R_L = 6Ω		2.5		µs
t_F	Fall Time			2.5		µs

Typical Characteristics

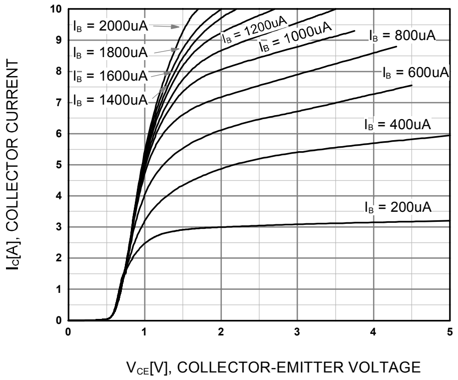


Figure 1. Static Characteristic

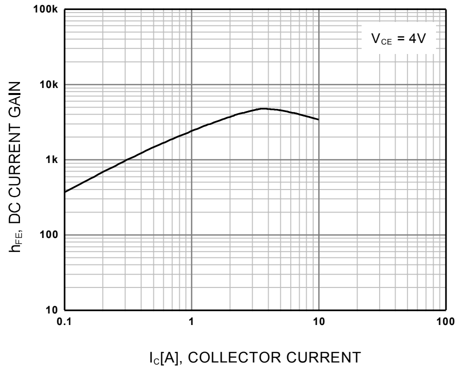


Figure 2. DC current Gain

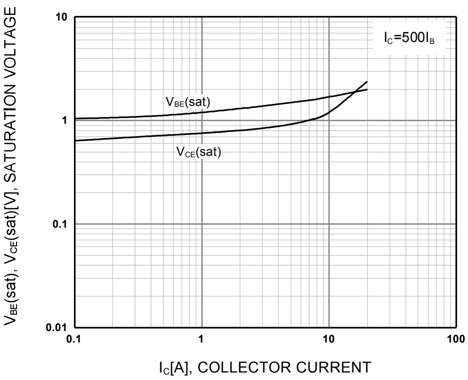


Figure 3. Base-Emitter Saturation Voltage Collector-Emitter Saturation Voltage

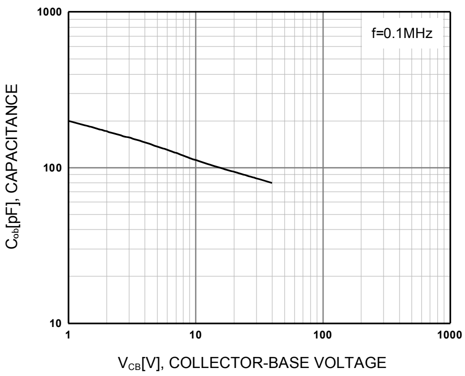


Figure 4. Collector Output Capacitance

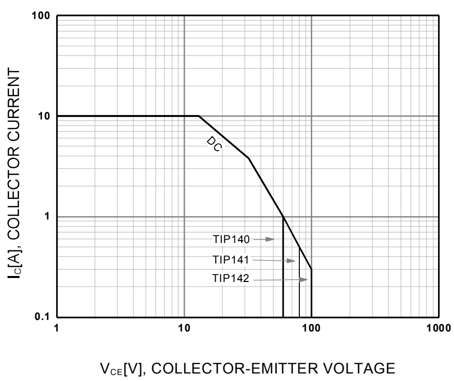


Figure 5. Safe Operating Area

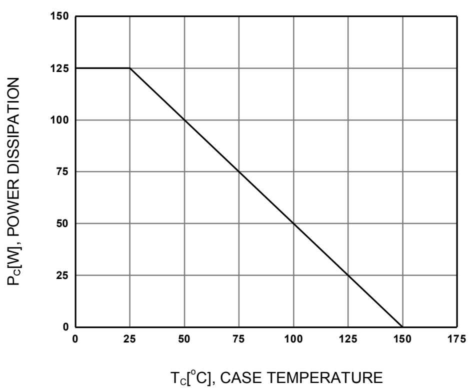
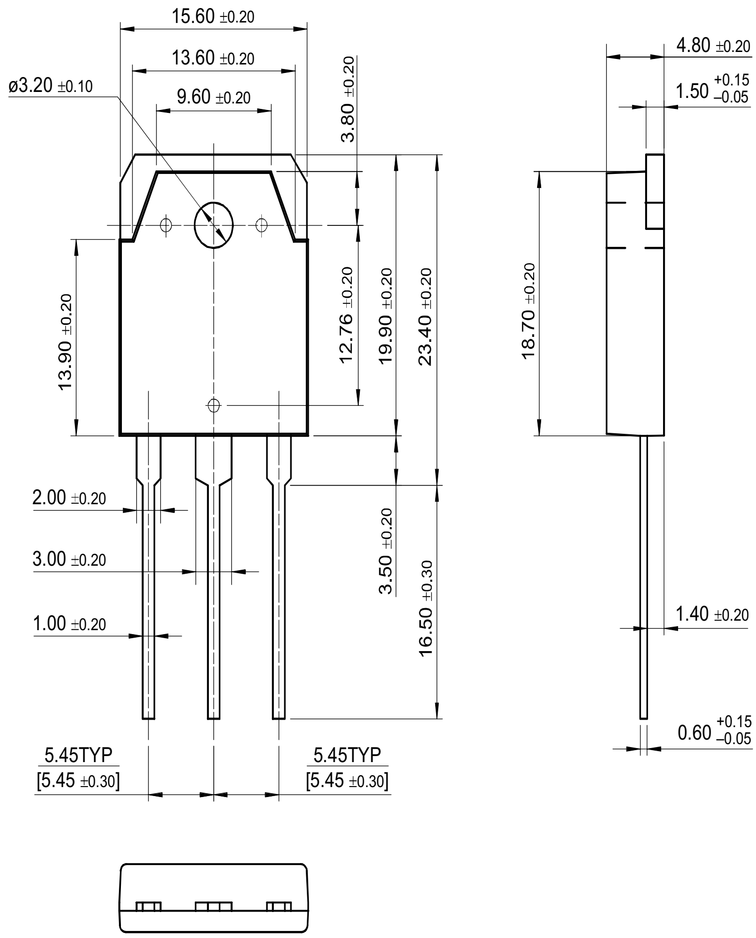


Figure 6. Power Derating

Package Demensions

TO-3P



Dimensions in Millimeters

TRADEMARKS

The following are registered and unregistered trademarks Fairchild Semiconductor owns or is authorized to use and is not intended to be an exhaustive list of all such trademarks.

ACEx™	HiSeC™	SuperSOT™-8
Bottomless™	ISOPLANAR™	SyncFET™
CoolFET™	MICROWIRE™	TinyLogic™
CROSSVOLT™	POP™	UHC™
E^2CMOS™	PowerTrench®	VCX™
FACT™	QFET™	
FACT Quiet Series™	QS™	
FAST®	Quiet Series™	
FASTr™	SuperSOT™-3	
GTO™	SuperSOT™-6	

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 FAIRCHILD SEMICONDUCTOR INTERNATIONAL.

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, or (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 significant injury to the user.

2. A critical component is 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.

PRODUCT STATUS DEFINITIONS

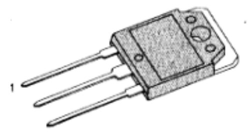
Definition of Terms

Datasheet Identification	Product Status	Definition
Advance Information	Formative or In Design	This datasheet contains the design specifications for product development. Specifications may change in any manner without notice.
Preliminary	First Production	This datasheet contains preliminary data, and supplementary data will be published at a later date. Fairchild Semiconductor reserves the right to make changes at any time without notice in order to improve design.
No Identification Needed	Full Production	This datasheet contains final specifications. Fairchild Semiconductor reserves the right to make changes at any time without notice in order to improve design.
Obsolete	Not In Production	This datasheet contains specifications on a product that has been discontinued by Fairchild semiconductor. The datasheet is printed for reference information only.

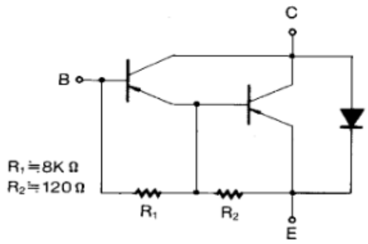
PNP EPITAXIAL
TIP145/146/147 SILICON DARLINGTON TRANSISTOR

HIGH DC CURRENT GAIN
MIN h_{FE} = 1000 @ V_{CE} = -4V, IC = -5A
MONOLITHIC CONSTRUCTION WITH BUILT IN BASE-EMITTER SHUNT RESISTORS
INDUSTRIAL USE
- Complement to TIP140/141/142

TO-3P



1. Base 2. Collector 3. Emitter



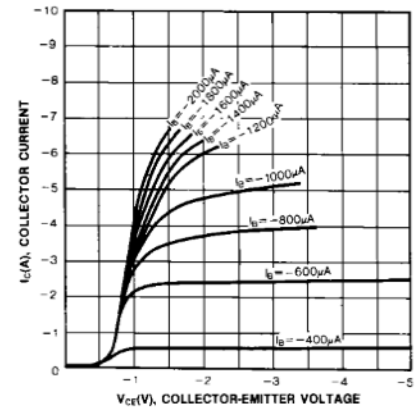
ABSOLUTE MAXIMUM RATINGS

Characteristic	Symbol	Rating	Unit
Collector-Base Voltage : TIP145	V_{CBO}	- 60	V
: TIP146		- 80	V
: TIP147		- 100	V
Collector Emitter Voltage			
: TIP145	V_{CEO}	- 60	V
: TIP146		- 80	V
: TIP147		- 100	V
Emitter-Base Voltage	V_{EBO}	- 5	V
Collector Current (DC)	I_C	- 10	A
Collector Current (Pulse)	I_C	- 15	A
Base Current (DC)	I_B	- 0.5	A
Collector Dissipation (T_C=25°C)	P_C	125	W
Junction Temperature	T_J	150	°C
Storage Temperature	T_{STG}	- 65 ~ 150	°C

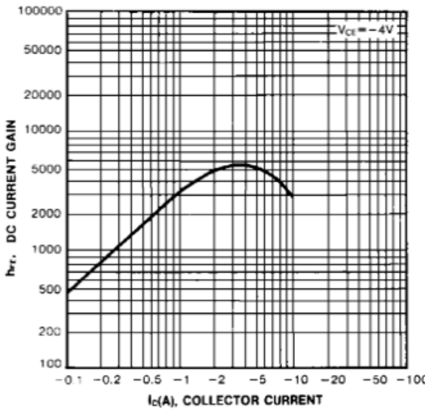
ELECTRICAL CHARACTERISTICS (T_C =25°C)

Characteristic	Symbol	Test Conditions	Min	Typ	Max	Unit
Collector Emitter Sustaining Voltage	V_{CEO}(sus)					
: TIP145		I_C = - 30mA, I_B = 0	- 60			V
: TIP146			- 80			V
: TIP147			- 100			V
Collector Cutoff Current : TIP145	I_{CEO}	V_{CE} = - 30V, I_B = 0			- 2	mA
: TIP146		V_{CE} = - 40V, I_B = 0			- 2	mA
: TIP147		V_{CE} = - 50V, I_B = 0			- 2	mA
Collector Cutoff Current : TIP145	I_{CBO}	V_{CB} = - 60V, I_E = 0			- 1	mA
: TIP146		V_{CB} = - 80V, I_E = 0			- 1	mA
: TIP147		V_{CB} = - 100V, I_E = 0			- 1	mA
Emitter Cutoff Current	I_{EBO}	V_{BE} = - 5V, I_C = 0			- 2	mA
DC Current Gain	h_{FE}	V_{CE} = - 4V,I_C = - 5A	1000			
		V_{CE} = - 4V, I_C = - 10A	500			
Collector Emitter Saturation Voltage	V_{CE}(sat)	I_C = - 5A, I_B = - 10mA			- 2	V
		I_C = - 10A, I_B = - 40mA			- 3	V
Base Emitter Saturation Voltage	V_{BE}(sat)	I_C = - 10A, I_B = - 40mA			- 3.5	V
Base Emitter On Voltage	V_{BE}(on)	V_{CE} = - 4V, I_C = - 10A			- 3	V
Delay Time	t_D	V_{CC} = - 30V, I_C = - 5A		0.15		µs
Rise Time	t_R	I_B = - 20mA, I_{B1} = -I_{B2}		0.55		µs
Storage Time	t_{STG}			2.5		µs
Fall Time	t_F			2.5		µs

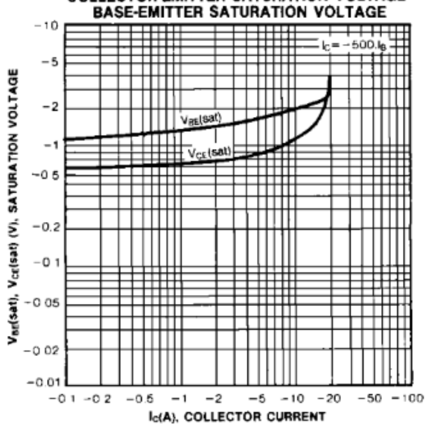
STATIC CHARACTERISTIC



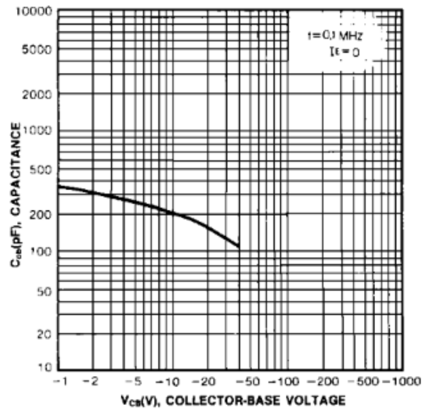
DC CURRENT GAIN



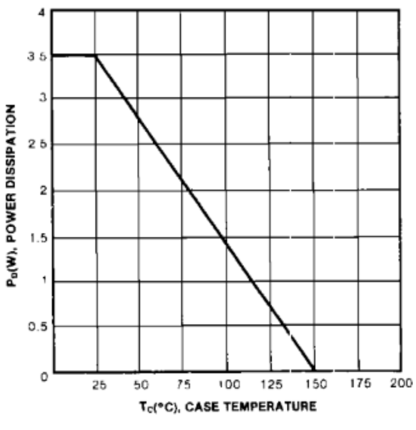
COLLECTOR-EMITTER SATURATION VOLTAGE
BASE-EMITTER SATURATION VOLTAGE



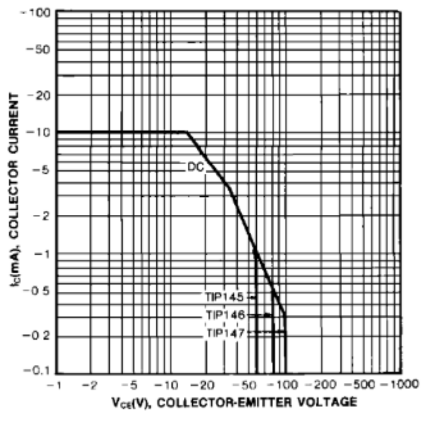
COLLECTOR OUTPUT CAPACITANCE



POWER DERATING



SAFE OPERATING AREA



TRADEMARKS

The following are registered and unregistered trademarks Fairchild Semiconductor owns or is authorized to use and is not intended to be an exhaustive list of all such trademarks.

ACEx™
CoolFET™
CROSSVOLT™
E^2CMOS™
FACT™
FACT Quiet Series™
FAST®
FASTr™
GTO™
HiSeC™
ISOPLANAR™
MICROWIRE™
POP™
PowerTrench™
QS™
Quiet Series™
SuperSOT™-3
SuperSOT™-6
SuperSOT™-8
TinyLogic™

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 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, or (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 significant injury to the user.

2. A critical component is 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.

PRODUCT STATUS DEFINITIONS

Definition of Terms

Datasheet Identification	Product Status	Definition
Advance Information	Formative or In Design	This datasheet contains the design specifications for product development. Specifications may change in any manner without notice.
Preliminary	First Production	This datasheet contains preliminary data, and supplementary data will be published at a later date. Fairchild Semiconductor reserves the right to make changes at any time without notice in order to improve design.
No Identification Needed	Full Production	This datasheet contains final specifications. Fairchild Semiconductor reserves the right to make changes at any time without notice in order to improve design.
Obsolete	Not In Production	This datasheet contains specifications on a product that has been discontinued by Fairchild semiconductor. The datasheet is printed for reference information only.

POWER SUPPLY SPECIFICATIONS

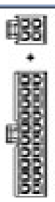
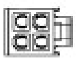
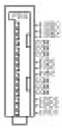
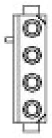
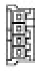
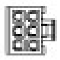
• Wattage:	180 Watt
• Fan:	80 mm
• +3.3V:	17 A
• +5V:	18 A
• +12V 1:	10 A

- -12V: 0.3A

- -5V: 0.2 A

- +5VSB: 1.5 A

Connectors

					
20+4 pin	**P4 MB**	**SATA**	**Peripheral**	**FDD**	**PCI-E**
x 1	x 1	x 0	x 4	x 2	x 0

(1)

By conservation of energy, energy supplied = energy lost = heat + work

> $A \cdot V = A^2 \cdot R + \tau \cdot \omega$

$$A\,V = A^2\,R + \tau\,\omega \quad (2)$$

By measurement, for the original D.C motor.

> $\omega := \frac{10}{19} \cdot 2 \cdot \pi;$

$$\omega := \frac{20}{19}\,\pi \quad (3)$$

> $R := 18;\ \ A := 0.15;\ V := 11.25;$

$$R := 18$$
$$A := 0.15$$
$$V := 11.25 \quad (4)$$

> $\tau := solve\left(A \cdot V = A^2 \cdot R + \tau \cdot \omega,\ \tau\right);$

$$\tau := 0.3878208076 \quad (5)$$

> $StallCurrent := 0.32;$

$$StallCurrent := 0.32 \quad (6)$$

> $\frac{\tau}{A};$

$$2.585472051 \quad (7)$$

> $StallTorque := \frac{\tau}{A} \cdot StallCurrent$

$$StallTorque := 0.8273510563 \quad (8)$$

> $Stallcurrent2 := \frac{StallTorque}{1.16};$

$$Stallcurrent2 := 0.7132336692 \quad (9)$$

To choose an approperiate resistor for the new motor, to get the same effects

> $R := 6 : V := 11.25 :$

> $Rresistor := solve\left(\frac{V}{R + Rresistor} = Stallcurrent2,\ Rresistor\right);$

$$Rresistor := 9.773231811 \quad (10)$$

>

So the resistor choosen should be bigger than 10, we decided to choose 12. In this case, the stall torque would be (in Nm)

> $\frac{V}{R + Rresistor} \cdot 1.16;$

$$0.8273510564 \quad (11)$$

>

PIC APPENDIX

1, RAM Layout

2, Stepper Motor Driving Calculations

3, PIC16F877 (A/D Section)

4, MM74C922N Keypad Decoder

5, PIC proto64 Board

6, KA7805 Power Regulator

7, Keypad

8, TS1620-1 LCD

9, DS 1307 Realtime Clock Chip

10, Code

PIC APPENDIX

1, RAM Layout

2, Stepper Motor Driving Calculations

3, PIC16F877 (A/D Section)

4, MM74C922N Keypad Decoder

5, PIC proto64 Board

6, KA7805 Power Regulator

7, Keypad

8, TS1620-1 LCD

9, DS 1307 Realtime Clock Chip

RAM Layout

	BANK0		BANK1		BANK2		BANK3
00	INDF	80	INDF	100	INDF	180	INDF
01	TMR0	81	OPTION_REG	101	TMR0	181	OPTION_REG
02	PCL	82	PCL	102	PCL	182	PCL
03	STATUS	83	STATUS	103	STATUS	183	STATUS
04	FSR	84	FSR	104	FSR	184	FSR
05	PORTA	85	TRISA	105		185	
06	PORTB	86	TRISB	106	PORTB	186	TRISB
07	PORTC	87	TRISC	107		187	
08	PORTD	88	TRISD	108		188	
09	PORTE	89	TRISE	109		189	
0A	PCLATH	8A	PCLATH	10A	PCLATH	18A	PCLATH
0B	INCTON	8B	INTCON	10B	INTCON	18B	INTCON
0C	PIR1	8C	PIE1	10C	EEDATA	18C	EECON1
0D	PIR2	8D	PIE2	10D	EEADR	18D	EECON2
0E	TMR1L	8E	PCON	10E	EEDATH	18E	RV
0F	TMR1H	8F		10F	EEADRH	18F	RV
10	TICON	90		110		190	
11	TMR2	91	SSPCON2	111		191	
12	T2CON	92	PR2	112		192	
13	SSPBUF	93	SSPADD	113		193	
14	SSPCON	94	SSPSTAT	114		194	
15	CCPR1L	95		115		195	
16	CCPR1H	96		116		196	
17	CCP1CON	97		117		197	
18	RCSTA	98	TXSTA	118		198	
19	TXREG	99	SPBRG	119		199	
1A	RCREG	9A		11A		19A	
1B	CCPR2L	9B		11B		19B	
1C	CCPR2H	9C		11C		19C	
1D	CCP2CON	9D		11D		19D	
1E	ADRESH	9E	ADRESL	11E		19E	
1F	ADCON2	9F	ADCON1	11F	Log Entry 0	19F	Log Entry 6
20	phase	A0	delaytemp	120		1A0	
21	report_num	A1	delaycount	121		1A1	
22	line_num	A2	delaytemp2	122		1A2	
23	LCDline(0)	A3	delaycount2	123		1A3	
24	LCDline(1)	A4	delaytemp3	124		1A4	
25	LCDline(2)	A5	delaycount3	125		1A5	
26	LCDline(3)	A6		126	Log Entry 1	1A6	Log Entry 7

27	LCDline(4)	A7		127		1A7	
28	LCDline(5)	A8		128		1A8	
29	LCDline(6)	A9		129		1A9	
2A	LCDline(7)	AA		12A		1AA	
2B	LCDline(8)	AB		12B		1AB	
2C	LCDline(9)	AC		12C		1AC	
2D	LCDline(A)	AD		12D		1AD	
2E	LCDline(B)	AE		12E		1AE	
2F	LCDline(C)	AF		12F		1AF	
30	LCDline(D)	B0	light_bg(0)	130		1B0	
31	LCDline(E)	B1	light_bg(1)	131		1B1	
32	LCDline(F)	B2	light_bg(2)	132		1B2	
33	LCDline(Null)	B3	light_bg(3)	133		1B3	
34	temp	B4	light_bg(4)	134		1B4	
35	temp2	B5	light_pos(0)	135		1B5	
36	temp3	B6	light_pos(1)	136		1B6	
37	temp4	B7	light_pos(2)	137		1B7	
38	temp5	B8	light_pos(3)	138		1B8	
39	temp6	B9	light_pos(4)	139		1B9	
3A	temp7	BA	light_cl(0)	13A		1BA	
3B	arg	BB	light_cl(1)	13B		1BB	
3C	arg2	BC	light_cl(2)	13C		1BC	
3D	literal_addr	BD	light_cl(3)	13D		1BD	
3E	rowleft	BE	light_cl(4)	13E		1BE	
3F	result_addr	BF	light_off(0)	13F	Log Entry 2	1BF	Log Entry 8
40	rt_year	C0	light_off(1)	140		1C0	
41	rt_month	C1	light_off(2)	141		1C1	
42	rt_day	C2	light_off(3)	142		1C2	
43	rt_hour	C3	light_off(4)	143		1C3	
44	rt_min	C4		144		1C4	
45	rt_sec	C5		145		1C5	
46	st_year	C6		146		1C6	
47	st_month	C7		147		1C7	
48	st_day	C8		148		1C8	
49	st_hour	C9		149		1C9	
4A	st_min	CA		14A		1CA	
4B	st_sec	CB		14B		1CB	
4C	end_hour	CC		14C		1CC	
4D	end_min	CD		14D		1CD	
4E	end_sec	CE		14E		1CE	
4F	runtime	CF		14F	Log Entry 3	1CF	Log Entry 9

50	cl_total	D0		150		1D0	
51	cl_pass	D1		151		1D1	
52	cl_fail	D2		152		1D2	
53	layout(0,0)	D3		153		1D3	
54	layout(0,1)	D4		154		1D4	
55	layout(0,2)	D5		155		1D5	
56	layout(0,3)	D6		156		1D6	
57	layout(0,4)	D7		157		1D7	
58	layout(1,0)	D8		158		1D8	
59	layout(1,1)	D9		159		1D9	
5A	layout(1,2)	DA		15A		1DA	
5B	layout(1,3)	DB		15B		1DB	
5C	layout(1,4)	DC		15C		1DC	
5D	layout(2,0)	DD		15D		1DD	
5E	layout(2,1)	DE		15E		1DE	
5F	layout(2,2)	DF		15F	Log Entry 4	1DF	Log Entry 10
60	layout(2,3)	E0		160		1E0	
61	layout(2,4)	E1		161		1E1	
62	layout(3,0)	E2		162		1E2	
63	layout(3,1)	E3		163		1E3	
64	layout(3,2)	E4		164		1E4	
65	layout(3,3)	E5		165		1E5	
66	layout(3,4)	E6		166		1E6	
67	smotor_dir	E7		167		1E7	
68	log_total	E8		168		1E8	
69	log_next	E9		169		1E9	
6A	arith_temp	EA		16A		1EA	
6B	arith_temp2	EB		16B		1EB	
6C	newsec	EC		16C		1EC	
6D		ED		16D		1ED	
6E		EE		16E		1EE	
6F		EF		16F	Log Entry 5	1EF	Log Entry 11
70	lcd_tmp	F0	lcd_tmp	170	lcd_tmp	1F0	lcd_tmp
71	w_temp	F1	w_temp	171	w_temp	1F1	w_temp
72	status_temp	F2	status_temp	172	status_temp	1F2	status_temp
73	FSR_temp	F3	FSR_temp	173	FSR_temp	1F3	FSR_temp
74	rt_counter	F4	rt_counter	174	rt_counter	1F4	rt_counter
75		F5		175		1F5	
76		F6		176		1F6	
77		F7		177		1F7	
78		F8		178		1F8	

79		F9		179		1F9	
7A		FA		17A		1FA	
7B		FB		17B		1FB	
7C		FC		17C		1FC	
7D		FD		17D		1FD	
7E		FE		17E		1FE	
7F		FF		17F		1FF	

Measurement Facts:
Stepper Motor: 48 steps/rotation(r)
T30: 30 steps (teeth)
T50: 50 steps (teeth)
T10: 10 steps (teeth)
Rack: $\frac{83.5\text{mm}}{25\text{steps}} = 3.34\text{mm/step}$
row on egg tray: max 14.25cm 3rows; min 13.60cm 3rows

Gear ratio:

$$48 \text{ step S. Motor} = 1\text{r S. Motor} = 1\text{r T30} = 30\text{steps T30} = 30\text{steps T50} = \frac{30}{50}\text{r T50} = 0.6\text{r T10} = 6\text{steps T10} = 6\text{steps rack}$$

$$\rightarrow 8 \text{ S. Motor: } 1 \text{ rack}$$

Row distance:

$$\frac{14.25\text{cm} + 13.6\text{cm}}{2} \times \frac{1}{3} = 46.4\text{mm/row}$$

S.Motor steps for each row:

$$\frac{46.4\text{mm/row}}{3.34\text{mm/step rack}} \times \frac{8 \text{ step S. Motor}}{1 \text{ step rack}} = 111.1 \text{ step/row} = 112 \text{ step/row}$$

The round up is due to the fact than PIC program can only run stepper motor in multiple of 4steps.

Actual distance testing arm travels:

$$112 \text{ step S. Motor/row} = 46.8\text{mm/row} = 14.0\text{cm 3rows}$$

Error between the travelling distance and max/min row distance are -0.25cm and +0.4cm, respectively. The error is the max accumulated error caused by stepper motor and inconsistency of egg-tray row-separation (testing arm moves exactly 3 row separation in each run), it is in the ±1cm design tolerance.

11.0 ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) Converter module has five inputs for the 28-pin devices and eight for the other devices.

The analog input charges a sample and hold capacitor. The output of the sample and hold capacitor is the input into the converter. The converter then generates a digital result of this analog level via successive approximation. The A/D conversion of the analog input signal results in a corresponding 10-bit digital number. The A/D module has high and low voltage reference input that is software selectable to some combination of V_{DD}, V_{SS}, RA2, or RA3.

The A/D converter has a unique feature of being able to operate while the device is in SLEEP mode. To operate in SLEEP, the A/D clock must be derived from the A/D's internal RC oscillator.

The A/D module has four registers. These registers are:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register0 (ADCON0)
- A/D Control Register1 (ADCON1)

The ADCON0 register, shown in Register 11-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 11-2, configures the functions of the port pins. The port pins can be configured as analog inputs (RA3 can also be the voltage reference), or as digital I/O.

Additional information on using the A/D module can be found in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

REGISTER 11-1: ADCON0 REGISTER (ADDRESS: 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

bit 7-6 **ADCS1:ADCS0:** A/D Conversion Clock Select bits
00 = $F_{OSC}/2$
01 = $F_{OSC}/8$
10 = $F_{OSC}/32$
11 = F_{RC} (clock derived from the internal A/D module RC oscillator)

bit 5-3 **CHS2:CHS0**: Analog Channel Select bits
000 = channel 0, (RA0/AN0)
001 = channel 1, (RA1/AN1)
010 = channel 2, (RA2/AN2)
011 = channel 3, (RA3/AN3)
100 = channel 4, (RA5/AN4)
101 = channel 5, (RE0/AN5)(1)
110 = channel 6, (RE1/AN6)(1)
111 = channel 7, (RE2/AN7)(1)

bit 2 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
If ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion)
0 = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)

bit 1 **Unimplemented**: Read as '0'

bit 0 **ADON**: A/D On bit
1 = A/D converter module is operating
0 = A/D converter module is shut-off and consumes no operating current

Note 1: These channels are not available on PIC16F873/876 devices.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 11-2: ADCON1 REGISTER (ADDRESS 9Fh)

U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified. 6 Most Significant bits of ADRESH are read as ‘0’.
0 = Left justified. 6 Least Significant bits of ADRESL are read as ‘0’.

bit 6-4 **Unimplemented:** Read as '0'

bit 3-0 **PCFG3:PCFG0**: A/D Port Configuration Control bits:

PCFG3: PCFG0	AN7(1) RE2	AN6(1) RE1	AN5(1) RE0	AN4 RA5	AN3 RA3	AN2 RA2	AN1 RA1	AN0 RA0	VREF+	VREF-	CHAN/ Refs(2)
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	RA3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	RA3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	RA3	VSS	2/1
011x	D	D	D	D	D	D	D	D	VDD	VSS	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	RA3	RA2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	RA3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	RA3	RA2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	RA3	RA2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	RA3	RA2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	RA3	RA2	1/2

A = Analog input D = Digital I/O

Note 1: These channels are not available on PIC16F873/876 devices.
2: This column indicates the number of analog channels available as A/D inputs and the number of analog channels used as voltage reference inputs.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
- n = Value at POR	’1’ = Bit is set	’0’ = Bit is cleared	x = Bit is unknown

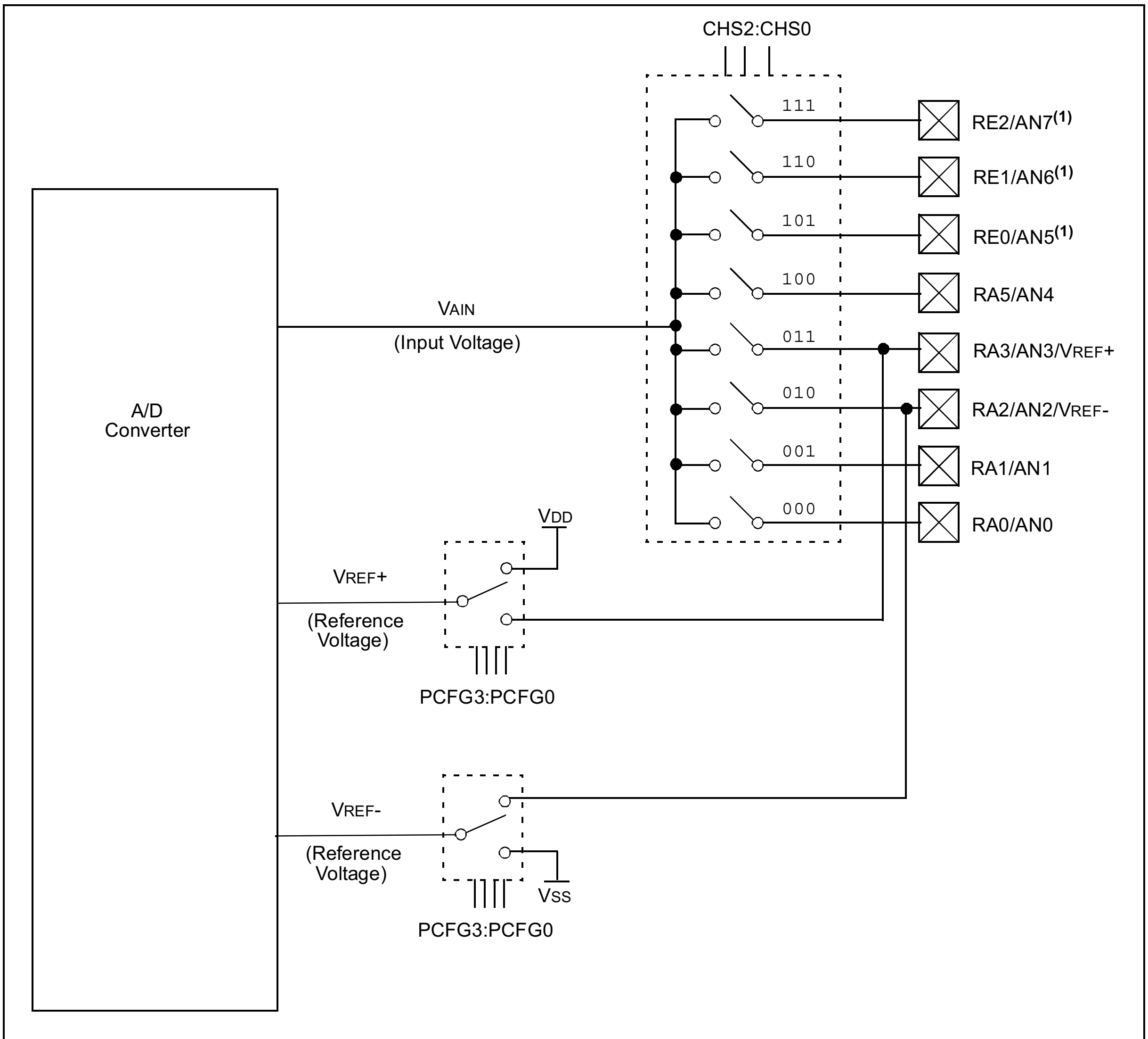
The ADRESH:ADRESL registers contain the 10-bit result of the A/D conversion. When the A/D conversion is complete, the result is loaded into this A/D result register pair, the GO/$\overline{\text{DONE}}$ bit (ADCON0<2>) is cleared and the A/D interrupt flag bit ADIF is set. The block diagram of the A/D module is shown in Figure 11-1.

After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as inputs. To determine sample time, see Section 11.1. After this acquisition time has elapsed, the A/D conversion can be started.

These steps should be followed for doing an A/D Conversion:

1. Configure the A/D module:
   - Configure analog pins/voltage reference and digital I/O (ADCON1)
   - Select A/D input channel (ADCON0)
   - Select A/D conversion clock (ADCON0)
   - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
   - Clear ADIF bit
   - Set ADIE bit
   - Set PEIE bit
   - Set GIE bit
3. Wait the required acquisition time.
4. Start conversion:
   - Set GO/$\overline{\text{DONE}}$ bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
   - Polling for the GO/$\overline{\text{DONE}}$ bit to be cleared (with interrupts enabled); OR
   - Waiting for the A/D interrupt
6. Read A/D result register pair (ADRESH:ADRESL), clear bit ADIF if required.
7. For the next conversion, go to step 1 or step 2, as required. The A/D conversion time per bit is defined as TAD. A minimum wait of 2TAD is required before the next acquisition starts.

FIGURE 11-1: A/D BLOCK DIAGRAM



Note 1: Not available on PIC16F873/876 devices.

11.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (C_{HOLD}) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 11-2. The source impedance (R_S) and the internal sampling switch (R_{SS}) impedance directly affect the time required to charge the capacitor C_{HOLD}. The sampling switch (R_{SS}) impedance varies over the device voltage (V_{DD}), see Figure 11-2. **The maximum recommended impedance for analog sources is 10 kΩ.** As the impedance is decreased, the acquisition time may be decreased.

After the analog input channel is selected (changed), this acquisition must be done before the conversion can be started.

To calculate the minimum acquisition time, Equation 11-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

To calculate the minimum acquisition time, T_{ACQ}, see the PICmicro™ Mid-Range Reference Manual (DS33023).

EQUATION 11-1: ACQUISITION TIME

$$\begin{aligned}
T_{ACQ} &= \text{Amplifier Settling Time} + \\
&\quad \text{Hold Capacitor Charging Time} + \\
&\quad \text{Temperature Coefficient} \\
&= T_{AMP} + T_C + T_{COFF} \\
&= 2\mu s + T_C + [(\text{Temperature} - 25°C)(0.05\mu s/°C)] \\
T_C &= C_{HOLD}\,(R_{IC} + R_{SS} + R_S)\ \text{In}(1/2047) \\
&= -120pF\,(1k\Omega + 7k\Omega + 10k\Omega)\ \text{In}(0.0004885) \\
&= 16.47\mu s \\
T_{ACQ} &= 2\mu s + 16.47\mu s + [(50°C - 25°C)(0.05\mu s/°C)] \\
&= 19.72\mu s
\end{aligned}$$

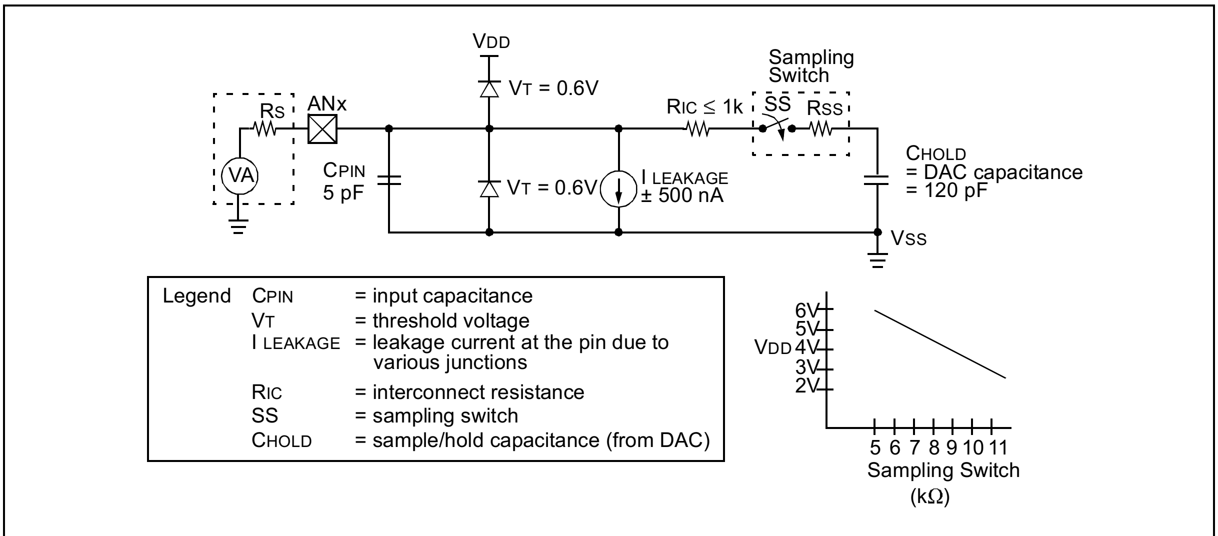
Note 1: The reference voltage (V_{REF}) has no effect on the equation, since it cancels itself out.

2: The charge holding capacitor (C_{HOLD}) is not discharged after each conversion.

3: The maximum recommended impedance for analog sources is 10 kΩ. This is required to meet the pin leakage specification.

4: After a conversion has completed, a 2.0T_{AD} delay must complete before acquisition can begin again. During this time, the holding capacitor is not connected to the selected A/D input channel.

FIGURE 11-2: ANALOG INPUT MODEL



11.2 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as T_{AD}. The A/D conversion requires a minimum $12T_{AD}$ per 10-bit conversion. The source of the A/D conversion clock is software selected. The four possible options for T_{AD} are:

- $2T_{OSC}$
- $8T_{OSC}$
- $32T_{OSC}$
- Internal A/D module RC oscillator (2-6 μs)

For correct A/D conversions, the A/D conversion clock (T_{AD}) must be selected to ensure a minimum T_{AD} time of 1.6 μs.

Table 11-1 shows the resultant T_{AD} times derived from the device operating frequencies and the A/D clock source selected.

TABLE 11-1: T_{AD} vs. MAXIMUM DEVICE OPERATING FREQUENCIES (STANDARD DEVICES (C))

AD Clock Source (T_{AD})		Maximum Device Frequency
Operation	**ADCS1:ADCS0**	**Max.**
$2T_{OSC}$	`00`	1.25 MHz
$8T_{OSC}$	`01`	5 MHz
$32T_{OSC}$	`10`	20 MHz
RC(1, 2, 3)	`11`	**(Note 1)**

Note 1: The RC source has a typical T_{AD} time of 4 μs, but can vary between 2-6 μs.
2: When the device frequencies are greater than 1 MHz, the RC A/D conversion clock source is only recommended for SLEEP operation.
3: For extended voltage devices (LC), please refer to the Electrical Characteristics (Sections 15.1 and 15.2).

11.3 Configuring Analog Port Pins

The ADCON1 and TRIS registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (V_{OH} or V_{OL}) will be converted.

The A/D operation is independent of the state of the CHS2:CHS0 bits and the TRIS bits.

> **Note 1:** When reading the port register, any pin configured as an analog input channel will read as cleared (a low level). Pins configured as digital inputs will convert an analog input. Analog levels on a digitally configured input will not affect the conversion accuracy.
>
> **2:** Analog levels on any pin that is defined as a digital input (including the AN7:AN0 pins), may cause the input buffer to consume current that is out of the device specifications.

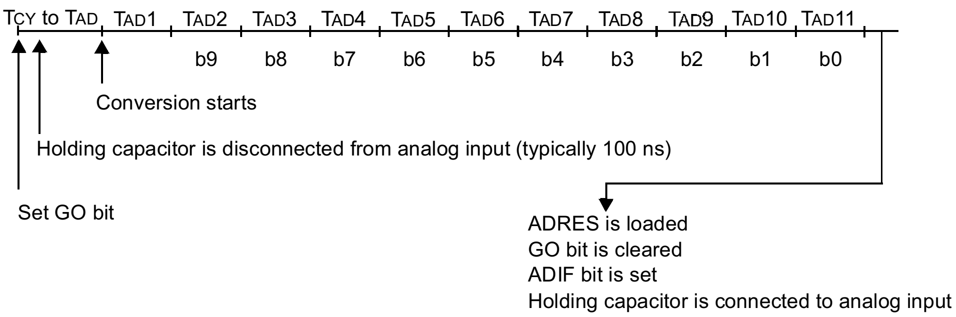
11.4 A/D Conversions

Clearing the GO/$\overline{\text{DONE}}$ bit during a conversion will abort the current conversion. The A/D result register pair will NOT be updated with the partially completed A/D conversion sample. That is, the ADRESH:ADRESL registers will continue to contain the value of the last completed conversion (or the last value written to the ADRESH:ADRESL registers). After the A/D conversion is aborted, a 2TAD wait is required before the next acquisition is started. After this 2TAD wait, acquisition on the selected channel is automatically started. The GO/$\overline{\text{DONE}}$ bit can then be set to start the conversion.

In Figure 11-3, after the GO bit is set, the first time segment has a minimum of TCY and a maximum of TAD.

Note:	The GO/$\overline{\text{DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D.

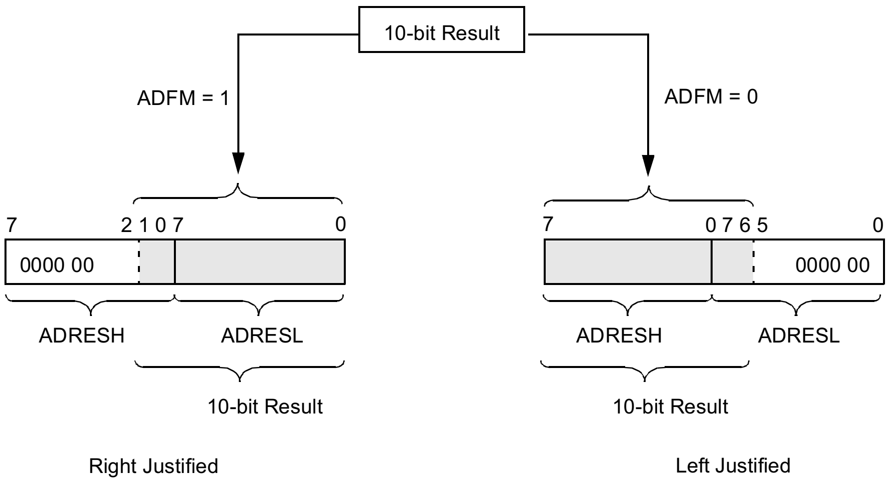
FIGURE 11-3: A/D CONVERSION TAD CYCLES



11.4.1 A/D RESULT REGISTERS

The ADRESH:ADRESL register pair is the location where the 10-bit A/D result is loaded at the completion of the A/D conversion. This register pair is 16-bits wide. The A/D module gives the flexibility to left or right justify the 10-bit result in the 16-bit result register. The A/D Format Select bit (ADFM) controls this justification. Figure 11-4 shows the operation of the A/D result justification. The extra bits are loaded with '0's'. When an A/D result will not overwrite these locations (A/D disable), these registers may be used as two general purpose 8-bit registers.

FIGURE 11-4: A/D RESULT JUSTIFICATION



11.5 A/D Operation During SLEEP

The A/D module can operate during SLEEP mode. This requires that the A/D clock source be set to RC (ADCS1:ADCS0 = 11). When the RC clock source is selected, the A/D module waits one instruction cycle before starting the conversion. This allows the SLEEP instruction to be executed, which eliminates all digital switching noise from the conversion. When the conversion is completed, the GO/$\overline{\text{DONE}}$ bit will be cleared and the result loaded into the ADRES register. If the A/D interrupt is enabled, the device will wake-up from SLEEP. If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

When the A/D clock source is another clock option (not RC), a SLEEP instruction will cause the present conversion to be aborted and the A/D module to be turned off, though the ADON bit will remain set.

Turning off the A/D places the A/D module in its lowest current consumption state.

Note:	For the A/D module to operate in SLEEP, the A/D clock source must be set to RC (ADCS1:ADCS0 = 11). To allow the conversion to occur during SLEEP, ensure the SLEEP instruction immediately follows the instruction that sets the GO/$\overline{\text{DONE}}$ bit.

11.6 Effects of a RESET

A device RESET forces all registers to their RESET state. This forces the A/D module to be turned off, and any conversion is aborted. All A/D input pins are configured as analog inputs.

The value that is in the ADRESH:ADRESL registers is not modified for a Power-on Reset. The ADRESH:ADRESL registers will contain unknown data after a Power-on Reset.

TABLE 11-2: REGISTERS/BITS ASSOCIATED WITH A/D

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on $\overline{\text{MCLR}}$, WDT
0Bh,8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
8Ch	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
1Eh	ADRESH	A/D Result Register High Byte								xxxx xxxx	uuuu uuuu
9Eh	ADRESL	A/D Result Register Low Byte								xxxx xxxx	uuuu uuuu
1Fh	ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON	0000 00-0	0000 00-0
9Fh	ADCON1	ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0	--0- 0000	--0- 0000
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	--11 1111
05h	PORTA	—	—	PORTA Data Latch when written: PORTA pins when read						--0x 0000	--0u 0000
89h(1)	TRISE	IBF	OBF	IBOV	PSPMODE	—	PORTE Data Direction bits			0000 -111	0000 -111
09h(1)	PORTE	—	—	—	—	—	RE2	RE1	RE0	---- -xxx	---- -uuu

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.
Note 1: These registers/bits are not available on the 28-pin devices.

FAIRCHILD
SEMICONDUCTOR™

October 1987
Revised January 1999

MM74C922 • MM74C923
16-Key Encoder • 20-Key Encoder

General Description

The MM74C922 and MM74C923 CMOS key encoders provide all the necessary logic to fully encode an array of SPST switches. The keyboard scan can be implemented by either an external clock or external capacitor. These encoders also have on-chip pull-up devices which permit switches with up to 50 kΩ on resistance to be used. No diodes in the switch array are needed to eliminate ghost switches. The internal debounce circuit needs only a single external capacitor and can be defeated by omitting the capacitor. A Data Available output goes to a high level when a valid keyboard entry has been made. The Data Available output returns to a low level when the entered key is released, even if another key is depressed. The Data Available will return high to indicate acceptance of the new key after a normal debounce period; this two-key roll-over is provided between any two switches.

An internal register remembers the last key pressed even after the key is released. The 3-STATE outputs provide for easy expansion and bus operation and are LPTTL compatible.

Features

- 50 kΩ maximum switch on resistance
- On or off chip clock
- On-chip row pull-up devices
- 2 key roll-over
- Keybounce elimination with single capacitor
- Last key register at outputs
- 3-STATE output LPTTL compatible
- Wide supply range: 3V to 15V
- Low power consumption

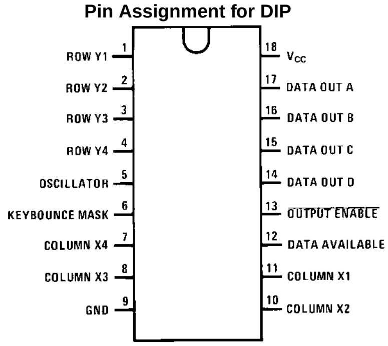
Ordering Code:

Order Number	Package Number	Package Description
MM74C922N	N18A	18-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300” Wide
MM74C922WM	M20B	20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300” Wide
MM74C923WM	M20B	20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300” Wide
MM74C923N	N20A	20-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300” Wide

Device also available in Tape and Reel. Specify by appending suffix letter “X” to the ordering code.

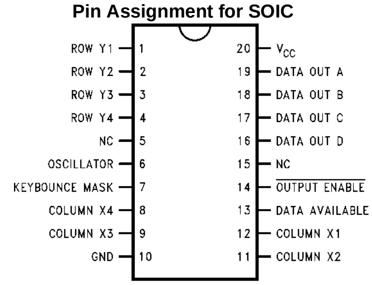
Connection Diagrams

Pin Assignment for DIP



Top View
MM94C922

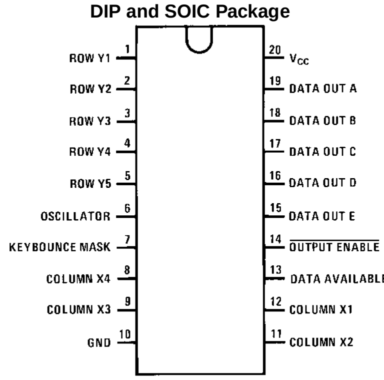
Pin Assignment for SOIC



Top View
MM74C922

Connection Diagrams (Continued)

Pin Assignment for
DIP and SOIC Package



Top View
MM74C923

Truth Tables

(Pins 0 through 11)

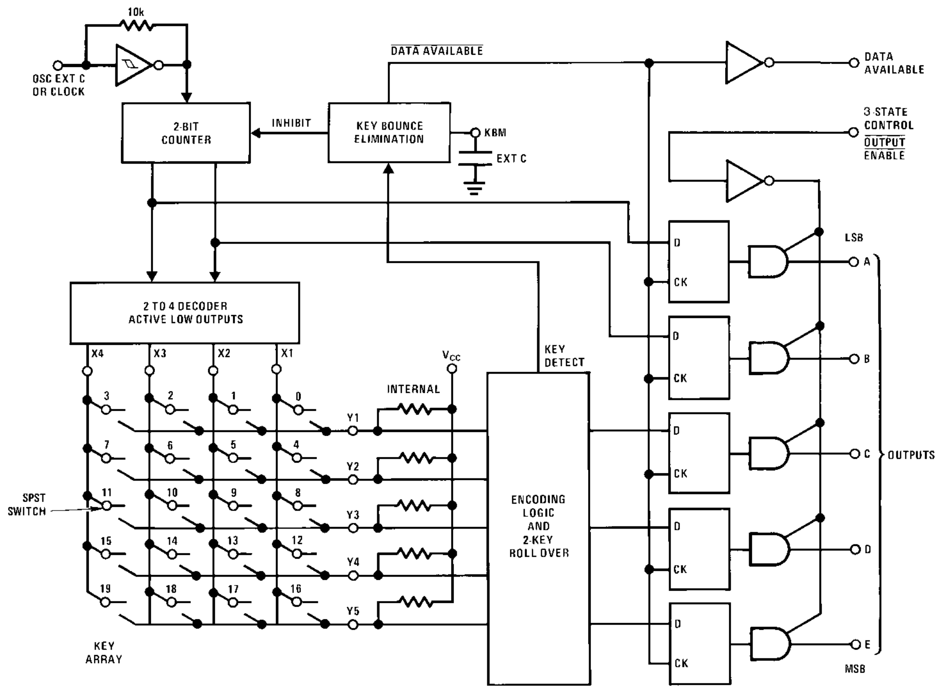
Switch Position		0 Y1,X1	1 Y1,X2	2 Y1,X3	3 Y1,X4	4 Y2,X1	5 Y2,X2	6 Y2,X3	7 Y2,X4	8 Y3,X1	9 Y3,X2	10 Y3,X3	11 Y3,X4
D													
A	A	0	1	0	1	0	1	0	1	0	1	0	1
T	B	0	0	1	1	0	0	1	1	0	0	1	1
A	C	0	0	0	0	1	1	1	1	0	0	0	0
O	D	0	0	0	0	0	0	0	0	1	1	1	1
U	E (Note 1)	0	0	0	0	0	0	0	0	0	0	0	0
T													

(Pins 12 through 19)

Switch Position		12 Y4,X1	13 Y4,X2	14 Y4,X3	15 Y4,X4	16 Y5(Note 1), X1	17 Y5 (Note 1), X2	18 Y5 (Note 1), X3	19 Y5 (Note 1), X4
D									
A	A	0	1	0	1	0	1	0	1
T	B	0	0	1	1	0	0	1	1
A	C	1	1	1	1	0	0	0	0
O	D	1	1	1	1	0	0	0	0
U	E (Note 1)	0	0	0	0	1	1	1	1
T									

Note 1: Omit for MM74C922

Block Diagram



Absolute Maximum Ratings (Note 2)

Voltage at Any Pin	V_{CC} – 0.3V to V_{CC} + 0.3V
Operating Temperature Range	
MM74C922, MM74C923	–40°C to +85°C
Storage Temperature Range	–65°C to +150°C
Power Dissipation (P_D)	
Dual-In-Line	700 mW
Small Outline	500 mW
Operating V_{CC} Range	3V to 15V
V_{CC}	18V
Lead Temperature	
(Soldering, 10 seconds)	260°C

Note 2: "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. Except for "Operating Temperature Range" they are not meant to imply that the devices should be operated at these limits. The table of "Electrical Characteristics" provides conditions for actual device operation.

DC Electrical Characteristics

Min/Max limits apply across temperature range unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Units
CMOS TO CMOS						
V_{T+}	Positive-Going Threshold Voltage at Osc and KBM Inputs	V_{CC} = 5V, I_{IN} ≥ 0.7 mA	3.0	3.6	4.3	V
		V_{CC} = 10V, I_{IN} ≥ 1.4 mA	6.0	6.8	8.6	V
		V_{CC} = 15V, I_{IN} ≥ 2.1 mA	9.0	10	12.9	V
V_{T-}	Negative-Going Threshold Voltage at Osc and KBM Inputs	V_{CC} = 5V, I_{IN} ≥ 0.7 mA	0.7	1.4	2.0	V
		V_{CC} = 10V, I_{IN} ≥ 1.4 mA	1.4	3.2	4.0	V
		V_{CC} = 15V, I_{IN} ≥ 2.1 mA	2.1	5	6.0	V
$V_{IN(1)}$	Logical "1" Input Voltage, Except Osc and KBM Inputs	V_{CC} = 5V	3.5	4.5		V
		V_{CC} = 10V	8.0	9		V
		V_{CC} = 15V	12.5	13.5		V
$V_{IN(0)}$	Logical "0" Input Voltage, Except Osc and KBM Inputs	V_{CC} = 5V		0.5	1.5	V
		V_{CC} = 10V		1	2	V
		V_{CC} = 15V		1.5	2.5	V
I_{rp}	Row Pull-Up Current at Y1, Y2, Y3, Y4 and Y5 Inputs	V_{CC} = 5V, V_{IN} = 0.1 V_{CC}		–2	–5	µA
		V_{CC} = 10V		–10	–20	µA
		V_{CC} = 15V		–22	–45	µA
$V_{OUT(1)}$	Logical "1" Output Voltage	V_{CC} = 5V, I_O = –10 µA	4.5			V
		V_{CC} = 10V, I_O = –10 µA	9			V
		V_{CC} = 15V, I_O = –10 µA	13.5			V
$V_{OUT(0)}$	Logical "0" Output Voltage	V_{CC} = 5V, I_O = 10 µA			0.5	V
		V_{CC} = 10V, I_O = 10 µA			1	V
		V_{CC} = 15V, I_O = 10 µA			1.5	V
R_{on}	Column "ON" Resistance at X1, X2, X3 and X4 Outputs	V_{CC} = 5V, V_O = 0.5V		500	1400	Ω
		V_{CC} = 10V, V_O = 1V		300	700	Ω
		V_{CC} = 15V, V_O = 1.5V		200	500	Ω
I_{CC}	Supply Current Osc at 0V, (one Y low)	V_{CC} = 5V		0.55	1.1	mA
		V_{CC} = 10V		1.1	1.9	mA
		V_{CC} = 15V		1.7	2.6	mA
$I_{IN(1)}$	Logical "1" Input Current at $\overline{\text{Output Enable}}$	V_{CC} = 15V, V_{IN} = 15V		0.005	1.0	µA
$I_{IN(0)}$	Logical "0" Input Current at $\overline{\text{Output Enable}}$	V_{CC} = 15V, V_{IN} = 0V	–1.0	–0.005		µA
CMOS/LPTTL INTERFACE						
$V_{IN(1)}$	Except Osc and KBM Inputs	V_{CC} = 4.75V	V_{CC} – 1.5			V
$V_{IN(0)}$	Except Osc and KBM Inputs	V_{CC} = 4.75V			0.8	V
$V_{OUT(1)}$	Logical "1" Output Voltage	I_O = –360 µA V_{CC} = 4.75V I_O = –360 µA	2.4			V
$V_{OUT(0)}$	Logical "0" Output Voltage	I_O = –360 µA V_{CC} = 4.75V I_O = –360 µA			0.4	V

DC Electrical Characteristics (Continued)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
OUTPUT DRIVE (See Family Characteristics Data Sheet) (Short Circuit Current)						
I_{SOURCE}	Output Source Current (P-Channel)	V_{CC} = 5V, V_{OUT} = 0V, T_A = 25°C	–1.75	–3.3		mA
I_{SOURCE}	Output Source Current (P-Channel)	V_{CC} = 10V, V_{OUT} = 0V, T_A = 25°C	–8	–15		mA
I_{SINK}	Output Sink Current (N-Channel)	V_{CC} = 5V, V_{OUT} = V_{CC}, T_A = 25°C	1.75	3.6		mA
I_{SINK}	Output Sink Current (N-Channel)	V_{CC} = 10V, V_{OUT} = V_{CC}, T_A = 25°C	8	16		mA

AC Electrical Characteristics (Note 3)

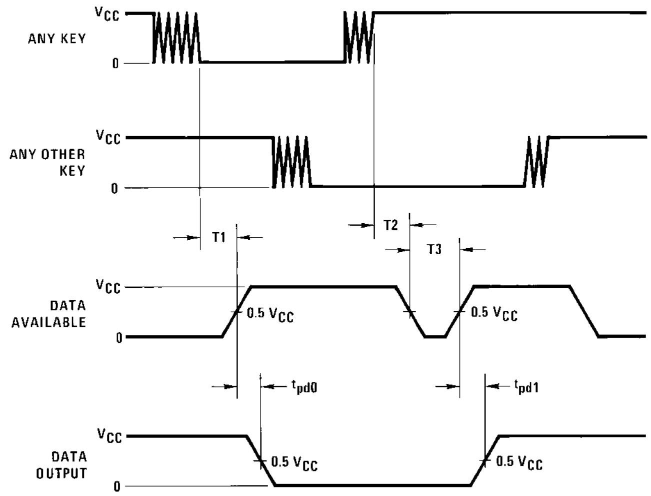
T_A = 25°C, C_L = 50 pF, unless otherwise noted

Symbol	Parameter	Conditions	Min	Typ	Max	Units
t_{pd0}, t_{pd1}	Propagation Delay Time to Logical “0” or Logical “1” from D.A.	C_L = 50 pF (Figure 1)				
		V_{CC} = 5V		60	150	ns
		V_{CC} = 10V		35	80	ns
		V_{CC} = 15V		25	60	ns
t_{0H}, t_{1H}	Propagation Delay Time from Logical “0” or Logical “1” into High Impedance State	R_L = 10k, C_L = 10 pF (Figure 2)				
		V_{CC} = 5V, R_L = 10k		80	200	ns
		V_{CC} = 10V, C_L = 10 pF		65	150	ns
		V_{CC} = 15V		50	110	ns
t_{H0}, t_{H1}	Propagation Delay Time from High Impedance State to a Logical “0” or Logical “1”	R_L = 10k, C_L = 50 pF (Figure 2)				
		V_{CC} = 5V, R_L = 10k		100	250	ns
		V_{CC} = 10V, C_L = 50 pF		55	125	ns
		V_{CC} = 15V		40	90	ns
C_{IN}	Input Capacitance	Any Input (Note 4)		5	7.5	pF
C_{OUT}	3-STATE Output Capacitance	Any Output (Note 4)		10		pF

Note 3: AC Parameters are guaranteed by DC correlated testing.

Note 4: Capacitance is guaranteed by periodic testing.

Switching Time Waveforms



T1 ≈ T2 ≈ RC, T3 ≈ 0.7 RC, where R ≈ 10k and C is external capacitor at KBM input.

FIGURE 1.

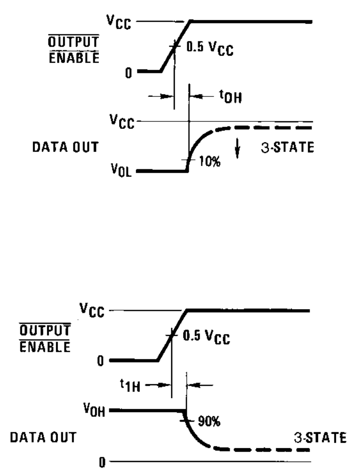
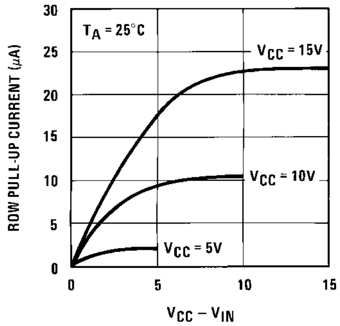


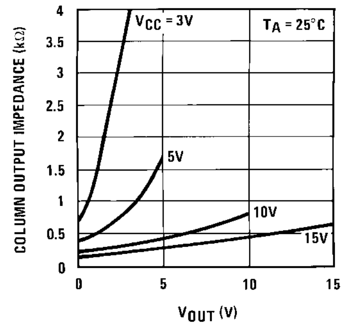
FIGURE 2.

Typical Performance Characteristics

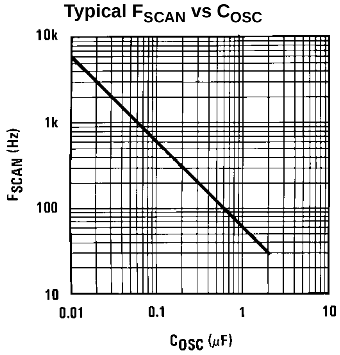
Typical I_{rp} vs V_{IN} at Any Y Input



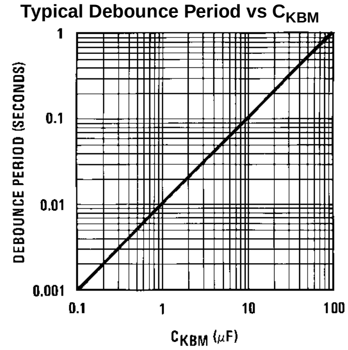
Typical R_{on} vs V_{OUT} at Any X Output



Typical F_{SCAN} vs C_{OSC}

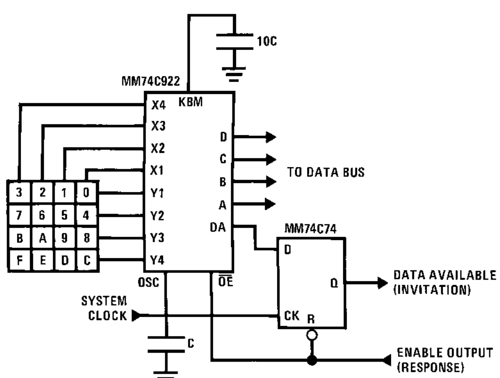


Typical Debounce Period vs C_{KBM}



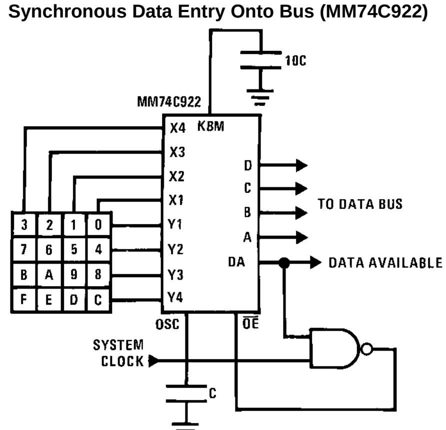
Typical Applications

Synchronous Handshake (MM74C922)



The keyboard may be synchronously scanned by omitting the capacitor at osc. and driving osc. directly if the system clock rate is lower than 10 kHz

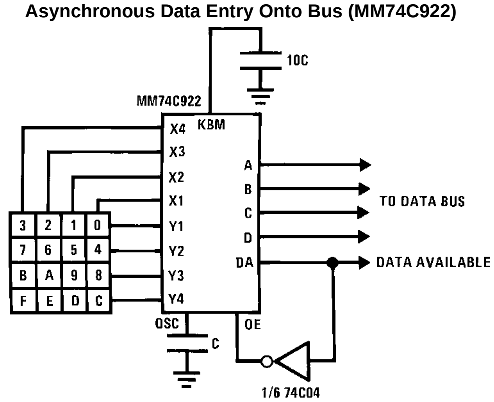
Synchronous Data Entry Onto Bus (MM74C922)



Outputs are enabled when valid entry is made and go into 3-STATE when key is released.

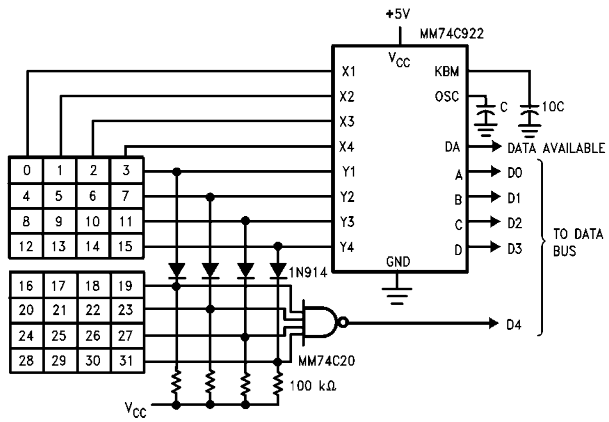
The keyboard may be synchronously scanned by omitting the capacitor at osc. and driving osc. directly if the system clock rate is lower than 10 kHz

Asynchronous Data Entry Onto Bus (MM74C922)



Outputs are in 3-STATE until key is pressed, then data is placed on bus. When key is released, outputs return to 3-STATE.

Expansion to 32 Key Encoder (MM74C922)



Theory of Operation

The MM74C922/MM74C923 Keyboard Encoders implement all the logic necessary to interface a 16 or 20 SPST key switch matrix to a digital system. The encoder will convert a key switch closer to a 4(MM74C922) or 5(MM74C923) bit nibble. The designer can control both the keyboard scan rate and the key debounce period by altering the oscillator capacitor, C_{OSE}, and the key bounce mask capacitor, C_{MSK}. Thus, the MM74C922/MM74C923's performance can be optimized for many keyboards.

The keyboard encoders connect to a switch matrix that is 4 rows by 4 columns (MM74C922) or 5 rows by 4 columns (MM74C923). When no keys are depressed, the row inputs are pulled high by internal pull-ups and the column outputs sequentially output a logic "0". These outputs are open drain and are therefore low for 25% of the time and otherwise off. The column scan rate is controlled by the oscillator input, which consists of a Schmitt trigger oscillator, a 2-bit counter, and a 2–4-bit decoder.

When a key is depressed, key 0, for example, nothing will happen when the X1 input is off, since Y1 will remain high. When the X1 column is scanned, X1 goes low and Y1 will go low. This disables the counter and keeps X1 low. Y1 going low also initiates the key bounce circuit timing and locks out the other Y inputs. The key code to be output is a combination of the frozen counter value and the decoded Y inputs. Once the key bounce circuit times out, the data is latched, and the Data Available (DAV) output goes high.

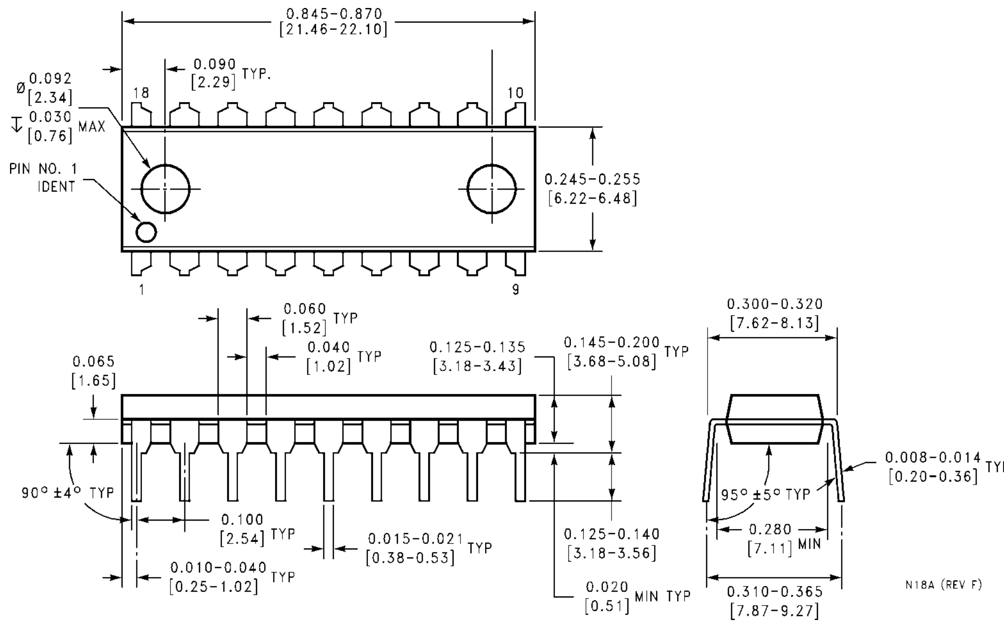
If, during the key closure the switch bounces, Y1 input will go high again, restarting the scan and resetting the key bounce circuitry. The key may bounce several times, but as soon as the switch stays low for a debounce period, the closure is assumed valid and the data is latched.

A key may also bounce when it is released. To ensure that the encoder does not recognize this bounce as another key closure, the debounce circuit must time out before another closure is recognized.

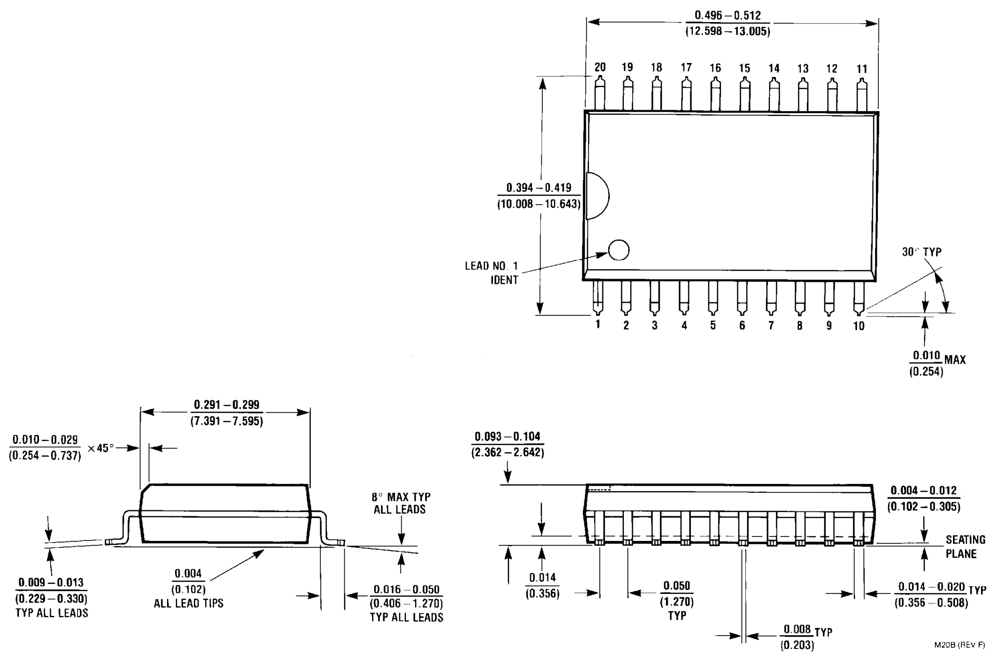
The two-key roll-over feature can be illustrated by assuming a key is depressed, and then a second key is depressed. Since all scanning has stopped, and all other Y inputs are disabled, the second key is not recognized until the first key is lifted and the key bounce circuitry has reset.

The output latches feed 3-STATE, which is enabled when the Output Enable ($\overline{OE}$) input is taken low.

Physical Dimensions inches (millimeters) unless otherwise noted

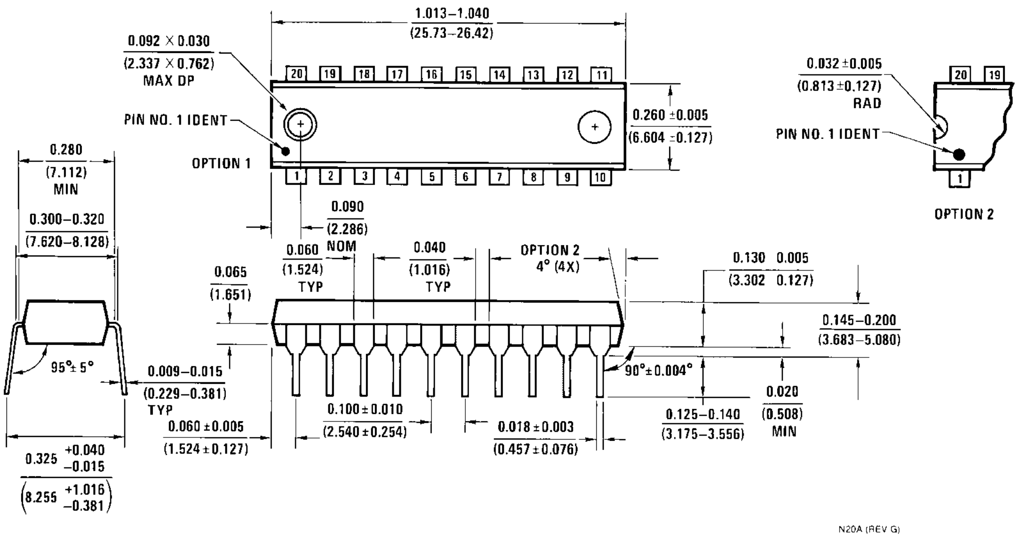


18-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300" Wide
Package Number N18A



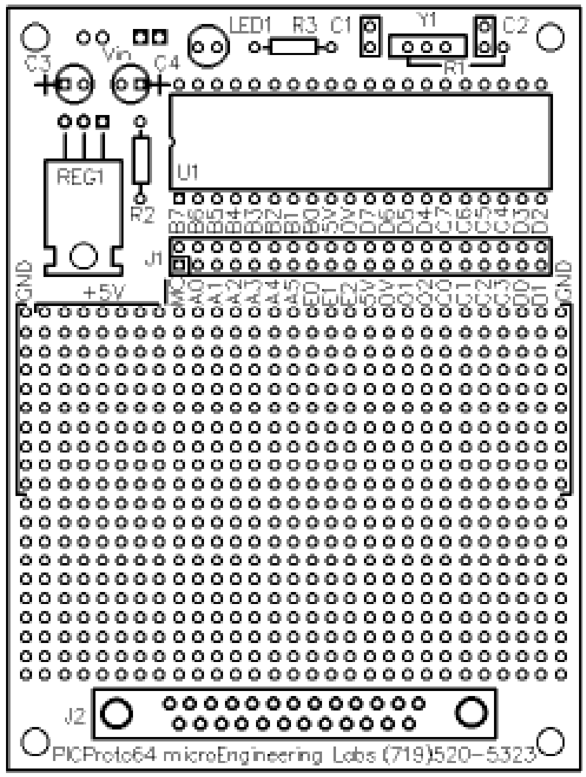
20-Lead Plastic Small Outline I.C. Package (M)
Package Number M20B

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

PICPROTO™64 Prototyping Board

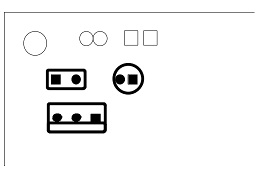


$16.95

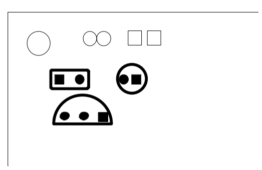
- High quality double-sided board
- Solder mask both sides
- More than 700 plated-through holes
- 4 mounting holes
- Overall dimensions 3" X 4"

U1 - 40-pin PIC® microcontroller
Y1 - crystal or ceramic resonator
C1, 2 - crystal capacitors
C3 - bypass capacitor
C4 - input capacitor
REG1 - 5 volt regulator
LED1 - LED
R1 - RC oscillator resistor
R2 - Master Clear resistor
R3 - LED series resistor
J1 - PIC I/O connector
J2 - DB9, 15, or 25

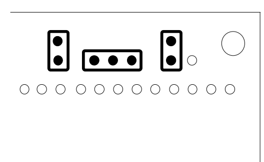
PARTS PLACEMENT:



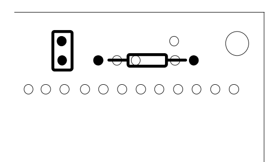
TO-220 Regulator
REG1 = 7805T
C3 = .01 - .1uf
C4 = .1 - 10uf



TO-92 Regulator
REG1 = 78L05
C3 = .01 - .1uf
C4 = .1 - 10uf



Crystal or Ceramic Resonator
Y1 = DC - 20MHZ
C1, 2 = 5 - 22pf



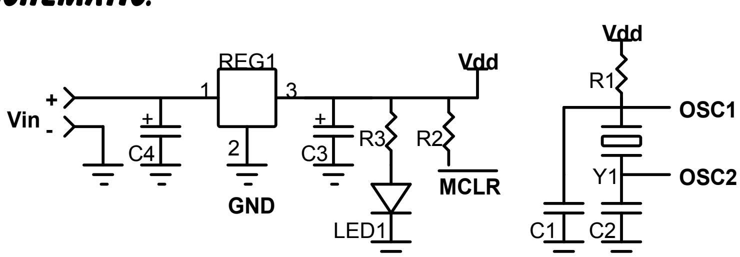
RC Oscillator
5k ≤ R ≤ 100K
C1 ≥ 20pf
C2 = none

ASSEMBLY NOTES:

Pin 1 of U1 is marked with a square pad.
Note polarity of Vin, REG1, LED1 and any polarized capacitors.

Don't forget to pull-up Master Clear to Vdd.
All unused inputs should be tied to +5V or ground.

SCHEMATIC:



SOURCES:

PIC® documentation is available from:
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler AZ 85224-6199
(480) 792-7200
(480) 792-7277 fax

microEngineering Labs, Inc.

Box 60039 Colorado Springs CO 80960 http://www.melabs.com
(719) 520-5323 (719) 520-1867 fax email: support@melabs.com

www.fairchildsemi.com

KA78XX/KA78XXA

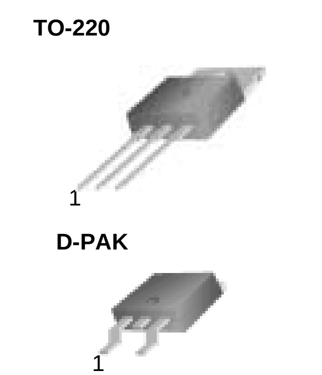
3-Terminal 1A Positive Voltage Regulator

Features

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

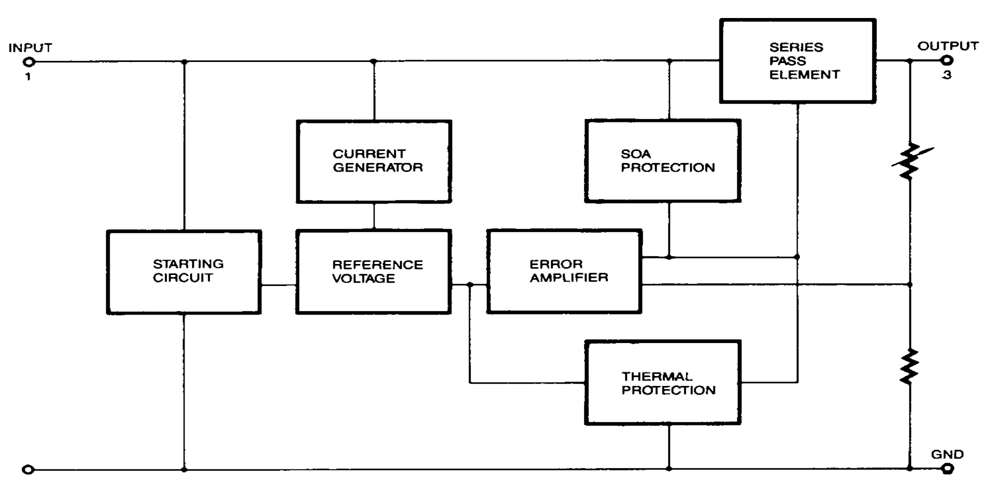
Description

The KA78XX/KA78XXA series of three-terminal positive regulator are available in the TO-220/D-PAK package and with several fixed output voltages, making them useful in a wide range of applications. Each type employs internal current limiting, thermal shut down and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.



1. Input 2. GND 3. Output

Internal Block Digram



Rev. 1.0.0

Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Input Voltage (for V_O = 5V to 18V) (for V_O = 24V)	V_I V_I	35 40	V V
Thermal Resistance Junction-Cases (TO-220)	$R_{\theta JC}$	5	°C/W
Thermal Resistance Junction-Air (TO-220)	$R_{\theta JA}$	65	°C/W
Operating Temperature Range (KA78XX/A/R)	T_{OPR}	0 ~ +125	°C
Storage Temperature Range	T_{STG}	-65 ~ +150	°C

Electrical Characteristics (KA7805/KA7805R)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I =10V, C_I= 0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		KA7805 Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25 °C		4.8	5.0	5.2	V
		5.0mA ≤ Io ≤ 1.0A, P_O ≤ 15W V_I = 7V to 20V		4.75	5.0	5.25	
Line Regulation (Note1)	Regline	T_J=+25 °C	V_O = 7V to 25V	-	4.0	100	mV
			V_I = 8V to 12V	-	1.6	50	
Load Regulation (Note1)	Regload	T_J=+25 °C	I_O = 5.0mA to1.5A	-	9	100	mV
			I_O =250mA to 750mA	-	4	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.0	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	0.03	0.5	mA
		V_I= 7V to 25V		-	0.3	1.3	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O= 5mA		-	-0.8	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A=+25 °C		-	42	-	µV/V_O
Ripple Rejection	RR	f = 120Hz V_O = 8V to 18V		62	73	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	15	-	mΩ
Short Circuit Current	I_{SC}	V_I = 35V, T_A =+25 °C		-	230	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7806/KA7806R)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I =11V, C_I= 0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		KA7806 Min.	KA7806 Typ.	KA7806 Max.	Unit
Output Voltage	V_O	T_J =+25 °C		5.75	6.0	6.25	V
		5.0mA ≤ I_O ≤ 1.0A, P_O ≤ 15W V_I = 8.0V to 21V		5.7	6.0	6.3	
Line Regulation (Note1)	Regline	T_J =+25 °C	V_I = 8V to 25V	-	5	120	mV
			V_I = 9V to 13V	-	1.5	60	
Load Regulation (Note1)	Regload	T_J =+25 °C	I_O =5mA to 1.5A	-	9	120	mV
			I_O =250mA to750mA	-	3	60	
Quiescent Current	I_Q	T_J =+25 °C		-	5.0	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1A		-	-	0.5	mA
		V_I = 8V to 25V		-	-	1.3	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O = 5mA		-	-0.8	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A =+25 °C		-	45	-	µV/Vo
Ripple Rejection	RR	f = 120Hz V_I = 9V to 19V		59	75	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	19	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A=+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7808/KA7808R)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I =14V, C_I= 0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		KA7808			Unit
				Min.	Typ.	Max.	
Output Voltage	V_O	T_J =+25 °C		7.7	8.0	8.3	V
		5.0mA ≤ I_O ≤ 1.0A, P_O ≤ 15W V_I = 10.5V to 23V		7.6	8.0	8.4	
Line Regulation (Note1)	Regline	T_J =+25 °C	V_I = 10.5V to 25V	-	5.0	160	mV
			V_I = 11.5V to 17V	-	2.0	80	
Load Regulation (Note1)	Regload	T_J =+25 °C	I_O = 5.0mA to 1.5A	-	10	160	mV
			I_O= 250mA to 750mA	-	5.0	80	
Quiescent Current	I_Q	T_J =+25 °C		-	5.0	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	0.05	0.5	mA
		V_I = 10.5A to 25V		-	0.5	1.0	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O = 5mA		-	-0.8	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A =+25 °C		-	52	-	µV/Vo
Ripple Rejection	RR	f = 120Hz, V_I= 11.5V to 21.5V		56	73	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J=+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	17	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 °C		-	230	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7809/KA7809R)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I =15V, C_I= 0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		KA7809 Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25 °C		8.65	9	9.35	V
		5.0mA≤ I_O ≤1.0A, P_O ≤15W V_I= 11.5V to 24V		8.6	9	9.4	
Line Regulation (Note1)	Regline	T_J=+25 °C	V_I = 11.5V to 25V	-	6	180	mV
			V_I = 12V to 17V	-	2	90	
Load Regulation (Note1)	Regload	T_J=+25 °C	I_O = 5mA to 1.5A	-	12	180	mV
			I_O = 250mA to 750mA	-	4	90	
Quiescent Current	I_Q	T_J=+25 °C		-	5.0	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	-	0.5	mA
		V_I = 11.5V to 26V		-	-	1.3	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O = 5mA		-	-1	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A =+25 °C		-	58	-	µV/Vo
Ripple Rejection	RR	f = 120Hz V_I = 13V to 23V		56	71	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J=+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	17	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J= +25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7810)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I =16V, C_I= 0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		KA7810 Min.	KA7810 Typ.	KA7810 Max.	Unit
Output Voltage	V_O	T_J =+25 °C		9.6	10	10.4	V
		5.0mA ≤ I_O ≤ 1.0A, P_O ≤ 15W V_I = 12.5V to 25V		9.5	10	10.5	
Line Regulation (Note1)	Regline	T_J =+25 °C	V_I = 12.5V to 25V	-	10	200	mV
			V_I = 13V to 25V	-	3	100	
Load Regulation (Note1)	Regload	T_J =+25 °C	I_O = 5mA to 1.5A	-	12	200	mV
			I_O = 250mA to 750mA	-	4	400	
Quiescent Current	I_Q	T_J =+25 °C		-	5.1	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	-	0.5	mA
		V_I = 12.5V to 29V		-	-	1.0	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O = 5mA		-	-1	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A =+25 °C		-	58	-	µV/Vo
Ripple Rejection	RR	f = 120Hz V_I = 13V to 23V		56	71	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J=+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	17	-	mΩ
Short Circuit Current	I_{SC}	V_I = 35V, T_A=+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7812/KA7812R)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I =19V, C_I= 0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		KA7812/KA7812R Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25 °C		11.5	12	12.5	V
		5.0mA ≤ I_O≤1.0A, P_O≤15W V_I = 14.5V to 27V		11.4	12	12.6	
Line Regulation (Note1)	Regline	T_J =+25 °C	V_I = 14.5V to 30V	-	10	240	mV
			V_I = 16V to 22V	-	3.0	120	
Load Regulation (Note1)	Regload	T_J =+25 °C	I_O = 5mA to 1.5A	-	11	240	mV
			I_O = 250mA to 750mA	-	5.0	120	
Quiescent Current	I_Q	T_J =+25 °C		-	5.1	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	0.1	0.5	mA
		V_I = 14.5V to 30V		-	0.5	1.0	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O = 5mA		-	-1	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A =+25 °C		-	76	-	µV/Vo
Ripple Rejection	RR	f = 120Hz V_I = 15V to 25V		55	71	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J=+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	18	-	mΩ
Short Circuit Current	I_{SC}	V_I = 35V, T_A=+25 °C		-	230	-	mA
Peak Current	I_{PK}	T_J = +25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7815)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I =23V, C_I= 0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		KA7815			Unit
				Min.	Typ.	Max.	
Output Voltage	V_O	T_J =+25 °C		14.4	15	15.6	V
		5.0mA ≤ I_O≤1.0A, P_O≤15W V_I = 17.5V to 30V		14.25	15	15.75	
Line Regulation (Note1)	Regline	T_J =+25 °C	V_I = 17.5V to 30V	-	11	300	mV
			V_I = 20V to 26V	-	3	150	
Load Regulation (Note1)	Regload	T_J =+25 °C	I_O = 5mA to 1.5A	-	12	300	mV
			I_O = 250mA to 750mA	-	4	150	
Quiescent Current	I_Q	T_J =+25 °C		-	5.2	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	-	0.5	mA
		V_I = 17.5V to 30V		-	-	1.0	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O = 5mA		-	-1	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A =+25 °C		-	90	-	µV/Vo
Ripple Rejection	RR	f = 120Hz V_I = 18.5V to 28.5V		54	70	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J=+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	19	-	mΩ
Short Circuit Current	I_{SC}	V_I = 35V, T_A=+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7818)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I =27V, C_I= 0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		KA7818 Min.	KA7818 Typ.	KA7818 Max.	Unit
Output Voltage	V_O	T_J =+25 °C		17.3	18	18.7	V
		5.0mA ≤ I_O ≤1.0A, P_O ≤15W V_I = 21V to 33V		17.1	18	18.9	
Line Regulation (Note1)	Regline	T_J =+25 °C	V_I = 21V to 33V	-	15	360	mV
			V_I = 24V to 30V	-	5	180	
Load Regulation (Note1)	Regload	T_J =+25 °C	I_O = 5mA to 1.5A	-	15	360	mV
			I_O = 250mA to 750mA	-	5.0	180	
Quiescent Current	I_Q	T_J =+25 °C		-	5.2	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	-	0.5	mA
		V_I = 21V to 33V		-	-	1	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O = 5mA		-	-1	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A =+25 °C		-	110	-	µV/Vo
Ripple Rejection	RR	f = 120Hz V_I = 22V to 32V		53	69	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J=+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	22	-	mΩ
Short Circuit Current	I_{SC}	V_I = 35V, T_A=+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7824)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I =33V, C_I= 0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		KA7824 Min.	KA7824 Typ.	KA7824 Max.	Unit
Output Voltage	V_O	T_J =+25 °C		23	24	25	V
		5.0mA ≤ I_O ≤ 1.0A, P_O ≤ 15W V_I = 27V to 38V		22.8	24	25.25	
Line Regulation (Note1)	Regline	T_J =+25 °C	V_I = 27V to 38V	-	17	480	mV
			V_I = 30V to 36V	-	6	240	
Load Regulation (Note1)	Regload	T_J =+25 °C	I_O = 5mA to 1.5A	-	15	480	mV
			I_O = 250mA to 750mA	-	5.0	240	
Quiescent Current	I_Q	T_J =+25 °C		-	5.2	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	0.1	0.5	mA
		V_I = 27V to 38V		-	0.5	1	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O = 5mA		-	-1.5	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A =+25 °C		-	60	-	µV/Vo
Ripple Rejection	RR	f = 120Hz V_I = 28V to 38V		50	67	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J=+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	28	-	mΩ
Short Circuit Current	I_{SC}	V_I = 35V, T_A=+25 °C		-	230	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7805A)

(Refer to the test circuits. $0^oC < T_J < +125\ ^oC$, I_o =1A, V_I = 10V, C_I=0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25 ^oC		4.9	5	5.1	V
		I_O = 5mA to 1A, $P_O \le$ 15W V_I = 7.5V to 20V		4.8	5	5.2	
Line Regulation (Note1)	Regline	V_I = 7.5V to 25V I_O = 500mA		-	5	50	mV
		V_I = 8V to 12V		-	3	50	
		T_J =+25 ^oC	V_I= 7.3V to 20V	-	5	50	
			V_I= 8V to 12V	-	1.5	25	
Load Regulation (Note1)	Regload	T_J =+25 ^oC I_O = 5mA to 1.5A		-	9	100	mV
		I_O = 5mA to 1A		-	9	100	
		I_O = 250mA to 750mA		-	4	50	
Quiescent Current	I_Q	T_J =+25 ^oC		-	5.0	6.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1A		-	-	0.5	mA
		V_I = 8 V to 25V, I_O = 500mA		-	-	0.8	
		V_I = 7.5V to 20V, T_J =+25 ^oC		-	-	0.8	
Output Voltage Drift	$\Delta V/\Delta T$	Io = 5mA		-	-0.8	-	mV/ ^oC
Output Noise Voltage	V_N	f = 10Hz to 100KHz T_A =+25 ^oC		-	10	-	µV/Vo
Ripple Rejection	RR	f = 120Hz, I_O = 500mA V_I = 8V to 18V		-	68	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 ^oC		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	17	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 ^oC		-	250	-	mA
Peak Current	I_{PK}	T_J= +25 ^oC		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7806A)

(Refer to the test circuits. $0^oC < T_J < +125\ ^oC$, I_O =1A, V_I = 11V, C_I=0.33μF, C_O=0.1μF, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25 ^oC		5.58	6	6.12	V
		I_O = 5mA to 1A, $P_O \leq$ 15W V_I = 8.6V to 21V		5.76	6	6.24	
Line Regulation (Note1)	Regline	V_I= 8.6V to 25V I_O = 500mA		-	5	60	mV
		V_I= 9V to 13V		-	3	60	
		T_J =+25 ^oC	V_I= 8.3V to 21V	-	5	60	
			V_I= 9V to 13V	-	1.5	30	
Load Regulation (Note1)	Regload	T_J =+25 ^oC I_O = 5mA to 1.5A		-	9	100	mV
		I_O = 5mA to 1A		-	4	100	
		I_O = 250mA to 750mA		-	5.0	50	
Quiescent Current	I_Q	T_J =+25 ^oC		-	4.3	6.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1A		-	-	0.5	mA
		V_I = 9V to 25V, I_O = 500mA		-	-	0.8	
		V_I= 8.5V to 21V, T_J =+25 ^oC		-	-	0.8	
Output Voltage Drift	$\Delta V/\Delta T$	I_O = 5mA		-	-0.8	-	mV/ ^oC
Output Noise Voltage	V_N	f = 10Hz to 100KHz T_A =+25 ^oC		-	10	-	μV/Vo
Ripple Rejection	RR	f = 120Hz, I_O = 500mA V_I = 9V to 19V		-	65	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 ^oC		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	17	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 ^oC		-	250	-	mA
Peak Current	I_{PK}	T_J=+25 ^oC		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7808A)

(Refer to the test circuits. 0°C < T_J < +125 °C, I_O =1A, V_I = 14V, C_I=0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25 °C		7.84	8	8.16	V
		I_O = 5mA to 1A, P_O ≤15W V_I = 10.6V to 23V		7.7	8	8.3	
Line Regulation (Note1)	Regline	V_I= 10.6V to 25V I_O = 500mA		-	6	80	mV
		V_I= 11V to 17V		-	3	80	
		T_J =+25 °C	V_I= 10.4V to 23V	-	6	80	
			V_I= 11V to 17V	-	2	40	
Load Regulation (Note1)	Regload	T_J =+25 °C I_O = 5mA to 1.5A		-	12	100	mV
		I_O = 5mA to 1A		-	12	100	
		I_O = 250mA to 750mA		-	5	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.0	6.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1A		-	-	0.5	mA
		V_I = 11V to 25V, I_O = 500mA		-	-	0.8	
		V_I= 10.6V to 23V, T_J =+25 °C		-	-	0.8	
Output Voltage Drift	$\Delta V/\Delta T$	I_O = 5mA		-	-0.8	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz T_A =+25 °C		-	10	-	µV/Vo
Ripple Rejection	RR	f = 120Hz, I_O = 500mA V_I = 11.5V to 21.5V		-	62	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	18	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J=+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7809A)

(Refer to the test circuits. $0^oC < T_J < +125\ ^oC$, I_O =1A, V_I = 15V, C_I=0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25°C		8.82	9.0	9.18	V
		I_O = 5mA to 1A, P_O≤15W V_I = 11.2V to 24V		8.65	9.0	9.35	
Line Regulation (Note1)	Regline	V_I= 11.7V to 25V I_O = 500mA		-	6	90	mV
		V_I= 12.5V to 19V		-	4	45	
		T_J =+25°C	V_I= 11.5V to 24V	-	6	90	
			V_I= 12.5V to 19V	-	2	45	
Load Regulation (Note1)	Regload	T_J =+25°C I_O = 5mA to 1.0A		-	12	100	mV
		I_O = 5mA to 1.0A		-	12	100	
		I_O = 250mA to 750mA		-	5	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.0	6.0	mA
Quiescent Current Change	ΔI_Q	V_I = 11.7V to 25V, T_J=+25 °C		-	-	0.8	mA
		V_I = 12V to 25V, I_O = 500mA		-	-	0.8	
		I_O = 5mA to 1.0A		-	-	0.5	
Output Voltage Drift	ΔV/ΔT	I_O = 5mA		-	-1.0	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz T_A =+25 °C		-	10	-	µV/Vo
Ripple Rejection	RR	f = 120Hz, I_O = 500mA V_I = 12V to 22V		-	62	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 °C		-	2.0	-	V
Output Resistance	r_O	f = 1KHz		-	17	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J=+25°C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7810A)

(Refer to the test circuits. $0^{o}C < T_J < +125\,^{o}C$, I_O =1A, V_I = 16V, C_I=0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25°C		9.8	10	10.2	V
		I_O = 5mA to 1A, $P_O \le$ 15W V_I =12.8V to 25V		9.6	10	10.4	
Line Regulation (Note1)	Regline	V_I= 12.8V to 26V I_O = 500mA		-	8	100	mV
		V_I= 13V to 20V		-	4	50	
		T_J =+25 °C	V_I= 12.5V to 25V	-	8	100	
			V_I= 13V to 20V	-	3	50	
Load Regulation (Note1)	Regload	T_J =+25 °C I_O = 5mA to 1.5A		-	12	100	mV
		I_O = 5mA to 1.0A		-	12	100	
		I_O = 250mA to 750mA		-	5	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.0	6.0	mA
Quiescent Current Change	ΔI_Q	V_I = 13V to 26V, T_J=+25 °C		-	-	0.5	mA
		V_I = 12.8V to 25V, I_O = 500mA		-	-	0.8	
		I_O = 5mA to 1.0A		-	-	0.5	
Output Voltage Drift	$\Delta V/\Delta T$	I_O = 5mA		-	-1.0	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz T_A =+25 °C		-	10	-	µV/Vo
Ripple Rejection	RR	f = 120Hz, I_O = 500mA V_I = 14V to 24V		-	62	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25°C		-	2.0	-	V
Output Resistance	r_O	f = 1KHz		-	17	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J=+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7812A)

(Refer to the test circuits. 0°C < T_J < +125 °C, I_O =1A, V_I = 19V, C_I=0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25 °C		11.75	12	12.25	V
		I_O = 5mA to 1A, P_O ≤15W V_I = 14.8V to 27V		11.5	12	12.5	
Line Regulation (Note1)	Regline	V_I= 14.8V to 30V I_O = 500mA		-	10	120	mV
		V_I= 16V to 22V		-	4	120	
		T_J =+25 °C	V_I= 14.5V to 27V	-	10	120	
			V_I= 16V to 22V	-	3	60	
Load Regulation (Note1)	Regload	T_J =+25 °C I_O = 5mA to 1.5A		-	12	100	mV
		I_O = 5mA to 1.0A		-	12	100	
		I_O = 250mA to 750mA		-	5	50	
Quiescent Current	I_Q	T_J =+25°C		-	5.1	6.0	mA
Quiescent Current Change	ΔI_Q	V_I = 15V to 30V, T_J=+25 °C		-		0.8	mA
		V_I = 14V to 27V, I_O = 500mA		-		0.8	
		I_O = 5mA to 1.0A		-		0.5	
Output Voltage Drift	ΔV/ΔT	I_O = 5mA		-	-1.0	-	mV/°C
Output Noise Voltage	V_N	f = 10Hz to 100KHz T_A =+25°C		-	10	-	µV/Vo
Ripple Rejection	RR	f = 120Hz, I_O = 500mA V_I = 14V to 24V		-	60	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25°C		-	2.0	-	V
Output Resistance	r_O	f = 1KHz		-	18	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J=+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7815A)

(Refer to the test circuits. 0°C < T_J < +125 °C, I_O =1A, V_I =23V, C_I=0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25 °C		14.7	15	15.3	V
		I_O = 5mA to 1A, P_O ≤15W V_I = 17.7V to 30V		14.4	15	15.6	
Line Regulation (Note1)	Regline	V_I= 17.9V to 30V I_O = 500mA		-	10	150	mV
		V_I= 20V to 26V		-	5	150	
		T_J =+25°C	V_I= 17.5V to 30V	-	11	150	
			V_I= 20V to 26V	-	3	75	
Load Regulation (Note1)	Regload	T_J =+25 °C I_O = 5mA to 1.5A		-	12	100	mV
		I_O = 5mA to 1.0A		-	12	100	
		I_O = 250mA to 750mA		-	5	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.2	6.0	mA
Quiescent Current Change	ΔI_Q	V_I = 17.5V to 30V, T_J =+25 °C		-	-	0.8	mA
		V_I = 17.5V to 30V, I_O = 500mA		-	-	0.8	
		I_O = 5mA to 1.0A		-	-	0.5	
Output Voltage Drift	$\Delta V/\Delta T$	I_O = 5mA		-	-1.0	-	mV/°C
Output Noise Voltage	V_N	f = 10Hz to 100KHz T_A =+25 °C		-	10	-	µV/Vo
Ripple Rejection	RR	f = 120Hz, I_O = 500mA V_I = 18.5V to 28.5V		-	58	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 °C		-	2.0	-	V
Output Resistance	r_O	f = 1KHz		-	19	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J=+25°C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7818A)

(Refer to the test circuits. $0^oC < T_J < +125\,^oC$, I_O =1A, V_I = 27V, C_I=0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25 °C		17.64	18	18.36	V
		I_O = 5mA to 1A, P_O ≤15W V_I = 21V to 33V		17.3	18	18.7	
Line Regulation (Note1)	Regline	V_I= 21V to 33V I_O = 500mA		-	15	180	mV
		V_I= 21V to 33V		-	5	180	
		T_J =+25 °C	V_I= 20.6V to 33V	-	15	180	
			V_I= 24V to 30V	-	5	90	
Load Regulation (Note1)	Regload	T_J =+25°C I_O = 5mA to 1.5A		-	15	100	mV
		I_O = 5mA to 1.0A		-	15	100	
		I_O = 250mA to 750mA		-	7	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.2	6.0	mA
Quiescent Current Change	ΔI_Q	V_I = 21V to 33V, T_J=+25 °C		-	-	0.8	mA
		V_I = 21V to 33V, I_O = 500mA		-	-	0.8	
		I_O = 5mA to 1.0A		-	-	0.5	
Output Voltage Drift	$\Delta V/\Delta T$	I_O = 5mA		-	-1.0	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz T_A =+25°C		-	10	-	µV/Vo
Ripple Rejection	RR	f = 120Hz, I_O = 500mA V_I = 22V to 32V		-	57	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25°C		-	2.0	-	V
Output Resistance	r_O	f = 1KHz		-	19	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25°C		-	250	-	mA
Peak Current	I_{PK}	T_J=+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Electrical Characteristics (KA7824A)

(Refer to the test circuits. $0^oC < T_J < +125\ ^oC$, I_O =1A, V_I = 33V, C_I=0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25 °C		23.5	24	24.5	V
		I_O = 5mA to 1A, P_O ≤15W V_I = 27.3V to 38V		23	24	25	
Line Regulation (Note1)	Regline	V_I= 27V to 38V I_O = 500mA		-	18	240	mV
		V_I= 21V to 33V		-	6	240	
		T_J =+25 °C	V_I= 26.7V to 38V	-	18	240	
			V_I= 30V to 36V	-	6	120	
Load Regulation (Note1)	Regload	T_J =+25 °C I_O = 5mA to 1.5A		-	15	100	mV
		I_O = 5mA to 1.0A		-	15	100	
		I_O = 250mA to 750mA		-	7	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.2	6.0	mA
Quiescent Current Change	ΔI_Q	V_I = 27.3V to 38V, T_J =+25 °C		-	-	0.8	mA
		V_I = 27.3V to 38V, I_O = 500mA		-	-	0.8	
		I_O = 5mA to 1.0A		-	-	0.5	
Output Voltage Drift	ΔV/ΔT	I_O = 5mA		-	-1.5	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz T_A = 25 °C		-	10	-	µV/Vo
Ripple Rejection	RR	f = 120Hz, I_O = 500mA V_I = 28V to 38V		-	54	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 °C		-	2.0	-	V
Output Resistance	r_O	f = 1KHz		-	20	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 °C		-	250	-	mA
Peak Current	I_{PK}	T_J=+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Typical Perfomance Characteristics

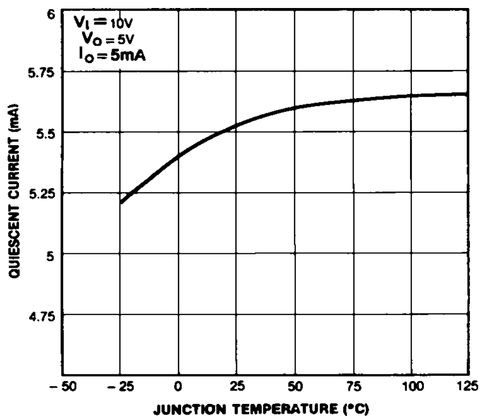


Figure 1. Quiescent Current

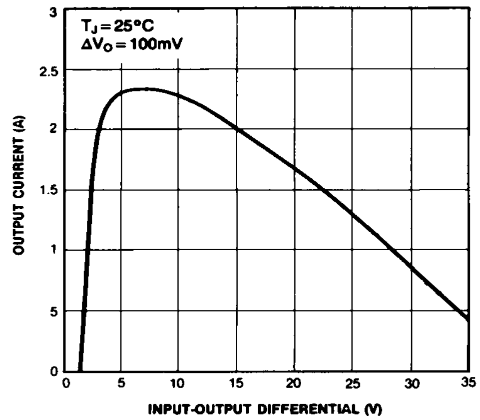


Figure 2. Peak Output Current

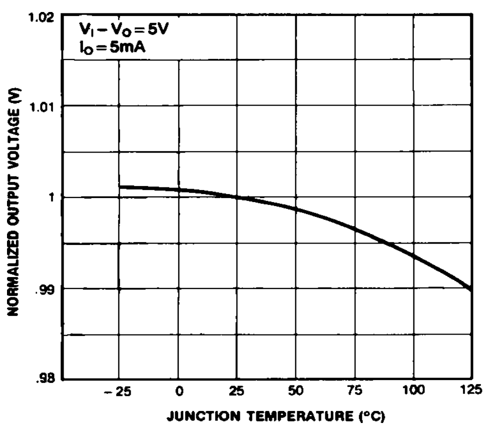


Figure 3. Output Voltage

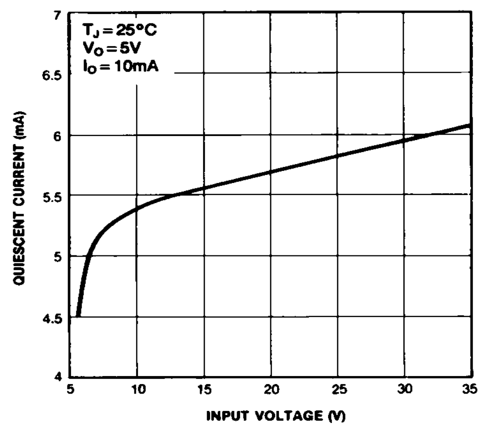


Figure 4. Quiescent Current

Typical Applications

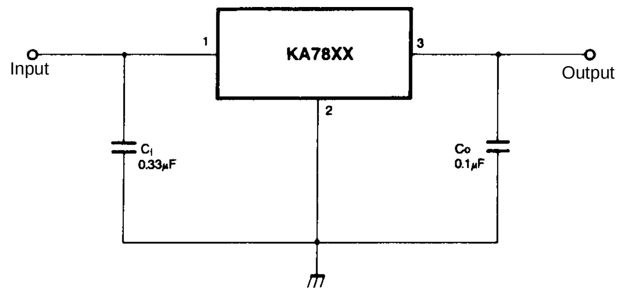


Figure 5. DC Parameters

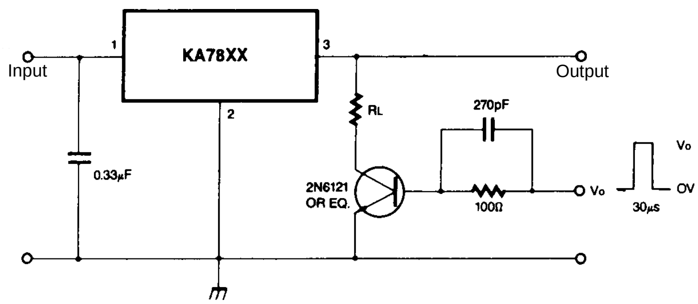


Figure 6. Load Regulation

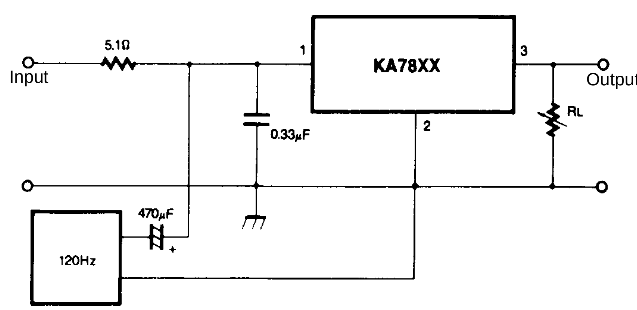


Figure 7. Ripple Rejection

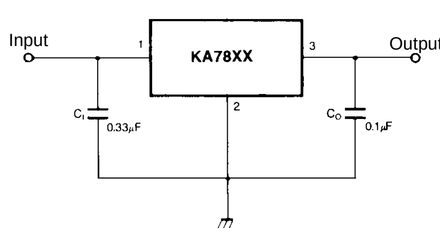
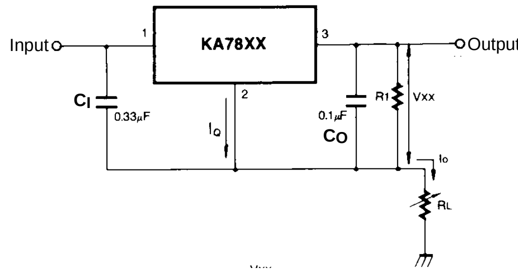


Figure 8. Fixed Output Regulator



$$I_O = \frac{V_{XX}}{R_1} + I_Q$$

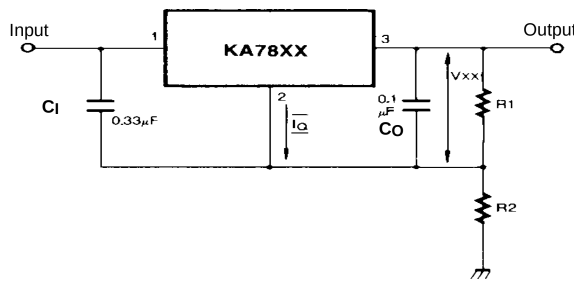
Figure 9. Constant Current Regulator

Notes:

(1) To specify an output voltage. substitute voltage value for "XX." A common ground is required between the input and the Output voltage. The input voltage must remain typically 2.0V above the output voltage even during the low point on the input ripple voltage.

(2) C_I is required if regulator is located an appreciable distance from power Supply filter.

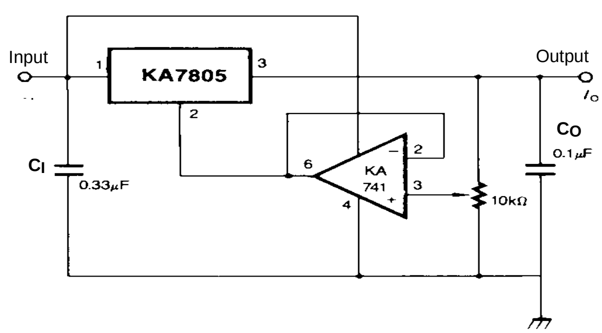
(3) C_O improves stability and transient response.



$I_{RI} \geq 5IQ$

$V_O = V_{XX}(1+R_2/R_1)+I_QR_2$

Figure 10. Circuit for Increasing Output Voltage



$I_{RI} \geq 5\ I_Q$

$V_O = V_{XX}(1+R_2/R_1)+I_QR_2$

Figure 11. Adjustable Output Regulator (7 to 30V)

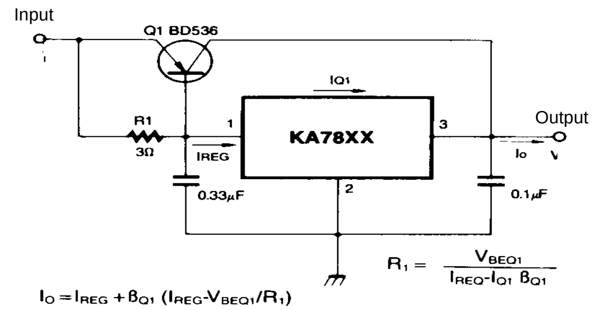


Figure 12. High Current Voltage Regulator

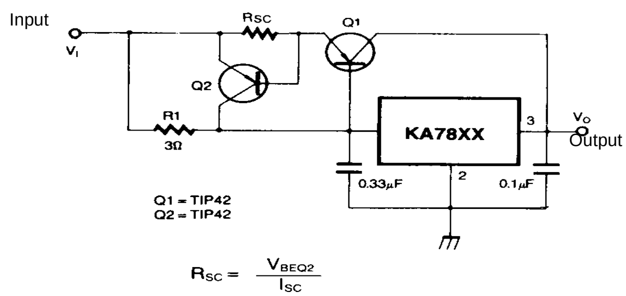


Figure 13. High Output Current with Short Circuit Protection

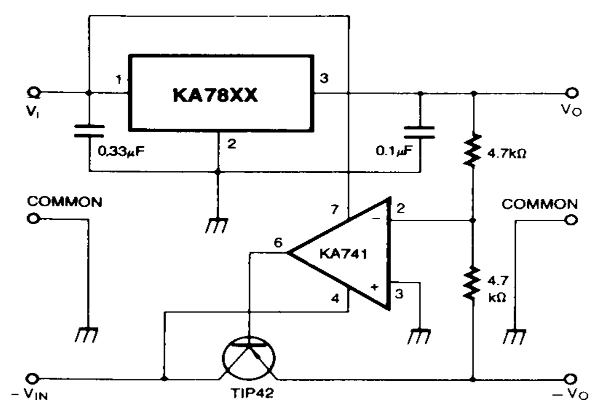


Figure 14. Tracking Voltage Regulator

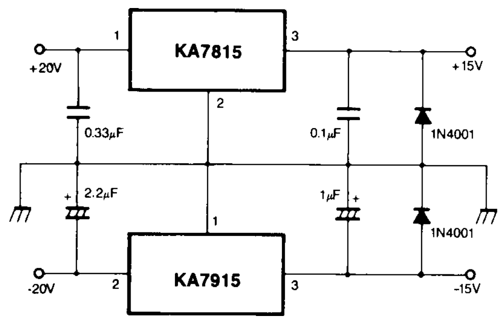


Figure 15. Split Power Supply (±15V-1A)

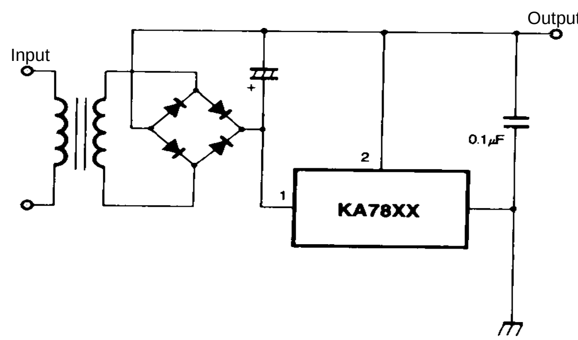


Figure 16. Negative Output Voltage Circuit

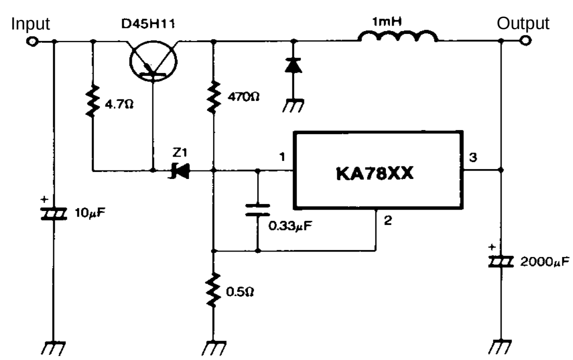
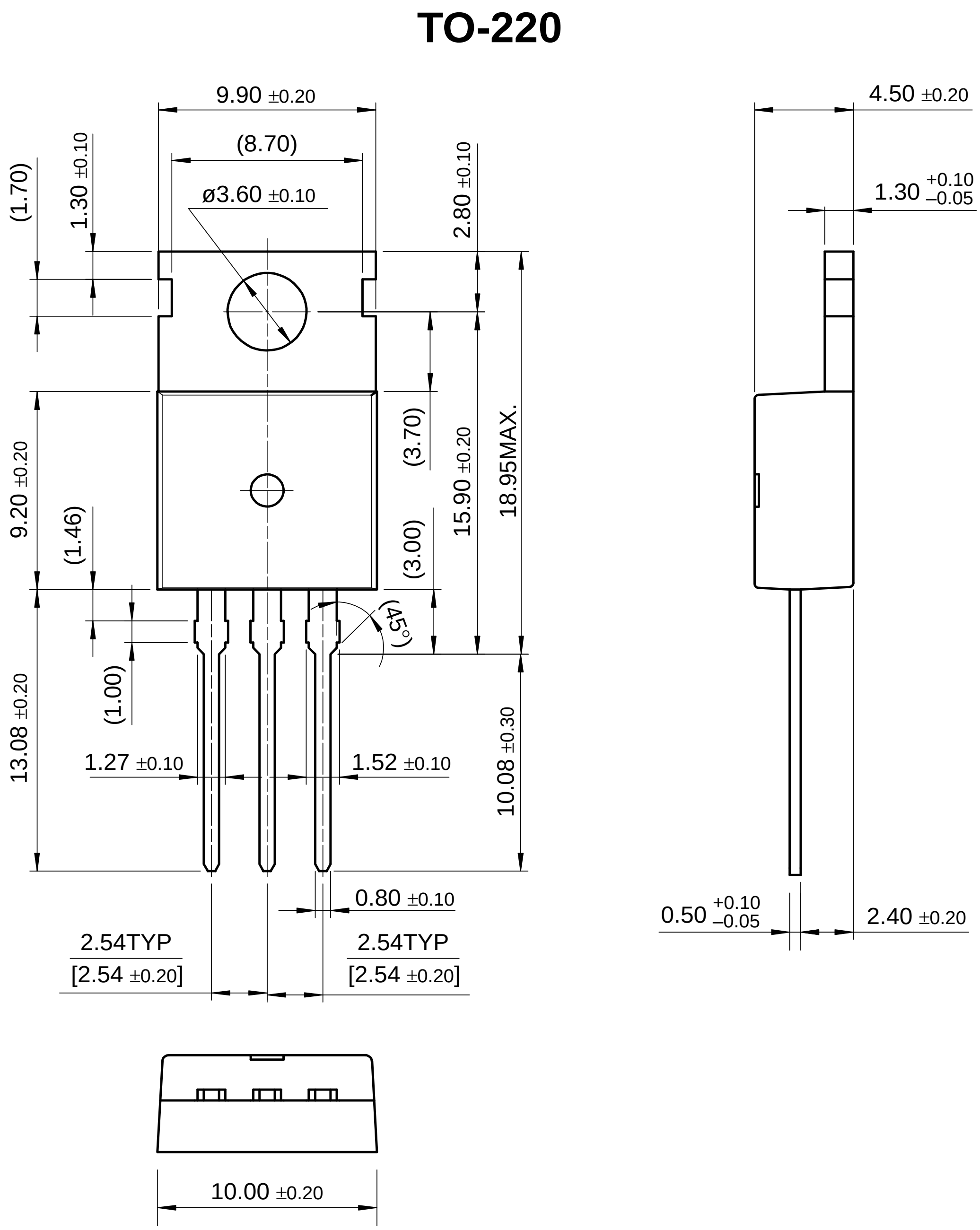


Figure 17. Switching Regulator

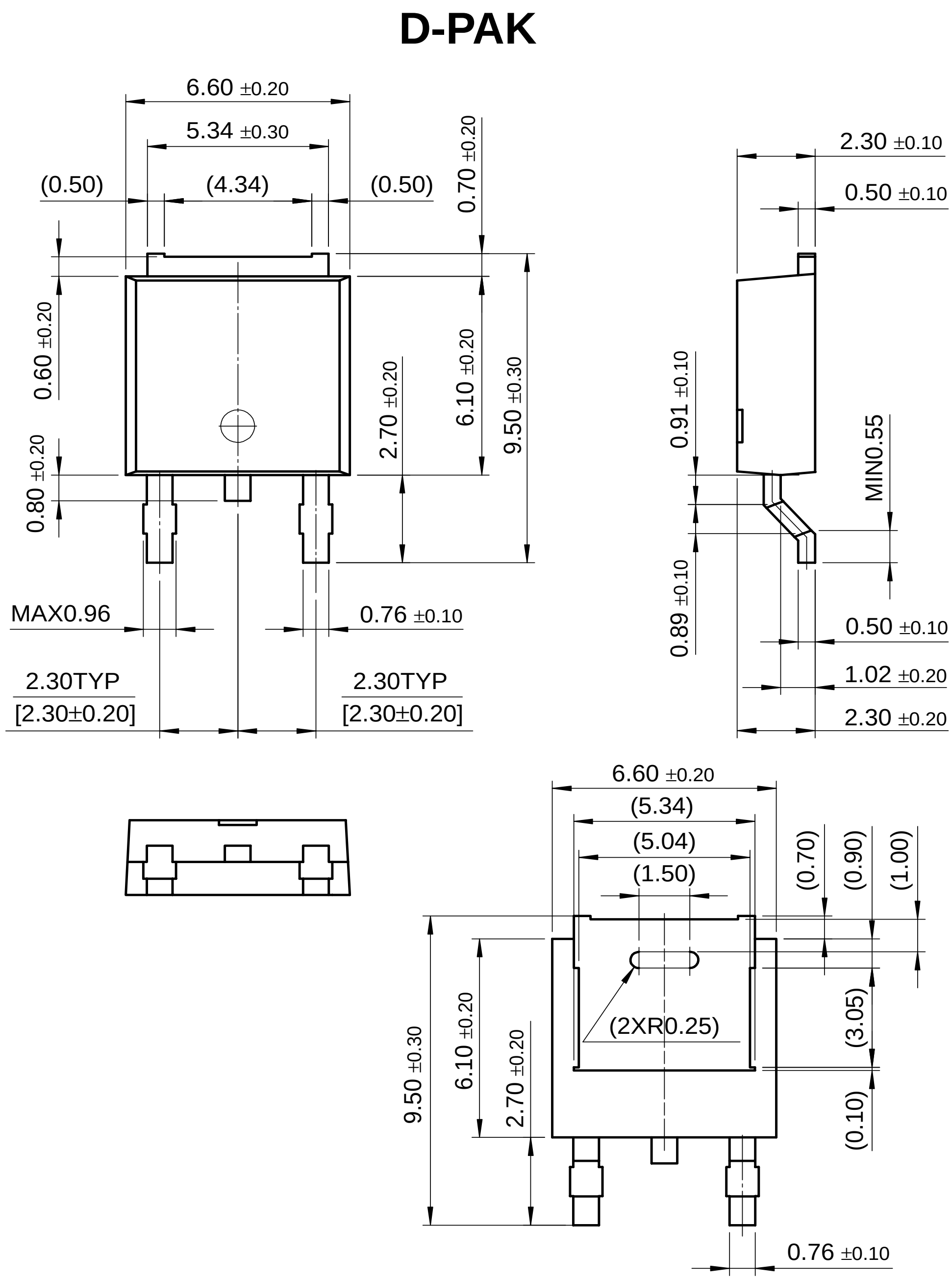
Mechanical Dimensions

Package



Mechancal Dimensions (Continued)

Package



Ordering Information

Product Number	Output Voltage Tolerance	Package	Operating Temperature
KA7805 / KA7806	±4%	TO-220	0 ~ + 125°C
KA7808 / KA7809			
KA7810			
KA7812 / KA7815			
KA7818 / KA7824			
KA7805A / KA7806A	±2%		
KA7808A / KA7809A			
KA7810A / KA7812A			
KA7815A / KA7818A			
KA7824A			
KA7805R / KA7806R	±4%	D-PAK	
KA7808R / KA7809R			
KA7812R			

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:

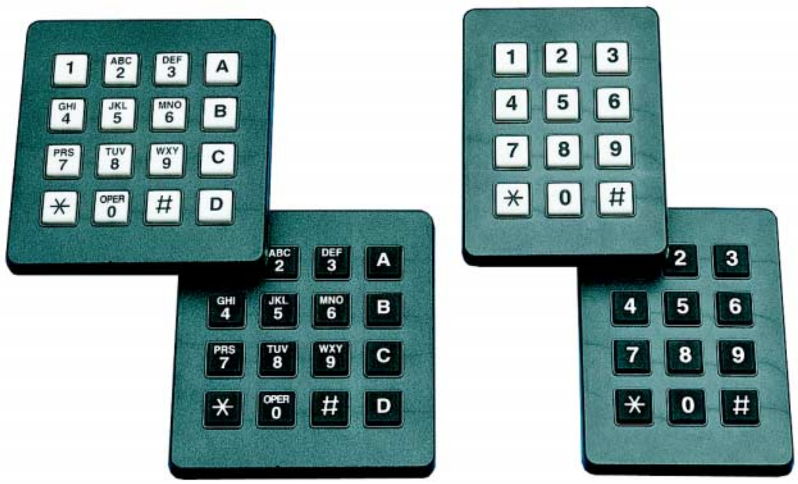
1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result 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.

6/1/01 0.0m 001
Stock#DSxxxxxxxx

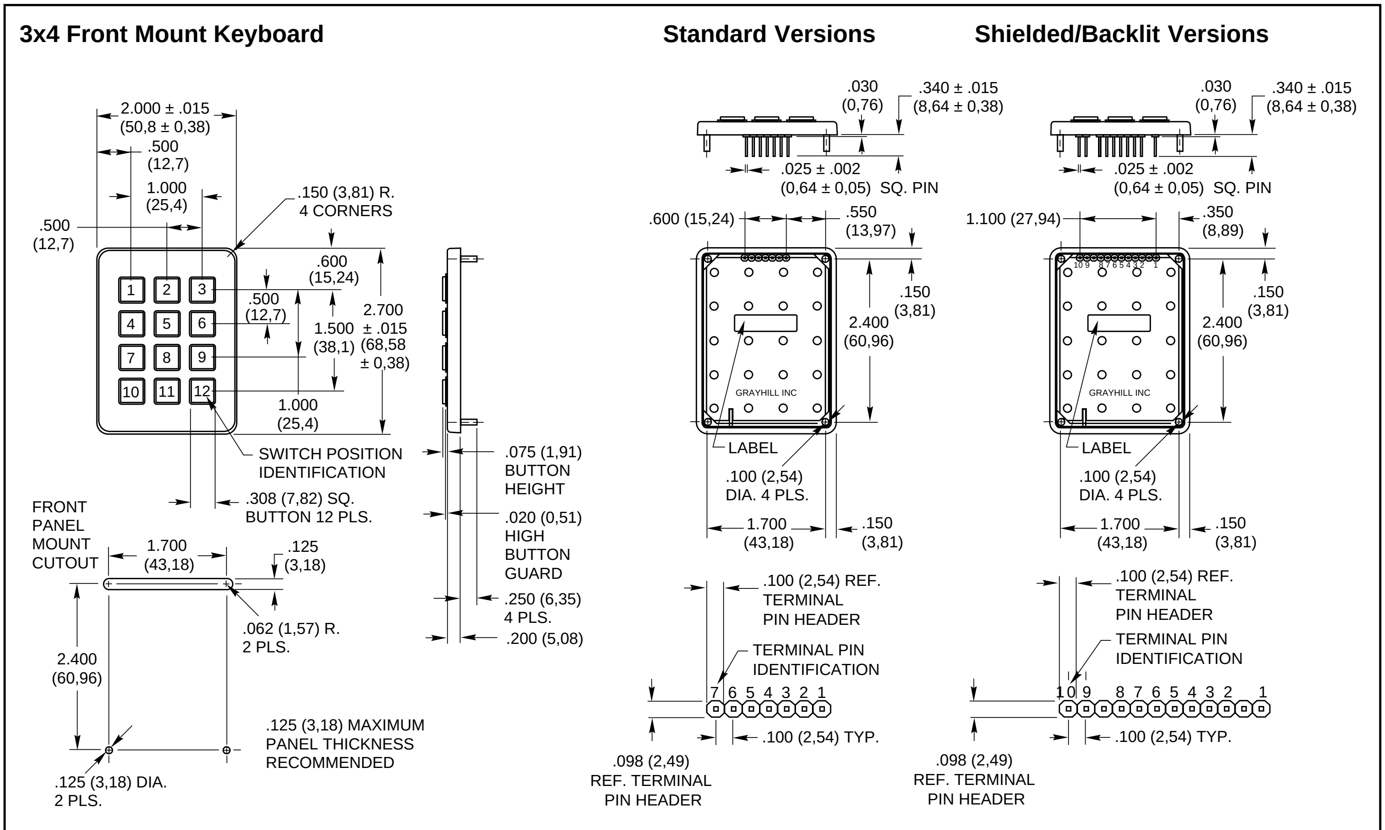
SERIES 96
Conductive Rubber

FEATURES
- Quality, Economical Keyboards
- Easily Customized Legends
- Matrix Circuitry
- Backlit and Shielded Options Available
- Termination Mates With Standard Connectors
- Tactile Feedback to Operator
- 1,000,000 Operations per Button
- Compatible With High Resistance Logic Inputs

The Series 96 is Grayhill's most economical 3x4 and 4x4 keypad family. The contact system utilizes conductive rubber to mate the appropriate PC board traces. Offered in matrix circuitry, with shielded and backlit options. Built with quality component parts, the Series 96 is subjected to our rigid statistical process control to insure that it meets our reliability standards.



DIMENSIONS In inches (and millimeters)

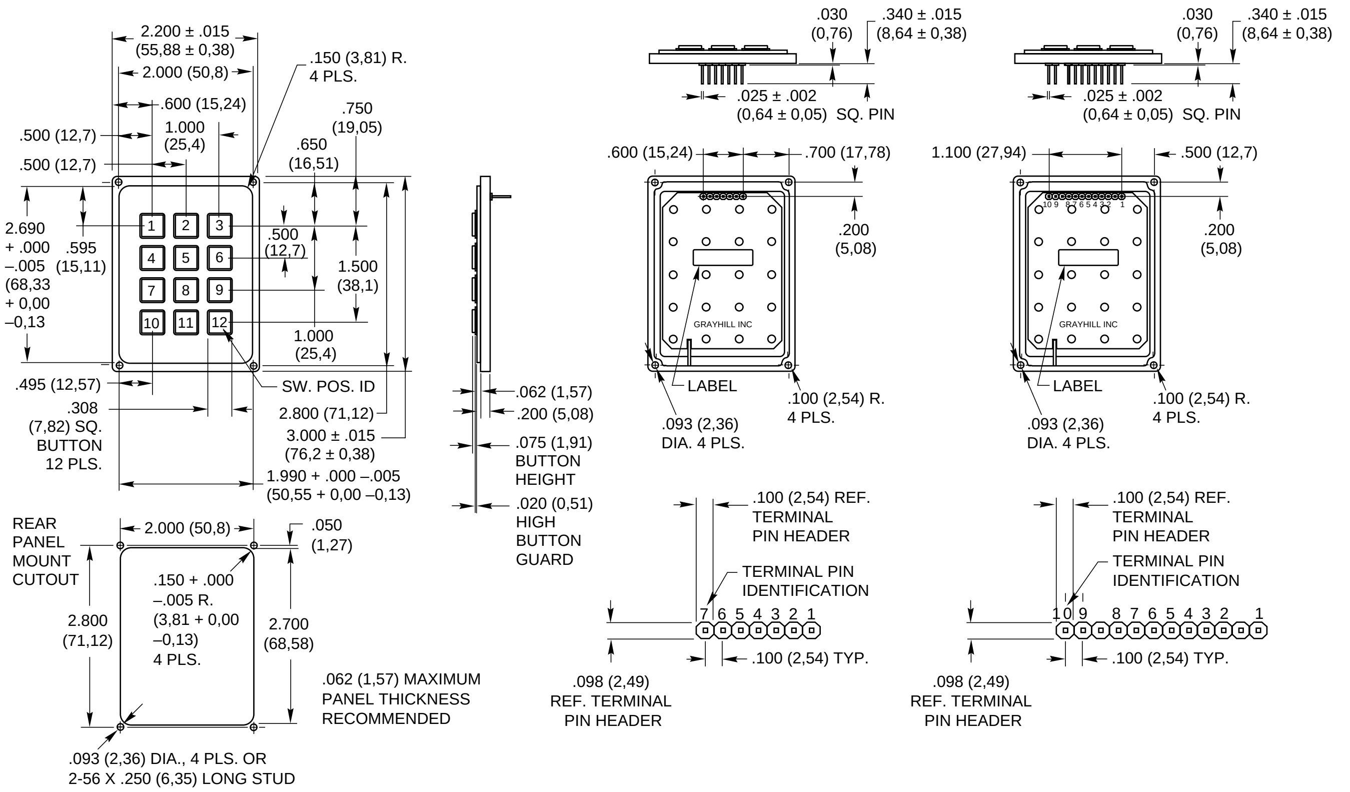


DIMENSIONS In inches (and millimeters)

3x4 Rear Mount Keyboard

Standard Versions

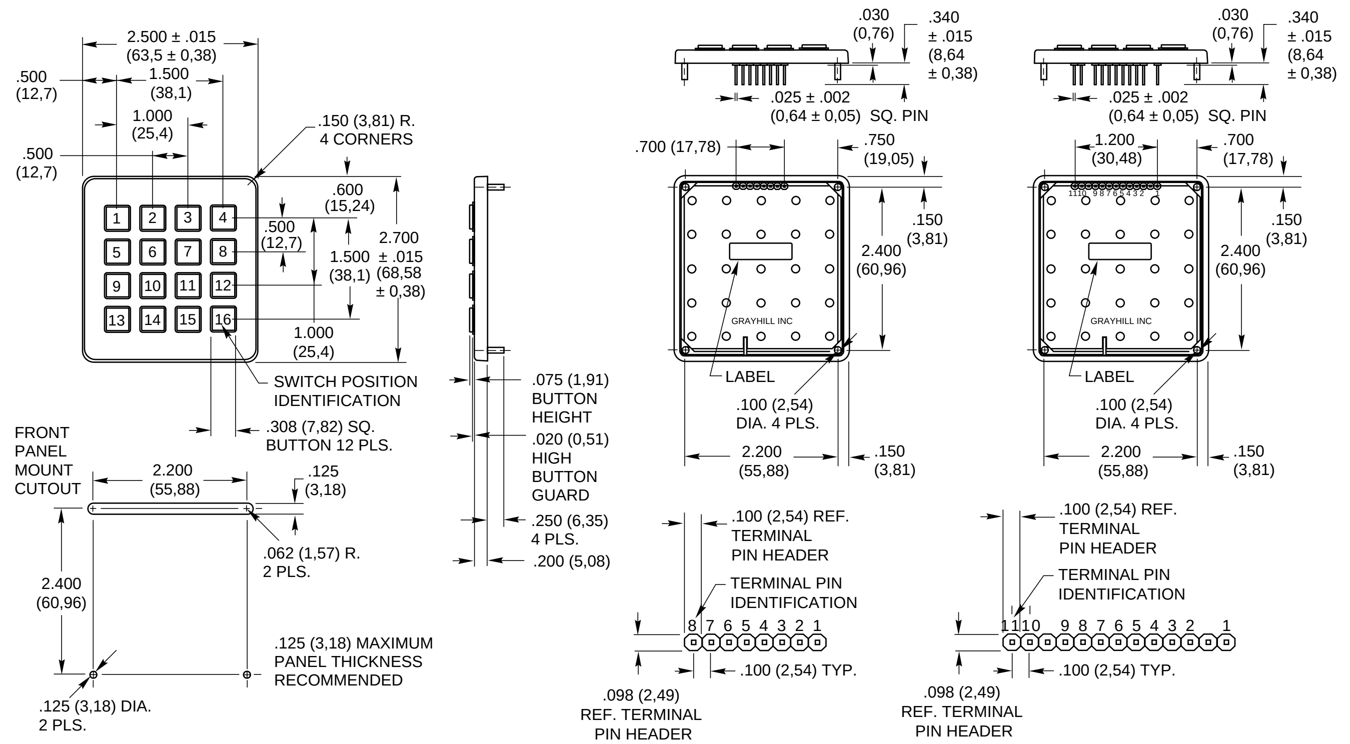
Shielded/Backlit Versions



4x4 Front Mount Keyboard

Standard Versions

Shielded/Backlit Versions

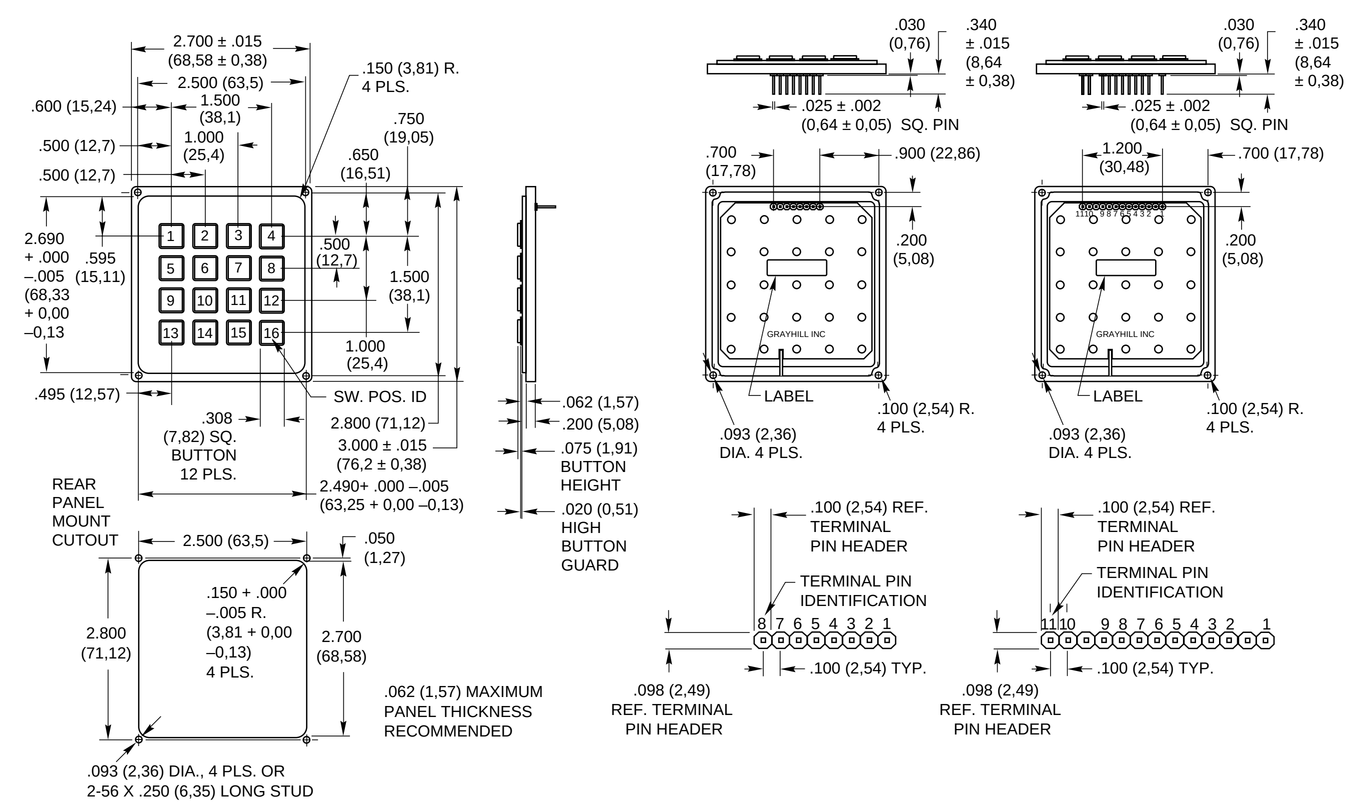


DIMENSIONS In inches (and millimeters)

4x4 Rear Mount Keyboard

Standard Versions

Shielded/Backlit Versions



CODE AND TRUTH TABLES

Dots in the chart indicate connected terminals when switch is closed.
Terminals are identified on the keyboard.

12 Button Keypads

3x4 MATRIX CODES — BUTTON LOCATION vs. TERMINAL LOCATION

3x4	Standard							Shielded/Backlit									
Button	5	6	7	1	2	3	4	6	7	8	2	3	4	5	1	9	10
1	•			•				•			•						
2		•		•					•		•						
3			•	•						•	•						
4	•				•			•				•					
5		•			•				•			•					
6			•		•					•		•					
7	•					•		•					•				
8		•				•			•				•				
9			•			•				•			•				
10	•						•	•						•			
11		•					•		•					•			
12			•				•			•				•			

Terminal 1: Shielded keypad = Shielded; Backlit keypad = NC; Shielded and backlit keypad = Shielded

Terminal 9: Shielded keypad = NC; Backlit keypad = EL Panel 1; Shielded and backlit keypad = EL Panel 1

Terminal 10: Shielded keypad = NC; Backlit keypad = EL Panel 2; Shielded and backlit keypad = EL Panel 2

16 Button Keypads

4x4 MATRIX CODES — BUTTON LOCATION vs. TERMINAL LOCATION

4x4	Standard								Shielded/Backlit										
Button	5	6	7	8	1	2	3	4	6	7	8	9	2	3	4	5	1	10	11
1	•				•				•				•						
2		•			•					•			•						
3			•		•						•		•						
4				•	•							•	•						
5	•					•			•					•					
6		•				•				•				•					
7			•			•					•			•					
8				•		•						•		•					
9	•						•		•						•				
10		•					•			•					•				
11			•				•				•				•				
12				•			•					•			•				
13	•							•	•							•			
14		•						•		•						•			
15			•					•			•					•			
16				•				•				•				•			

Terminal 1: Shielded keypad = Shielded; Backlit keypad = NC; Shielded and backlit keypad = Shielded

Terminal 10: Shielded keypad = NC; Backlit keypad = EL Panel 1; Shielded and backlit keypad = EL Panel 1

Terminal 11: Shielded keypad = NC; Backlit keypad = EL Panel 2; Shielded and backlit keypad = EL Panel 2

SPECIFICATIONS

Rating Criteria

Rating at 12 Vdc: 5 milliamps for .5 seconds
Contact Bounce: < 12 milliseconds
Contact Resistance: < 100 ohms (at stated operating force)
Voltage Breakdown: 250 Vac between components
Mechanical Operation Life: 1,000,000 operations per key
Insulation Resistance: > 10^{12} ohms @ 500 Vdc
Push Out Force Per Pin: 5 lbs.

Operating Features

Travel: .040 minimum
Operating Force: 175 ± 40 grams
Operating Temperature: -30°C to +80°C

Material and Finishes

Terminal Pin: Phosphor bronze, solder-plated
PC Board: FR-4 glass cloth epoxy
Keypad: Silicone rubber, durometer 50 ± 5
Housing: ABS, cycolac "KJW"
Housing Color: Black

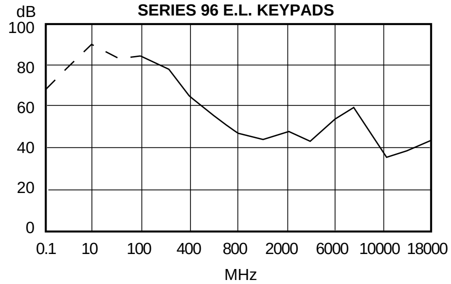
Shielding Effectiveness

Results shown are typical for a standard Grayhill Series 84S keyboard. A conductive gasket will generally increase the shielding, depending on the size and shape of the gasket and its material. Data derived for E-Field Radiation.

Test Method:
Measurements were made with the keyboard mounted to a brass plate, which in turn was mounted to a shielded enclosure containing the receiving equipment. A signal generator provided the frequency source that was radiated from the transmitting antenna to the enclosed receiving antenna. The spacing between antennas was maintained constant throughout the frequency range. The effectiveness rating is determined by establishing a reference reading without obstruction between the two antennas and determining the difference between that reading and the test setup reading.

Note:
When measured in actual equipment, shielding effectiveness is determined by many factors. This method accurately represents the shielding effectiveness of the Grayhill Series 84S under Ideal test conditions.

SHIELDING EFFECTIVENESS OF SERIES 96 E.L. KEYPADS



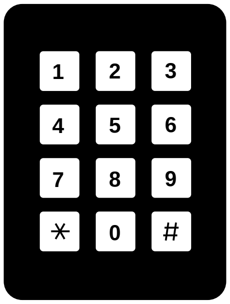
— – Represents shielding effectiveness greater than or equal to line.

Frequency M Hz	Rating in dB
0.1	≥ 66.2
10	≥ 94.8
100	90.5
400	64.2
800	42.3
2,000	40.5
6,000	33.1
10,000	34.4
18,000	37.0

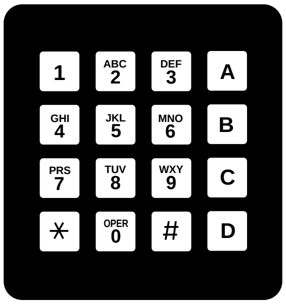
STANDARD LEGENDS

Available through Grayhill Distributors

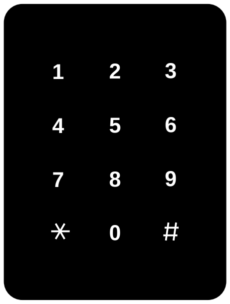
To order one of the configurations below, use the dash number shown here; select the keypad size and code, and order the part number with the appropriate legend dash number.



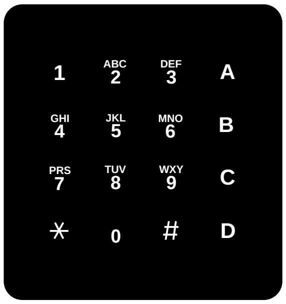
-102



-006



-152



-056

ORDERING INFORMATION

96AB2-102-FS-EL

Grayhill Series Number
Keyboard Size: A = 3x4, B = 4x4
Circuitry: B2 = Matrix (terminal pin header)

E.L. Panel Backlighting Option
EL = Backlit, Blank = Non-backlit
EMI/RFI Shielding Option
S = Shielded, Blank = Non-shielded
Mounting Option: F = Front panel mount, R = Rear panel mount
Standard Legend Choices
12 Position legends
102 = Black legends on a white button
152 = White legends on a black button
16 Position legends
006 = Black legends on a white button
056 = White legends on a black button

Available from your local Grayhill Distributor.
For prices and discounts, contact a local Sales Office, an authorized local Distributor or Grayhill.

Grayhill, Inc. • 561 Hillgrove Avenue • LaGrange, Illinois 60525-5997 • USA • Phone: 708-354-1040 • Fax: 708-354-2820 • www.grayhill.com

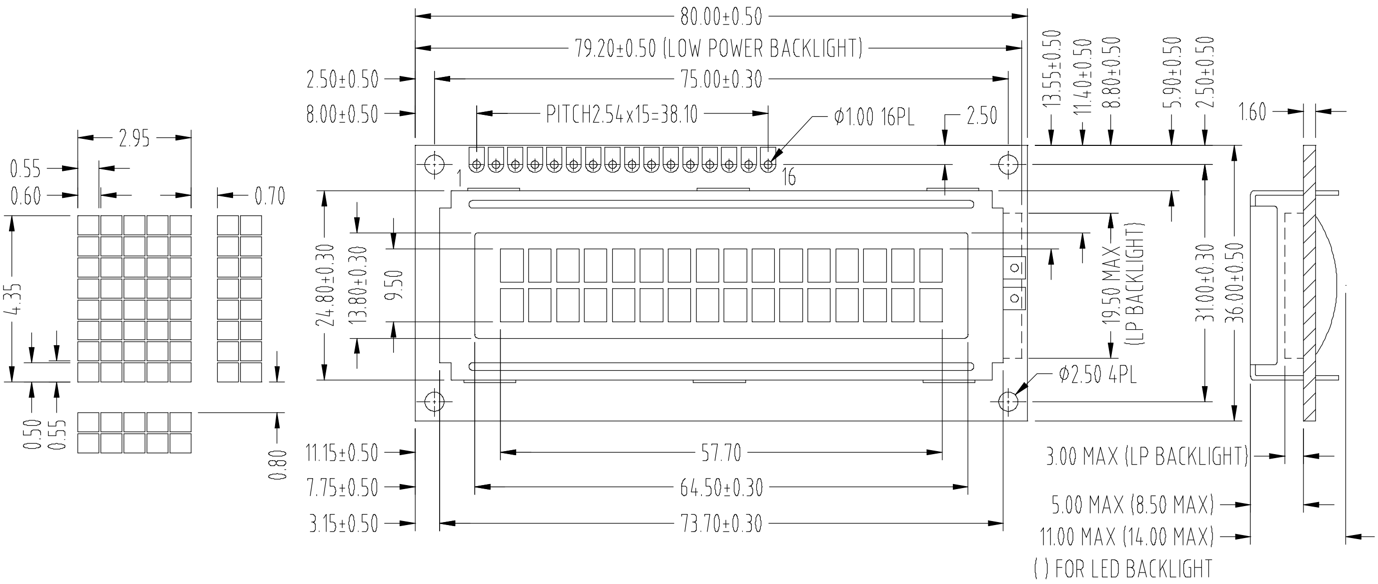
PRODUCT SPECIFICATIONS

- PHYSICAL DATA
- EXTERNAL DIMENSIONS
- BLOCK DIAGRAM
- ABSOLUTE MAXIMUM RATINGS
- ELECTRICAL CHARACTERISTICS
- OPERATING PRINCIPLES & METHODS
- DISPLAY DATA RAM ADDRESS MAP
- ELECTRO-OPTICAL CHARACTERISTICS
- INTERFACE PIN CONNECTIONS
- CIRCUIT DIAGRAM
- RELIABILITY
- QUALITY GUARANTEE
- INSPECTION CRITERIA
- PRECAUTIONS FOR USING LCD MODULES
- USING LCD MODULES

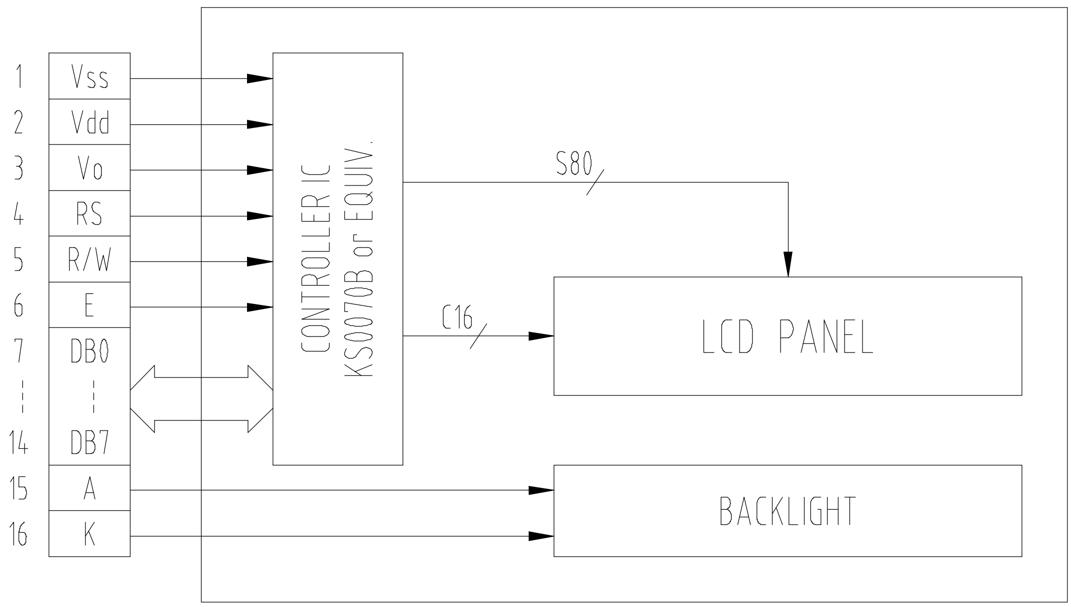
■ PHYSICAL DATA

Item	Contents	Unit
LCD type	TN / STN / FSTN	---
LCD duty	1/16	---
LCD bias	1/5	---
Viewing direction	6 / 12	o'clock
Module size (W×H×T)	80.0 × 36.0 × 11.0 MAX (14.0 MAX W/LED BACKLIGHT)	mm
Viewing area (W×H)	64.5 × 13.8	mm
Number of characters (characters×lines)	16 × 2	---
Character matrix (W×H)	5 × 8	dots
Character size (W×H)	2.95 × 4.35	mm
Dot size (W×H)	0.55 × 0.50	mm
Dot pitch (W×H)	0.60 × 0.55	mm

■ EXTERNAL DIMENSIONS



■ BLOCK DIAGRAM



■ ABSOLUTE MAXIMUM RATINGS (Ta = 25°C)

Parameter	Symbol	Min	Max	Unit
Supply voltage for logic	VDD	-0.3	7.0	V
Supply voltage for LCD	VDD - VO	-0.3	VDD+0.3	V
Input voltage	VI	-0.3	VDD+0.3	V
Normal operating temperature	TOP	0	50	°C
Normal storage temperature	TST	-10	60	°C
Wide operating / storage temperature (except FSTN)	TOP / TST	-30	80	°C
Wide operating / storage temperature (FSTN)	TOP / TST	-30	70	°C

■ ELECTRICAL CHARACTERISTICS (VDD = +5V±10% , VSS = 0V, Ta = 25°C)

◆ DC Characteristics

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Supply voltage for logic	VDD	---	4.5	5.0	5.5	V
Supply current for logic	IDD	---	---	1.38	3	mA
Operating voltage for LCD	VDD - VO	25°C	4.5	4.8	5.1	V
Input voltage ' H ' level	VIH	---	2.2	---	VDD	V
Input voltage ' L ' level	VIL	---	-0.3	---	0.6	V

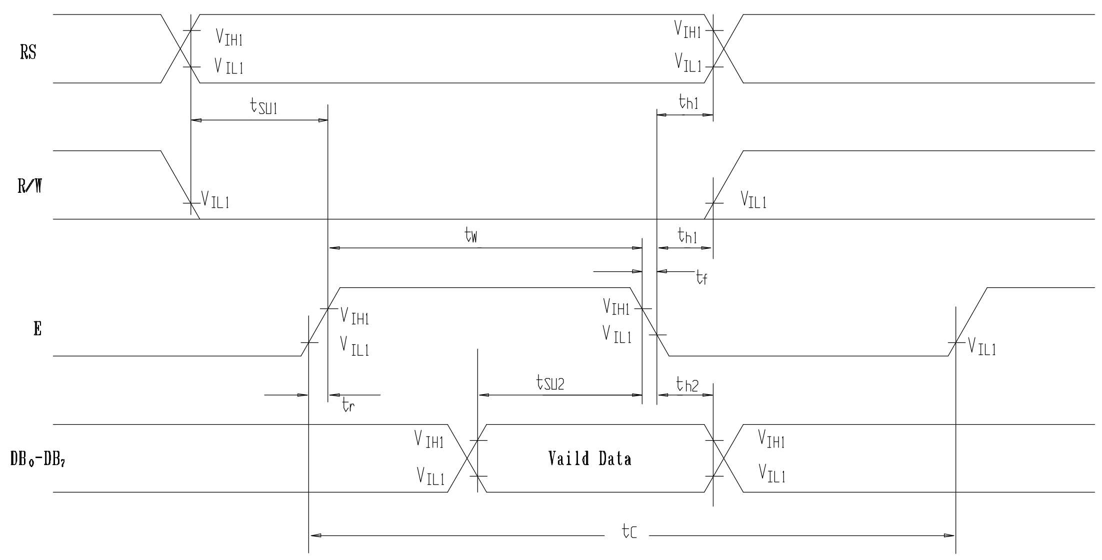
● Backlight operating information (Ta = 25°C)

	Supply voltage VF (V)			Supply current IF (mA)		
LED Backlight	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**
Light box Y/G (-2)	---	4.2	4.6	---	80	120
White (-3LP)	---	3.4	3.5	---	20	25
Blue (-4LP)	---	3.4	3.5	---	20	25
Green (-5LP)	---	3.4	3.5	---	20	25
Amber (-6LP)	---	1.8	1.9	---	20	25
	EL Enable voltage EON (VAC)			**EL frequency LF (Hz)**		
EL Backlight	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**
EL (B)	---	100	150	---	400	1000

◆ AC Characteristics

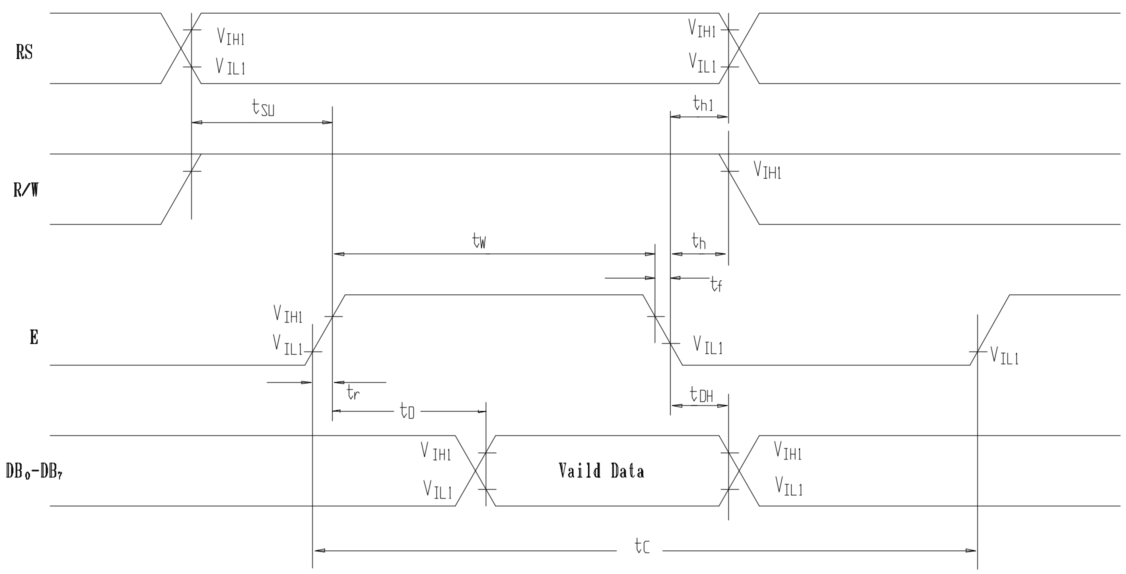
● Write mode

Characteristic	Symbol	Min.	Typ.	Max.	Unit	Test pin
E cycle time	t_C	500	---	---	ns	E
E rise time	t_r	---	---	25	ns	E
E fall time	t_f	---	---	25	ns	E
E pulse width (High, Low)	t_W	220	---	---	ns	E
R/W and RS set-up time	t_{SU1}	40	---	---	ns	R/W, RS
R/W and RS hold time	t_{h1}	10	---	---	ns	R/W, RS
Data set-up time	t_{SU2}	60	---	---	ns	DB_0 ~ DB_7
Data hold time	t_{h2}	10	---	---	ns	DB_0 ~ DB_7



● Read mode

Characteristic	Symbol	Min.	Typ.	Max.	Unit	Test pin
E cycle time	t_C	500	---	---	ns	E
E rise time	t_r	---	---	25	ns	E
E fall time	t_f	---	---	25	ns	E
E pulse width	t_W	220	---	---	ns	E
R/W and RS set-up time	t_{SU}	40	---	---	ns	R/W, RS
R/W and RS hold time	t_h	10	---	---	ns	R/W, RS
Data output delay time	t_D	---	---	120	ns	DB_0 ~ DB_7
Data hold time	t_{DH}	20	---	---	ns	DB_0 ~ DB_7



■ OPERATING PRINCIPLES & METHODS

◆ Control and Display Command

Command	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0	Execution Time (f_{osc} = 250kHz)	Remark
DISPLAY CLEAR	L	L	L	L	L	L	L	L	L	H	1.64ms	
RETURN HOME	L	L	L	L	L	L	L	L	H	X	1.64ms	Cursor move to first digit
ENTRY MODE SET	L	L	L	L	L	L	L	H	I/D	SH	42µs	• I/D : Set cursor move direction I/D: H Increase; L Decrease • SH : Specifies shift of display SH: H Display is shifted; L Display is not shifted
DISPLAY ON/OFF	L	L	L	L	L	L	H	D	C	B	42µs	• Display D: H Display on; L Display off • Cursor C: H Cursor on; L Cursor off • Blinking B: H Blinking on; L Blinking off
SHIFT	L	L	L	L	L	H	S/C	R/L	X	X	42µs	S/C: H Display shift; L Cursor move R/L: H Right shift; L Left shift
SET FUNCTION	L	L	L	L	H	DL	N	F	X	X	42µs	DL: H 8 bits interface; L 4 bits interface N: H 2 line display; L 1 line display F: H 5 X 10 dots; L 5 X 7 dots
SET CG RAM ADDRESS	L	L	L	H	CG RAM address (corresponds to cursor address)						42µs	CG RAM Data is sent and received after this setting
SET DD RAM ADDRESS	L	L	H	DD RAM address							42µs	DD RAM Data is sent and received after this setting
READ BUSY FLAG & ADDRESS	L	H	BF	Address Counter used for both DD & CG RAM address							0µs	BF: H Busy; L Ready – Reads BF indication internal operating is being performed – Reads address counter contents
WRITE DATA	H	L	Write Data								46µs	Write data into DD or CG RAM
READ DATA	H	H	Read Data								46µs	Read data from DD or CG RAM

X : Don’t care

◆ Initializing by Internal Reset Circuit

The KS0070B automatically initializes (resets) when the power is on using the internal reset circuit. The following instruction are executed in initialization. The busy flag is kept in busy state (BF=1) until initialization ends. The busy state is 10ms after VDD rises to 4.5V.

(1) Display Clear

(2) Function Set

- DL = 1 : 8-bit interface data
- N = 0 : 1-line display
- F = 0 : 5x7-dot character font

(3) Display On/Off Control

- D = 0 : Display Off
- C = 0 : Cursor Off
- B = 0 : Blink Off

(4) Entry Mode Set

- I/D = 1 : +1 (Increment)
- S = 0 : No Shift

◆ Initializing by Instruction

Power On

↓

Wait for more than 15ms after VDD rises to 4.5V

↓

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	1	*	*	*	*

BF cannot be checked before this instruction.
Function Set

↓

Wait for more than 4.1ms

↓

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	1	*	*	*	*

BF cannot be checked before this instruction.
Function Set

↓

Wait for more than 100µs

↓

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	1	*	*	*	*

BF cannot be checked before this instruction.
Function Set

↓

BF can be checked after following instruction.
When BF is not checked, the waiting time between instructions is longer than execution instruction time.

↓

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
0	0	0	0	1	1	N	F	*	*	Function Set (Specify the number of display lines and character font.) The number of display lines and character font cannot be changed afterwards.
0	0	0	0	0	0	1	0	0	0	Display Off
0	0	0	0	0	0	0	0	0	1	Display Clear
0	0	0	0	0	0	0	1	I/D	S	Entry Mode Set

↓

Initialization ends

◆ Standard Character Pattern

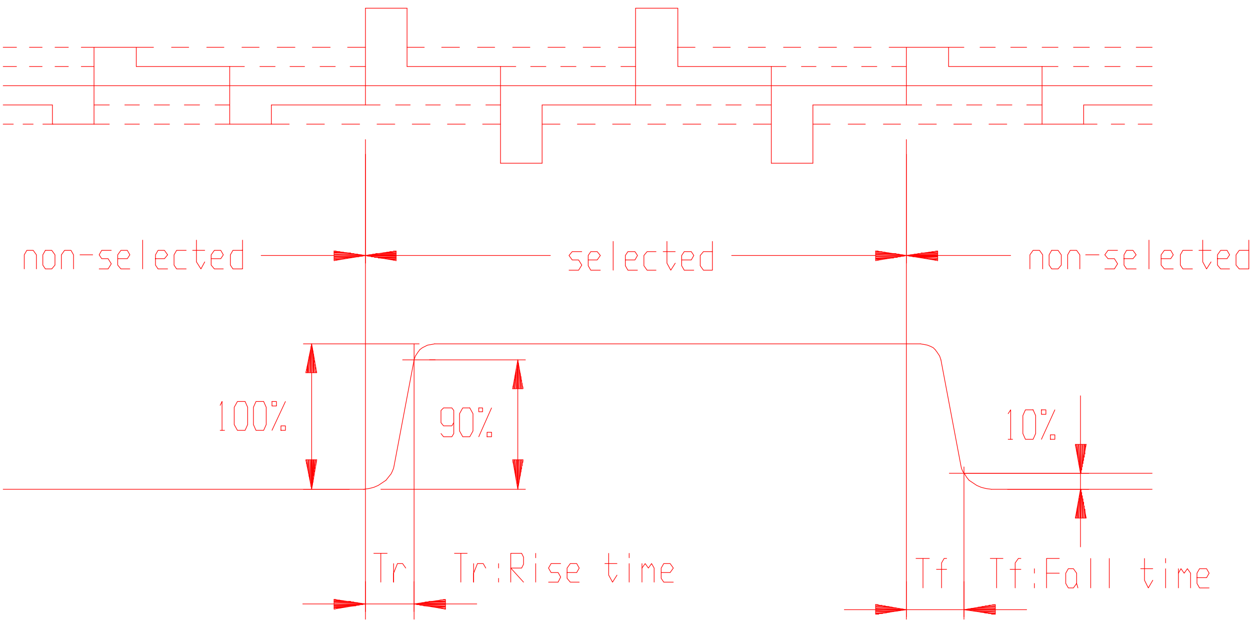
■ DISPLAY DATA RAM ADDRESS MAP

Characters	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
First line	00H	01H	02H	03H	04H	05H	06H	07H	08H	09H	0AH	0BH	0CH	0DH	0EH	0FH
Second line	40H	41H	42H	43H	44H	45H	46H	47H	48H	49H	4AH	4BH	4CH	4DH	4EH	4FH

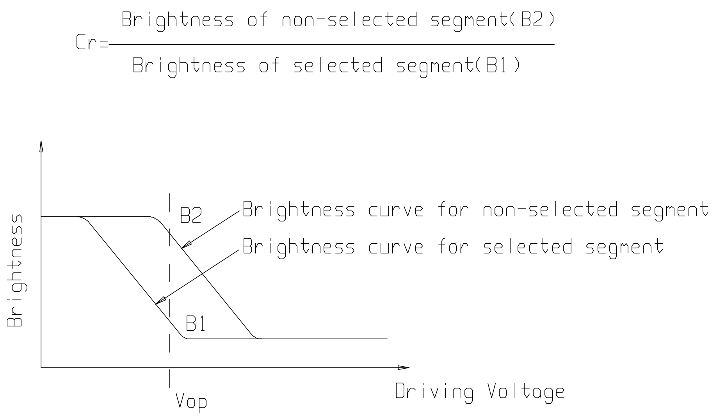
■ ELECTRO-OPTICAL CHARACTERISTICS (V_{OP} = 5.0V, Ta = 25°C, Transflective version)

LCD mode	Typ response time Tr (ms)		Typ response time Tf (ms)		Typ contrast ratio Cr	Typ viewing angle q (deg)			
	Normal temp	Wide temp	Normal temp	Wide temp		Æ = 0°	Æ = 90°	Æ = 180°	Æ = 270°
TN (A)	275	147	61	57	28	20	40	5	40
STN Y/G (B)					30	60	48	57	47
STN Blue (C)					6	52	25	33	33
STN Grey (D)					12	60	37	55	38
FSTN (F)					38	65	49	58	48
FSTN Negative (G)					18	53	25	34	33

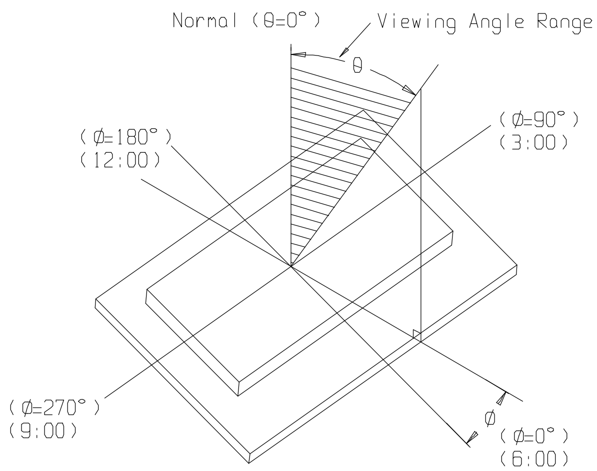
Note1: Definition of response time.



Note2: Definition of contrast ratio 'Cr' .



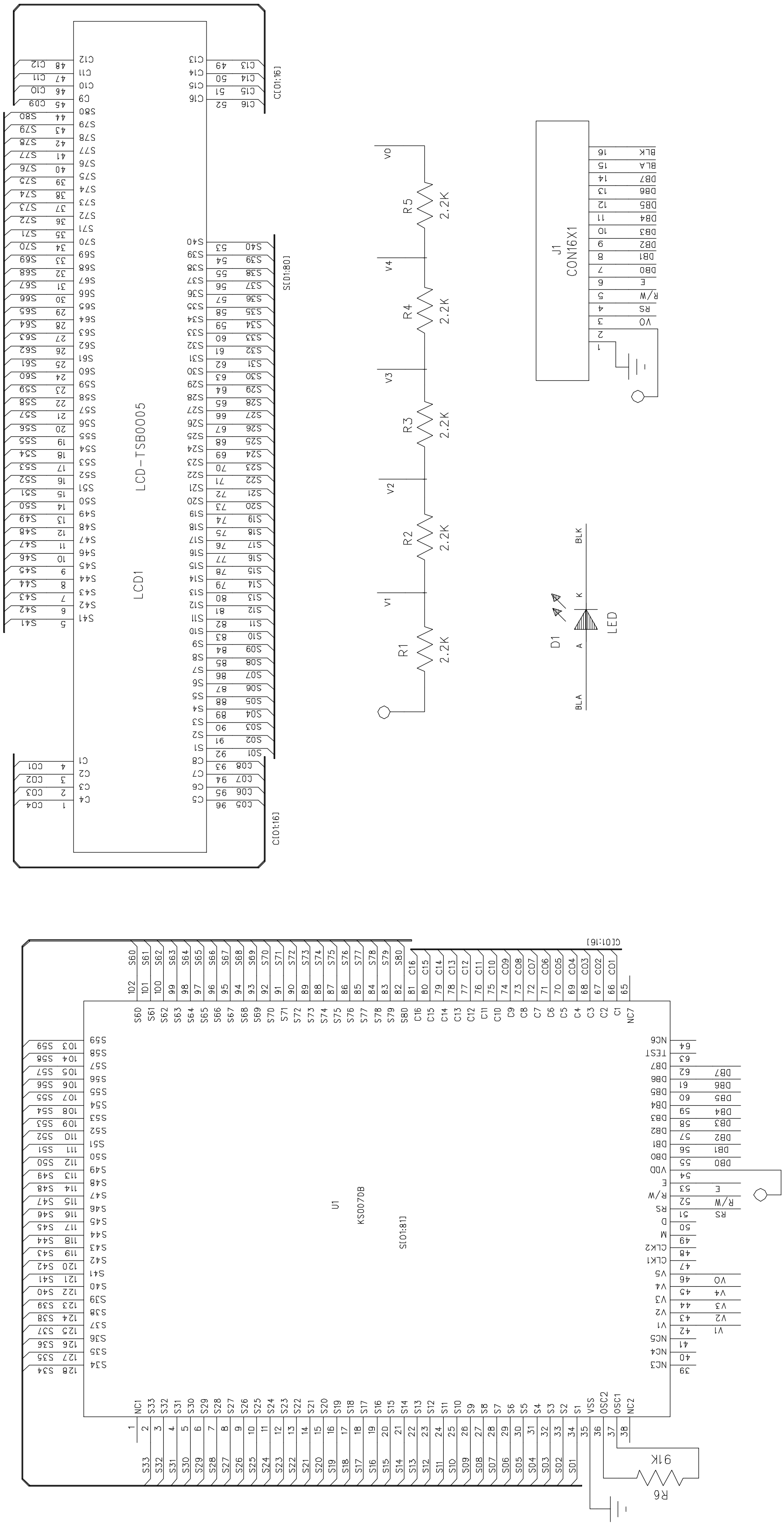
Note3: Definition of viewing angle range 'θ'.



■ INTERFACE PIN CONNECTIONS

Pin NO.	Symbol	Level	Description
1	VSS	0V	Ground
2	VDD	5.0V	Supply voltage for logic
3	VO	---	Input voltage for LCD
4	RS	H/L	H : Data, L : Instruction code
5	R/W	H/L	H : Read mode, L : Write mode
6	E	H, H → L	Chip enable signal
7	DB0	H/L	Data bit 0
8	DB1	H/L	Data bit 1
9	DB2	H/L	Data bit 2
10	DB3	H/L	Data bit 3
11	DB4	H/L	Data bit 4
12	DB5	H/L	Data bit 5
13	DB6	H/L	Data bit 6
14	DB7	H/L	Data bit 7
15	A	---	Backlight anode
16	K	---	Backlight cathode

■ CIRCUIT DIAGRAM



■ RELIABILITY

◆ Content of Reliability Test

No.	Test Item	Content of Test	Test Condition	Applicable Standard
Environmental Test				
1	High temperature storage	Endurance test applying the high storage temperature for a long time.	60 °C 200 hrs	------
2	Low temperature storage	Endurance test applying the low storage temperature for a long time.	-10 °C 200 hrs	------
3	High temperature operation	Endurance test applying the electric stress (Voltage & Current) and the thermal stress to the element for a long time.	50 °C 200 hrs	------
4	Low temperature operation	Endurance test applying the electric stress under low temperature for a long time.	0 °C 200 hrs	------
5	High temperature / Humidity storage	Endurance test applying the high temperature and high humidity storage for a long time.	60 °C , 90 %RH 96 hrs	MIL-202E-103B JIS-C5023
6	High temperature / Humidity operation	Endurance test applying the electric stress (Voltage & Current) and temperature / humidity stress to the element for a long time.	40 °C , 90 %RH 96 hrs	MIL-202E-103B JIS-C5023
7	Temperature cycle	Endurance test applying the low and high temperature cycle. -10°C 30min ⇌ 25°C 5min. ⇌ 60°C 30min ←— 1 cycle —→	-10°C / 60°C 10 cycles	------
Mechanical Test				
8	Vibration test	Endurance test applying the vibration during transportation and using.	10~22Hz → 1.5mmp-p 22~500Hz → 1.5G Total 0.5hrs	MIL-202E-201A JIS-C5025 JIS-C7022-A-10
9	Shock test	Constructional and mechanical endurance test applying the shock during transportation.	50G half sign wave 1l msedc 3 times of each direction	MIL-202E-213B
10	Atmospheric pressure test	Endurance test applying the atmospheric pressure during transportation by air.	115 mbar 40 hrs	MIL-202E-105C
Others				
11	Static electricity test	Endurance test applying the electric stress to the terminal.	VS=800V , RS=1.5 kΩ CS=100 pF 1 time	MIL-883B-3015.1

*** Supply voltage for logic system = 5V. Supply voltage for LCD system = Operating voltage at 25°C.

◆ Failure Judgement Criterion

Criterion Item	Test Item No.											Failure Judgment Criterion
	1	2	3	4	5	6	7	8	9	10	11	
Basic specification												Out of the Basic Specification
Electrical characteristic												Out of the DC and AC Characterstic
Mechanical characterstic												Out of the Mechanical Specification Color change : Out of Limit Apperance Specification
Optical characterstic												Out of the Apperance Standard

■ QUALITY GUARANTEE

◆ Acceptable Quality Level

Each lot should satisfy the quality level defined as follows.

- Inspection method : MIL-STD-105E LEVEL II Normal one time sampling
- AQL

Partition	AQL	Definition
A: Major	0.4%	Functional defective as product
B: Minor	1.5%	Satisfy all functions as product but not satisfy cosmetic standard

◆ Definition of 'LOT'

One lot means the delivery quantity to customer at one time.

◆ Conditions of Cosmetic Inspection

● Environmental condition

The inspection should be performed at the 1m of height from the LCD module under 2 pieces of 40W white fluorescent lamps (Normal temperature 20~25°C and normal humidity 60±15%RH).

● Inspection method

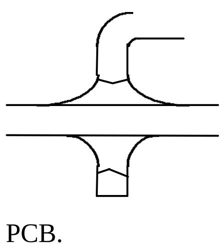
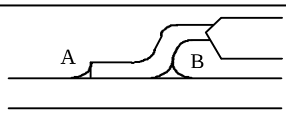
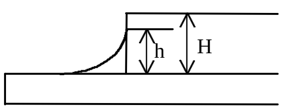
The visual check should be performed vertically at more than 30cm distance from the LCD panel.

● Driving voltage

The V_O value which the most optimal contrast can be obtained near the specified V_O in the specification. (Within ±0.5V of the typical value at 25°C.).

■ INSPECTION CRITERIA

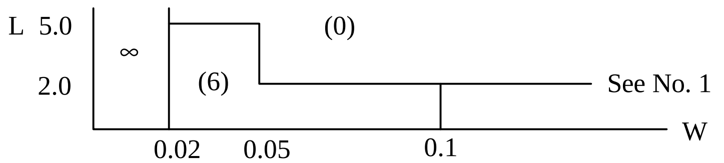
◆ Module Cosmetic Criteria

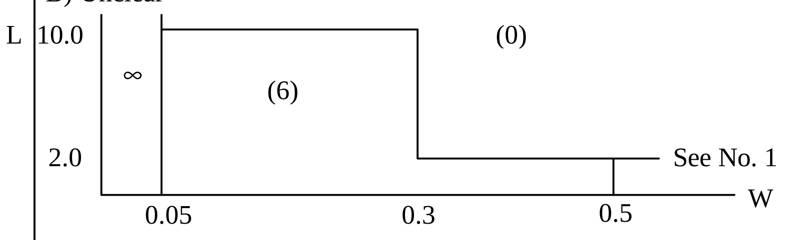
No.	Item	Judgement Criterion	Partition
1	Difference in Spec.	None allowed	Major
2	Pattern peeling	No substrate pattern peeling and floating	Major
3	Soldering defects	No soldering missing No soldering bridge No cold soldering	Major Major Minor
4	Resist flaw on substrate	Invisible copper foil (∅0.5mm or more) on substrate pattern	Minor
5	Accretion of metallic Foreign matter	No soldering dust No accretion of metallic foreign matters (Not exceed ∅0.2mm)	Minor Minor
6	Stain	No stain to spoil cosmetic badly	Minor
7	Plate discoloring	No plate fading, rusting and discoloring	Minor
8	Solder amount 1. Lead parts	a. Soldering side of PCB Solder to form a 'Filet' all around the lead. Solder should not hide the lead form perfectly. (too much) b. Components side (In case of 'Through Hole PCB')  Solder to reach the Components side of PCB.	Minor
	2. Flat packages	Either 'toe' (A) or 'heal' (B) of the lead to be covered by 'Filet'.  Lead form to be assume over solder.	Minor
	3. Chips	(3/2) H ≥ h ≥ (1/2) H 	Minor

◆ Screen Cosmetic Criteria (Non-Operating)

<table>
<tr><th>No.</th><th>Defect</th><th>Judgement Criterion</th><th>Partition</th></tr>
<tr><td>1</td><td>Spots</td><td>In accordance with Screen Cosmetic Criteria (Operating) No.1.</td><td>Minor</td></tr>
<tr><td>2</td><td>Lines</td><td>In accordance with Screen Cosmetic Criteria (Operating) No.2.</td><td>Minor</td></tr>
<tr><td>3</td><td>Bubbles in polarizer</td><td><table><tr><th>Size : d mm</th><th>Acceptable Qty in active area</th></tr><tr><td>d ≤ 0.3</td><td>Disregard</td></tr><tr><td>0.3 < d ≤ 1.0</td><td>3</td></tr><tr><td>1.0 < d ≤ 1.5</td><td>1</td></tr><tr><td>1.5 < d</td><td>0</td></tr></table></td><td>Minor</td></tr>
<tr><td>4</td><td>Scratch</td><td>In accordance with spots and lines operating cosmetic criteria. When the light reflects on the panel surface, the scratches are not to be remarkable.</td><td>Minor</td></tr>
<tr><td>5</td><td>Allowable density</td><td>Above defects should be separated more than 30mm each other.</td><td>Minor</td></tr>
<tr><td>6</td><td>Coloration</td><td>Not to be noticeable coloration in the viewing area of the LCD panels.
Back-lit type should be judged with back-lit on state only.</td><td>Minor</td></tr>
<tr><td>7</td><td>Contamination</td><td>Not to be noticeable.</td><td>Minor</td></tr>
</table>

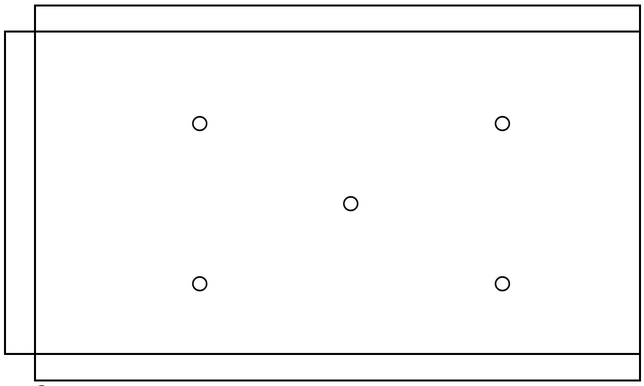
◆ Screen Cosmetic Criteria (Operating)

<table>
<tr><th>No.</th><th>Defect</th><th>Judgement Criterion</th><th>Partition</th></tr>
<tr><td>1</td><td>Spots</td><td>A) Clear<table><tr><th>Size : d mm</th><th>Acceptable Qty in active area</th></tr><tr><td>d ≤ 0.1</td><td>Disregard</td></tr><tr><td>0.1 < d ≤ 0.2</td><td>6</td></tr><tr><td>0.2 < d ≤ 0.3</td><td>2</td></tr><tr><td>0.3 < d</td><td>0</td></tr></table>Note : Including pin holes and defective dots which must be within one pixel size.
B) Unclear<table><tr><th>Size : d mm</th><th>Acceptable Qty in active area</th></tr><tr><td>d ≤ 0.2</td><td>Disregard</td></tr><tr><td>0.2 < d ≤ 0.5</td><td>6</td></tr><tr><td>0.5 < d ≤ 0.7</td><td>2</td></tr><tr><td>0.7 < d</td><td>0</td></tr></table></td><td>Minor</td></tr>
<tr><td>2</td><td>Lines</td><td>A) Clear


Note : ( ) - Acceptable Qty in active area
L - Length (mm)
W - Width (mm)
∞ - Disregard
B) Unclear

</td><td>Minor</td></tr>
</table>

'Clear' = The shade and size are not changed by V_O.
'Unclear' = The shade and size are changed by V_O.

◆ Screen Cosmetic Criteria (Operating) (Continued)

No.	Defect	Judgement Criterion	Partition
3	Rubbing line	Not to be noticeable.	
4	Allowable density	Above defects should be separated more than 10mm each other.	Minor
5	Rainbow	Not to be noticeable.	Minor
6	Dot size	To be 95% ~ 105% of the dot size (Typ.) in drawing. Partial defects of each dot (ex. pin-hole) should be treated as 'spot'. (see *Screen Cosmetic Criteria (Operating) No.1*)	Minor
7	Uneven brightness (only back-lit type module)	Uneven brightness must be BMAX / BMIN ≤ 2 - BMAX : Max. value by measure in 5 points - BMIN : Min. value by measure in 5 points Divide active area into 4 vertically and horizontally. Measure 5 points shown in the following figure.  ○ : Measuring points	Minor

Note :

(1) Size : d = (long length + short length) / 2

(2) The limit samples for each item have priority.

(3) Complexed defects are defined item by item, but if the number of defects are defined in above table, the total number should not exceed 10.

(4) In case of 'concentration', even the spots or the lines of 'disregarded' size should not allowed. Following three situations should be treated as 'concentration'.

- 7 or over defects in circle of ∅5mm.
- 10 or over defects in circle of ∅10mm.
- 20 or over defects in circle of ∅20mm.

■ PRECAUTIONS FOR USING LCD MODULES

◆ Handing Precautions

(1) The display panel is made of glass. Do not subject it to a mechanical shock by dropping it or impact.

(2) If the display panel is damaged and the liquid crystal substance leaks out, be sure not to get any in your mouth. If the substance contacts your skin or clothes, wash it off using soap and water.

(3) Do not apply excessive force to the display surface or the adjoining areas since this may cause the color tone to vary.

(4) The polarizer covering the display surface of the LCD module is soft and easily scratched. Handle this polarizer carefully.

(5) If the display surface becomes contaminated, breathe on the surface and gently wipe it with a soft dry cloth. If it is heavily contaminated, moisten cloth with one of the following solvents :

- Isopropyl alcohol
- Ethyl alcohol

(6) Solvents other than those above-mentioned may damage the polarizer. Especially, do not use the following.

- Water
- Ketone
- Aromatic solvents

(7) Exercise care to minimize corrosion of the electrode. Corrosion of the electrodes is accelerated by water droplets, moisture condensation or a current flow in a high-humidity environment.

(8) Install the LCD Module by using the mounting holes. When mounting the LCD module make sure it is free of twisting, warping and distortion. In particular, do not forcibly pull or bend the I/O cable or the backlight cable.

(9) Do not attempt to disassemble or process the LCD module.

(10) NC terminal should be open. Do not connect anything.

(11) If the logic circuit power is off, do not apply the input signals.

(12) To prevent destruction of the elements by static electricity, be careful to maintain an optimum work environment.

- Be sure to ground the body when handling the LCD modules.
- Tools required for assembling, such as soldering irons, must be properly grounded.
- To reduce the amount of static electricity generated, do not conduct assembling and other work under dry conditions.
- The LCD module is coated with a film to protect the display surface. Exercise care when peeling off this protective film since static electricity may be generated.

◆ Storage Precautions

When storing the LCD modules, avoid exposure to direct sunlight or to the light of fluorescent lamps. Keep the modules in bags (avoid high temperature / high humidity and low temperatures below 0°C). Whenever possible, the LCD modules should be stored in the same conditions in which they were shipped from our company.

◆ Others

Liquid crystals solidify under low temperature (below the storage temperature range) leading to defective orientation or the generation of air bubbles (black or white). Air bubbles may also be generated if the module is subject to a low temperature.

If the LCD modules have been operating for a long time showing the same display patterns, the display patterns may remain on the screen as ghost images and a slight contrast irregularity may also appear. A normal operating status can be regained by suspending use for some time. It should be noted that this phenomenon does not adversely affect performance reliability.

To minimize the performance degradation of the LCD modules resulting from destruction caused by static electricity etc., exercise care to avoid holding the following sections when handling the modules.

- Exposed area of the printed circuit board.
- Terminal electrode sections.

■ USING LCD MODULES

◆ Liquid Crystal Display Modules

LCD is composed of glass and polarizer. Pay attention to the following items when handling.

(1) Please keep the temperature within specified range for use and storage. Polarization degradation, bubble generation or polarizer peel-off may occur with high temperature and high humidity.

(2) Do not touch, push or rub the exposed polarizers with anything harder than an HB pencil lead (glass, tweezers, etc.).

(3) N-hexane is recommended for cleaning the adhesives used to attach front/rear polarizers and reflectors made of organic substances which will be damaged by chemicals such as acetone, toluene, ethanol and isopropylalcohol.

(4) When the display surface becomes dusty, wipe gently with absorbent cotton or other soft material like chamois soaked in petroleum benzin. Do not scrub hard to avoid damaging the display surface.

(5) Wipe off saliva or water drops immediately, contact with water over a long period of time may cause deformation or color fading.

(6) Avoid contacting oil and fats.

(7) Condensation on the surface and contact with terminals due to cold will damage, stain or dirty the polarizers. After products are tested at low temperature they must be warmed up in a container before coming is contacting with room temperature air.

(8) Do not put or attach anything on the display area to avoid leaving marks on.

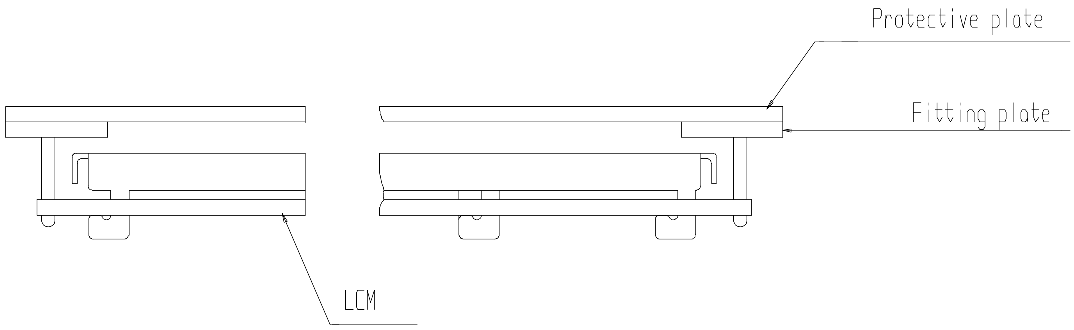
(9) Do not touch the display with bare hands. This will stain the display area and degradate insulation between terminals (some cosmetics are determinated to the polarizers).

(10) As glass is fragile. It tends to become or chipped during handling especially on the edges. Please avoid dropping or jarring.

◆ Installing LCD Modules

The hole in the printed circuit board is used to fix LCM as shown in the picture below. Attend to the following items when installing the LCM.

(1) Cover the surface with a transparent protective plate to protect the polarizer and LC cell.



(2) When assembling the LCM into other equipment, the spacer to the bit between the LCM and the fitting plate should have enough height to avoid causing stress to the module surface, refer to the individual specifications for measurements. The measurement tolerance should be ±0.1mm.

◆ Precaution for Handing LCD Modules

Since LCM has been assembled and adjusted with a high degree of precision, avoid applying excessive shocks to the module or making any alterations or modifications to it.

(1) Do not alter, modify or change the the shape of the tab on the metal frame.

(2) Do not make extra holes on the printed circuit board, modify its shape or change the positions of components to be attached.

(3) Do not damage or modify the pattern writing on the printed circuit board.

(4) Absolutely do not modify the zebra rubber strip (conductive rubber) or heat seal connector.

(5) Except for soldering the interface, do not make any alterations or modifications with a soldering iron.

(6) Do not drop, bend or twist LCM.

◆ Electro-Static Discharge Control

Since this module uses a CMOS LSI, the same careful attention should be paid to electrostatic discharge as for an ordinary CMOS IC.

(1) Make certain that you are grounded when handing LCM.

(2) Before remove LCM from its packing case or incorporating it into a set, be sure the module and your body have the same electric potential.

(3) When soldering the terminal of LCM, make certain the AC power source for the soldering iron does not leak.

(4) When using an electric screwdriver to attach LCM, the screwdriver should be of ground potentiality to minimize as much as possible any transmission of electromagnetic waves produced sparks coming from the commutator of the motor.

(5) As far as possible make the electric potential of your work clothes and that of the work bench the ground potential.

(6) To reduce the generation of static electricity be careful that the air in the work is not too dried. A relative humidity of 50%-60% is recommended.

◆ Precaution for soldering to the LCM

(1) Observe the following when soldering lead wire, connector cable and etc. to the LCM.

- Soldering iron temperature : 280°C ± 10°C.
- Soldering time : 3-4 sec.
- Solder : eutectic solder.

If soldering flux is used, be sure to remove any remaining flux after finishing to soldering operation. (This does not apply in the case of a non-halogen type of flux.) It is recommended that you protect the LCD surface with a cover during soldering to prevent any damage dur to flux spatters.

(2) When soldering the electroluminescent panel and PC board, the panel and board should not be detached more than three times. This maximum number is determined by the temperature and time conditions mentioned above, though there may be some variance depending on the temperature of the soldering iron.

(3) When remove the electoluminescent panel from the PC board, be sure the solder has completely melted, the soldered pad on the PC board could be damaged.

◆ Precautions for Operation

(1) Viewing angle varies with the change of liquid crystal driving voltage (V_O). Adjust V_O to show the best contrast.

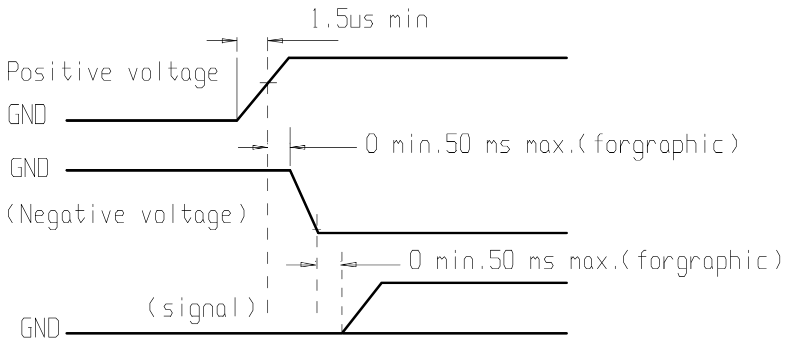
(2) Driving the LCD in the voltage above the limit shortens its life.

(3) Response time is greatly delayed at temperature below the operating temperature range. However, this does not mean the LCD will be out of the order. It will recover when it returns to the specified temperature range.

(4) If the display area is pushed hard during operation, the display will become abnormal. However, it will return to normal if it is turned off and then back on.

(5) Condensation on terminals can cause an electrochemical reaction disrupting the terminal circuit. Therefore, it must be used under the relative condition of 40°C , 50% RH.

(6) When turning the power on, input each signal after the positive/negative voltage becomes stable.



◆ Storage

When storing LCDs as spares for some years, the following precaution are necessary.

(1) Store them in a sealed polyethylene bag. If properly sealed, there is no need for dessicant.

(2) Store them in a dark place. Do not expose to sunlight or fluorescent light, keep the temperature between 0°C and 35°C.

(3) The polarizer surface should not come in contact with any other objects. (We advise you to store them in the container in which they were shipped.)

(4) Environmental conditions :

- Do not leave them for more than 168hrs. at 60°C.
- Should not be left for more than 48hrs. at -20°C.

◆ Safety

(1) It is recommended to crush damaged or unnecessary LCDs into pieces and wash them off with solvents such as acetone and ethanol, which should later be burned.

(2) If any liquid leakes out of a damaged glass cell and comes in contact with the hands, wash off thoroughly with soap and water.

◆ Limited Warranty

Unless agreed between DISPLAYTECH and customer, DISPLAYTECH will replace or repair any of its LCD modules which are found to be functionally defective when inspected in accordance with DISPLAYTECH LCD acceptance standards (copies available upon request) for a period of one year from date of shipments. Cosmetic/visual defects must be returned to DISPLAYTECH within 90 days of shipment. Confirmation of such date shall be based on freight documents. The warranty liability of DISPLAYTECH limited to repair and/or replacement on the terms set forth above. DISPLAYTECH will not be responsible for any subsequent or consequential events.

◆ Return LCM under warranty

No warranty can be granted if the precautions stated above have been disregarded. The typical examples of violations are :

- Broken LCD glass.
- PCB eyelet's damaged or modified.
- PCB conductors damaged.
- Circuit modified in any way, including addition of components.
- PCB tampered with by grinding, engraving or painting varnish.
- soldering to or modifying the bezel in any manner.

Module repairs will be invoiced to the customer upon mutual agreement. Modules must be returned with sufficient description of the failures or defects. Any connectors or cable installed by the customer must be removed completely without damaging the PCB eyelet's, conductors and terminals.

DALLAS SEMICONDUCTOR

www.maxim-ic.com

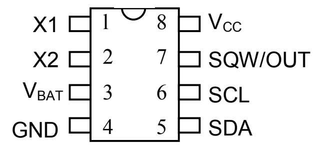
FEATURES

- Real-time clock (RTC) counts seconds, minutes, hours, date of the month, month, day of the week, and year with leap-year compensation valid up to 2100
- 56-byte, battery-backed, nonvolatile (NV) RAM for data storage
- Two-wire serial interface
- Programmable squarewave output signal
- Automatic power-fail detect and switch circuitry
- Consumes less than 500nA in battery backup mode with oscillator running
- Optional industrial temperature range: -40°C to +85°C
- Available in 8-pin DIP or SOIC
- Underwriters Laboratory (UL) recognized

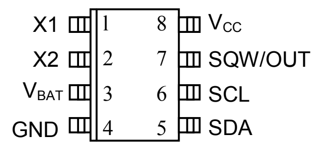
ORDERING INFORMATION

DS1307	8-Pin DIP (300-mil)
DS1307Z	8-Pin SOIC (150-mil)
DS1307N	8-Pin DIP (Industrial)
DS1307ZN	8-Pin SOIC (Industrial)

PIN ASSIGNMENT



DS1307 8-Pin DIP (300-mil)



DS1307 8-Pin SOIC (150-mil)

PIN DESCRIPTION

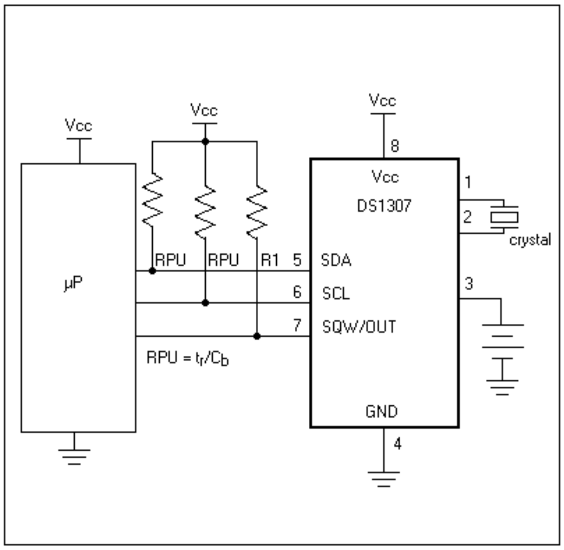
V_{CC}	- Primary Power Supply
X1, X2	- 32.768kHz Crystal Connection
V_{BAT}	- +3V Battery Input
GND	- Ground
SDA	- Serial Data
SCL	- Serial Clock
SQW/OUT	- Square Wave/Output Driver

DESCRIPTION

The DS1307 Serial Real-Time Clock is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially via a 2-wire, bi-directional bus. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 has a built-in power sense circuit that detects power failures and automatically switches to the battery supply.

 100101

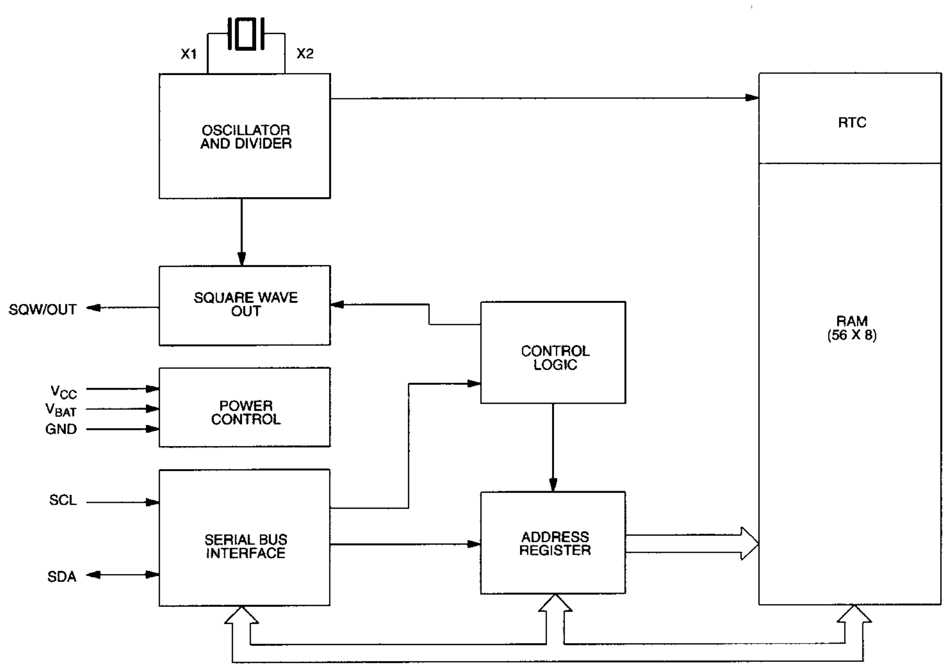
TYPICAL OPERATING CIRCUIT



OPERATION

The DS1307 operates as a slave device on the serial bus. Access is obtained by implementing a START condition and providing a device identification code followed by a register address. Subsequent registers can be accessed sequentially until a STOP condition is executed. When V_{CC} falls below 1.25 x V_{BAT} the device terminates an access in progress and resets the device address counter. Inputs to the device will not be recognized at this time to prevent erroneous data from being written to the device from an out of tolerance system. When V_{CC} falls below V_{BAT} the device switches into a low-current battery backup mode. Upon power-up, the device switches from battery to V_{CC} when V_{CC} is greater than V_{BAT} + 0.2V and recognizes inputs when V_{CC} is greater than 1.25 x V_{BAT}. The block diagram in Figure 1 shows the main elements of the serial RTC.

DS1307 BLOCK DIAGRAM Figure 1



SIGNAL DESCRIPTIONS

V_{CC}, GND – DC power is provided to the device on these pins. V_{CC} is the +5V input. When 5V is applied within normal limits, the device is fully accessible and data can be written and read. When a 3V battery is connected to the device and V_{CC} is below 1.25 x V_{BAT}, reads and writes are inhibited. However, the timekeeping function continues unaffected by the lower input voltage. As V_{CC} falls below V_{BAT} the RAM and timekeeper are switched over to the external power supply (nominal 3.0V DC) at V_{BAT}.

V_{BAT} – Battery input for any standard 3V lithium cell or other energy source. Battery voltage must be held between 2.0V and 3.5V for proper operation. The nominal write protect trip point voltage at which access to the RTC and user RAM is denied is set by the internal circuitry as 1.25 x V_{BAT} nominal. A lithium battery with 48mAhr or greater will back up the DS1307 for more than 10 years in the absence of power at 25ºC. UL recognized to ensure against reverse charging current when used in conjunction with a lithium battery.

See "Conditions of Acceptability" at http://www.maxim-ic.com/TechSupport/QA/ntrl.htm**.**

SCL (Serial Clock Input) – SCL is used to synchronize data movement on the serial interface.

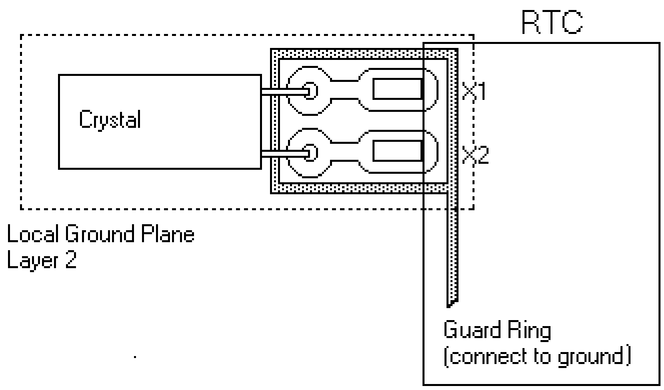
SDA (Serial Data Input/Output) – SDA is the input/output pin for the 2-wire serial interface. The SDA pin is open drain which requires an external pullup resistor.

SQW/OUT (Square Wave/Output Driver) – When enabled, the SQWE bit set to 1, the SQW/OUT pin outputs one of four square wave frequencies (1Hz, 4kHz, 8kHz, 32kHz). The SQW/OUT pin is open drain and requires an external pull-up resistor. SQW/OUT will operate with either Vcc or Vbat applied.

X1, X2 – Connections for a standard 32.768kHz quartz crystal. The internal oscillator circuitry is designed for operation with a crystal having a specified load capacitance (CL) of 12.5pF.

For more information on crystal selection and crystal layout considerations, please consult Application Note 58, "Crystal Considerations with Dallas Real-Time Clocks." The DS1307 can also be driven by an external 32.768kHz oscillator. In this configuration, the X1 pin is connected to the external oscillator signal and the X2 pin is floated.

RECOMMENDED LAYOUT FOR CRYSTAL



CLOCK ACCURACY

The accuracy of the clock is dependent upon the accuracy of the crystal and the accuracy of the match between the capacitive load of the oscillator circuit and the capacitive load for which the crystal was trimmed. Additional error will be added by crystal frequency drift caused by temperature shifts. External circuit noise coupled into the oscillator circuit may result in the clock running fast. See Application Note 58, “Crystal Considerations with Dallas Real-Time Clocks” for detailed information.

Please review Application Note 95, “Interfacing the DS1307 with a 8051-Compatible Microcontroller” for additional information.

RTC AND RAM ADDRESS MAP

The address map for the RTC and RAM registers of the DS1307 is shown in Figure 2. The RTC registers are located in address locations 00h to 07h. The RAM registers are located in address locations 08h to 3Fh. During a multi-byte access, when the address pointer reaches 3Fh, the end of RAM space, it wraps around to location 00h, the beginning of the clock space.

DS1307 ADDRESS MAP Figure 2

00H	SECONDS
	MINUTES
	HOURS
	DAY
	DATE
	MONTH
	YEAR
07H	CONTROL
08H 3FH	RAM 56 x 8

CLOCK AND CALENDAR

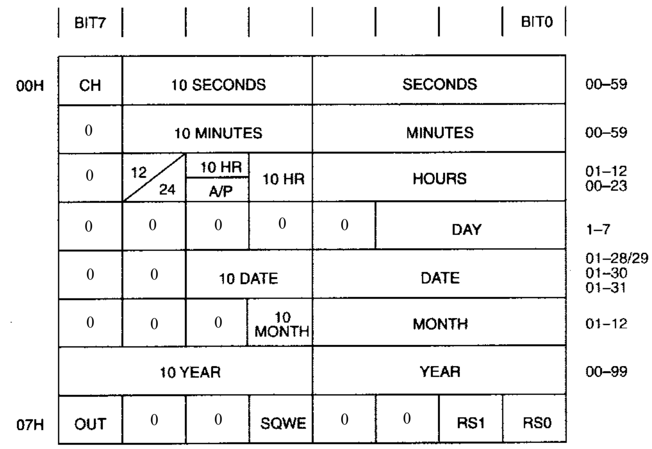
The time and calendar information is obtained by reading the appropriate register bytes. The RTC registers are illustrated in Figure 3. The time and calendar are set or initialized by writing the appropriate register bytes. The contents of the time and calendar registers are in the BCD format. Bit 7 of register 0 is the clock halt (CH) bit. When this bit is set to a 1, the oscillator is disabled. When cleared to a 0, the oscillator is enabled.

Please note that the initial power-on state of all registers is not defined. Therefore, it is important to enable the oscillator (CH bit = 0) during initial configuration.

The DS1307 can be run in either 12-hour or 24-hour mode. Bit 6 of the hours register is defined as the 12- or 24-hour mode select bit. When high, the 12-hour mode is selected. In the 12-hour mode, bit 5 is the AM/PM bit with logic high being PM. In the 24-hour mode, bit 5 is the second 10 hour bit (20-23 hours).

On a 2-wire START, the current time is transferred to a second set of registers. The time information is read from these secondary registers, while the clock may continue to run. This eliminates the need to re-read the registers in case of an update of the main registers during a read.

DS1307 TIMEKEEPER REGISTERS Figure 3



CONTROL REGISTER

The DS1307 control register is used to control the operation of the SQW/OUT pin.

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
OUT	0	0	SQWE	0	0	RS1	RS0

OUT (Output control): This bit controls the output level of the SQW/OUT pin when the square wave output is disabled. If SQWE = 0, the logic level on the SQW/OUT pin is 1 if OUT = 1 and is 0 if OUT = 0.

SQWE (Square Wave Enable): This bit, when set to a logic 1, will enable the oscillator output. The frequency of the square wave output depends upon the value of the RS0 and RS1 bits. With the square wave output set to 1Hz, the clock registers update on the falling edge of the square wave.

RS (Rate Select): These bits control the frequency of the square wave output when the square wave output has been enabled. Table 1 lists the square wave frequencies that can be selected with the RS bits.

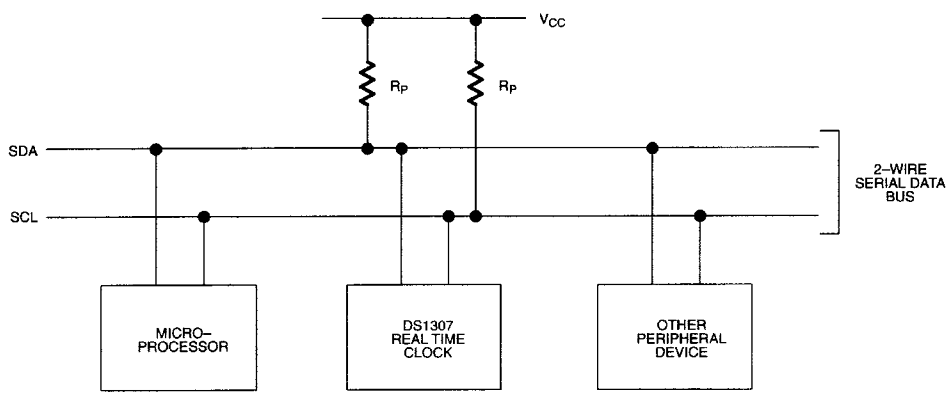
SQUAREWAVE OUTPUT FREQUENCY Table 1

RS1	RS0	SQW OUTPUT FREQUENCY
0	0	1Hz
0	1	4.096kHz
1	0	8.192kHz
1	1	32.768kHz

2-WIRE SERIAL DATA BUS

The DS1307 supports a bi-directional, 2-wire bus and data transmission protocol. A device that sends data onto the bus is defined as a transmitter and a device receiving data as a receiver. The device that controls the message is called a master. The devices that are controlled by the master are referred to as slaves. The bus must be controlled by a master device that generates the serial clock (SCL), controls the bus access, and generates the START and STOP conditions. The DS1307 operates as a slave on the 2-wire bus. A typical bus configuration using this 2-wire protocol is show in Figure 4.

TYPICAL 2-WIRE BUS CONFIGURATION Figure 4



Figures 5, 6, and 7 detail how data is transferred on the 2-wire bus.

- Data transfer may be initiated only when the bus is not busy.
- During data transfer, the data line must remain stable whenever the clock line is HIGH. Changes in the data line while the clock line is high will be interpreted as control signals.

Accordingly, the following bus conditions have been defined:

Bus not busy: Both data and clock lines remain HIGH.

Start data transfer: A change in the state of the data line, from HIGH to LOW, while the clock is HIGH, defines a START condition.

Stop data transfer: A change in the state of the data line, from LOW to HIGH, while the clock line is HIGH, defines the STOP condition.

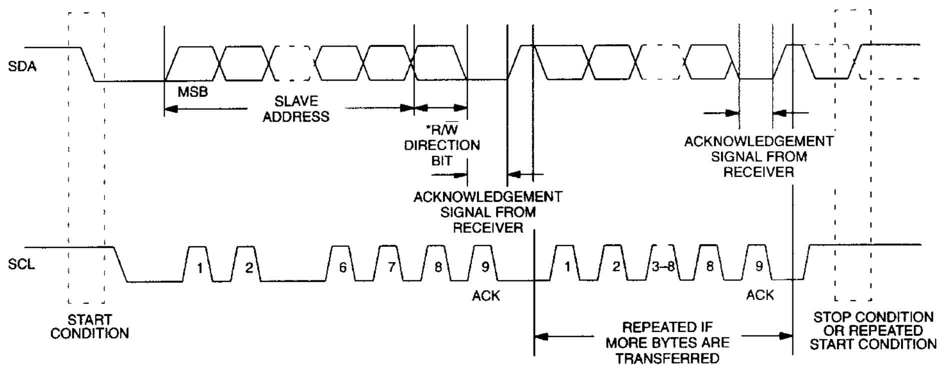
Data valid: The state of the data line represents valid data when, after a START condition, the data line is stable for the duration of the HIGH period of the clock signal. The data on the line must be changed during the LOW period of the clock signal. There is one clock pulse per bit of data.

Each data transfer is initiated with a START condition and terminated with a STOP condition. The number of data bytes transferred between START and STOP conditions is not limited, and is determined by the master device. The information is transferred byte-wise and each receiver acknowledges with a ninth bit. Within the 2-wire bus specifications a regular mode (100kHz clock rate) and a fast mode (400kHz clock rate) are defined. The DS1307 operates in the regular mode (100kHz) only.

Acknowledge: Each receiving device, when addressed, is obliged to generate an acknowledge after the reception of each byte. The master device must generate an extra clock pulse which is associated with this acknowledge bit.

A device that acknowledges must pull down the SDA line during the acknowledge clock pulse in such a way that the SDA line is stable LOW during the HIGH period of the acknowledge related clock pulse. Of course, setup and hold times must be taken into account. A master must signal an end of data to the slave by not generating an acknowledge bit on the last byte that has been clocked out of the slave. In this case, the slave must leave the data line HIGH to enable the master to generate the STOP condition.

DATA TRANSFER ON 2-WIRE SERIAL BUS Figure 5



Depending upon the state of the $R/\overline{W}$ bit, two types of data transfer are possible:

1. **Data transfer from a master transmitter to a slave receiver.** The first byte transmitted by the master is the slave address. Next follows a number of data bytes. The slave returns an acknowledge bit after each received byte. Data is transferred with the most significant bit (MSB) first.

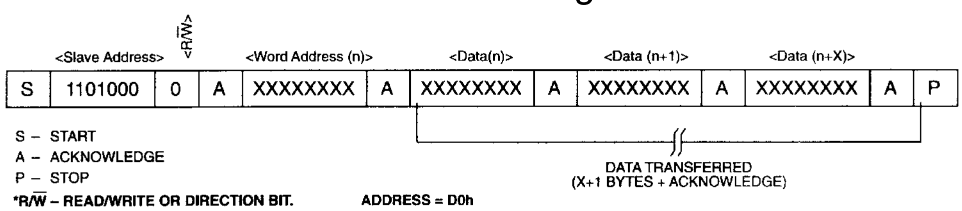
2. **Data transfer from a slave transmitter to a master receiver.** The first byte (the slave address) is transmitted by the master. The slave then returns an acknowledge bit. This is followed by the slave transmitting a number of data bytes. The master returns an acknowledge bit after all received bytes other than the last byte. At the end of the last received byte, a “not acknowledge” is returned.

The master device generates all of the serial clock pulses and the START and STOP conditions. A transfer is ended with a STOP condition or with a repeated START condition. Since a repeated START condition is also the beginning of the next serial transfer, the bus will not be released. Data is transferred with the most significant bit (MSB) first.

The DS1307 may operate in the following two modes:

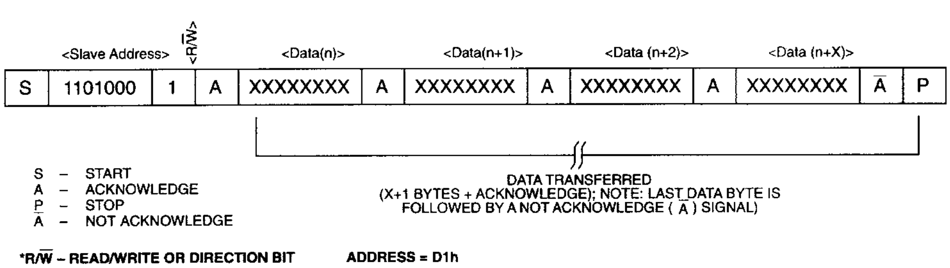
1. **Slave receiver mode (DS1307 write mode):** Serial data and clock are received through SDA and SCL. After each byte is received an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and *direction bit (See Figure 6). The address byte is the first byte received after the start condition is generated by the master. The address byte contains the 7 bit DS1307 address, which is 1101000, followed by the *direction bit ($R/\overline{W}$) which, for a write, is a 0. After receiving and decoding the address byte the device outputs an acknowledge on the SDA line. After the DS1307 acknowledges the slave address + write bit, the master transmits a register address to the DS1307 This will set the register pointer on the DS1307. The master will then begin transmitting each byte of data with the DS1307 acknowledging each byte received. The master will generate a stop condition to terminate the data write.

DATA WRITE – SLAVE RECEIVER MODE Figure 6



2. **Slave transmitter mode (DS1307 read mode):** The first byte is received and handled as in the slave receiver mode. However, in this mode, the *direction bit will indicate that the transfer direction is reversed. Serial data is transmitted on SDA by the DS1307 while the serial clock is input on SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer (See Figure 7). The address byte is the first byte received after the start condition is generated by the master. The address byte contains the 7-bit DS1307 address, which is 1101000, followed by the *direction bit ($R/\overline{W}$) which, for a read, is a 1. After receiving and decoding the address byte the device inputs an acknowledge on the SDA line. The DS1307 then begins to transmit data starting with the register address pointed to by the register pointer. If the register pointer is not written to before the initiation of a read mode the first address that is read is the last one stored in the register pointer. The DS1307 must receive a "not acknowledge" to end a read.

DATA READ – SLAVE TRANSMITTER MODE Figure 7



ABSOLUTE MAXIMUM RATINGS*

Voltage on Any Pin Relative to Ground	-0.5V to +7.0V
Storage Temperature	-55°C to +125°C
Soldering Temperature	260°C for 10 seconds DIP See JPC/JEDEC Standard J-STD-020A for Surface Mount Devices

* This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

Range	Temperature	V_{CC}
Commercial	0°C to +70°C	4.5V to 5.5V V_{CC1}
Industrial	-40°C to +85°C	4.5V to 5.5V V_{CC1}

RECOMMENDED DC OPERATING CONDITIONS

(Over the operating range*)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Supply Voltage	V_{CC}	4.5	5.0	5.5	V	
Logic 1	V_{IH}	2.2		V_{CC} + 0.3	V	
Logic 0	V_{IL}	-0.5		+0.8	V	
V_{BAT} Battery Voltage	V_{BAT}	2.0		3.5	V	

*Unless otherwise specified.

DC ELECTRICAL CHARACTERISTICS

(Over the operating range*)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Leakage (SCL)	I_{LI}			1	µA	
I/O Leakage (SDA & SQW/OUT)	I_{LO}			1	µA	
Logic 0 Output (I_{OL} = 5mA)	V_{OL}			0.4	V	
Active Supply Current	I_{CCA}			1.5	mA	7
Standby Current	I_{CCS}			200	µA	1
Battery Current (OSC ON); SQW/OUT OFF	I_{BAT1}		300	500	nA	2
Battery Current (OSC ON); SQW/OUT ON (32kHz)	I_{BAT2}		480	800	nA	
Power-Fail Voltage	V_{PF}	1.216 x V_{BAT}	1.25 x V_{BAT}	1.284 x V_{BAT}	V	8

*Unless otherwise specified.

AC ELECTRICAL CHARACTERISTICS

(Over the operating range*)

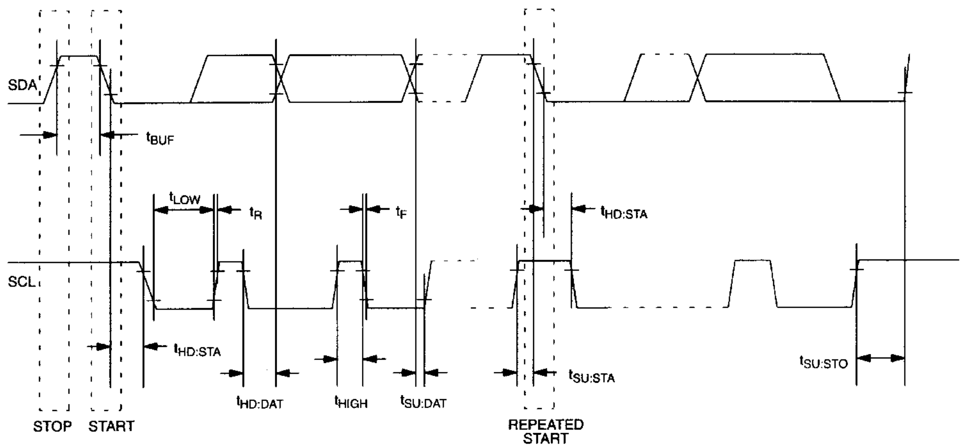
PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
SCL Clock Frequency	f_{SCL}	0		100	kHz	
Bus Free Time Between a STOP and START Condition	t_{BUF}	4.7			µs	
Hold Time (Repeated) START Condition	$t_{HD:STA}$	4.0			µs	3
LOW Period of SCL Clock	t_{LOW}	4.7			µs	
HIGH Period of SCL Clock	t_{HIGH}	4.0			µs	
Set-up Time for a Repeated START Condition	$t_{SU:STA}$	4.7			µs	
Data Hold Time	$t_{HD:DAT}$	0			µs	4,5
Data Set-up Time	$t_{SU:DAT}$	250			ns	
Rise Time of Both SDA and SCL Signals	t_R			1000	ns	
Fall Time of Both SDA and SCL Signals	t_F			300	ns	
Set-up Time for STOP Condition	$t_{SU:STO}$	4.7			µs	
Capacitive Load for each Bus Line	C_B			400	pF	6
I/O Capacitance (T_A = 25ºC)	$C_{I/O}$		10		pF	
Crystal Specified Load Capacitance (T_A = 25ºC)			12.5		pF	

*Unless otherwise specified.

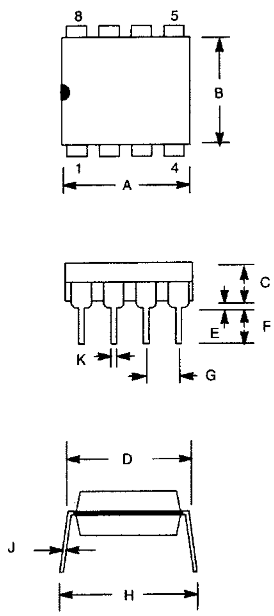
NOTES:

1. I_{CCS} specified with V_{CC} = 5.0V and SDA, SCL = 5.0V.
2. V_{CC} = 0V, V_{BAT} = 3V.
3. After this period, the first clock pulse is generated.
4. A device must internally provide a hold time of at least 300ns for the SDA signal (referred to the V_{IHMIN} of the SCL signal) in order to bridge the undefined region of the falling edge of SCL.
5. The maximum $t_{HD:DAT}$ has only to be met if the device does not stretch the LOW period (t_{LOW}) of the SCL signal.
6. C_B – Total capacitance of one bus line in pF.
7. I_{CCA} – SCL clocking at max frequency = 100kHz.
8. V_{PF} measured at V_{BAT} = 3.0V.

TIMING DIAGRAM Figure 8

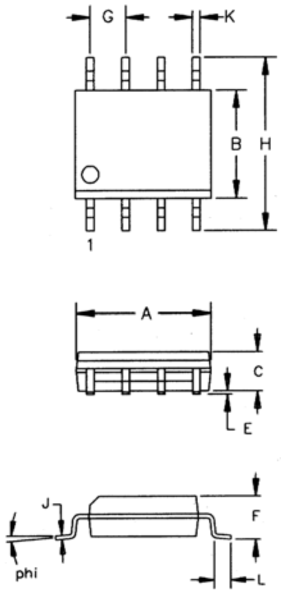


DS1307 64 X 8 SERIAL REAL-TIME CLOCK 8-PIN DIP MECHANICAL DIMENSIONS



PKG	8-PIN	
DIM	**MIN**	**MAX**
A IN. **MM**	0.360 9.14	0.400 10.16
B IN. **MM**	0.240 6.10	0.260 6.60
C IN. **MM**	0.120 3.05	0.140 3.56
D IN. **MM**	0.300 7.62	0.325 8.26
E IN. **MM**	0.015 0.38	0.040 1.02
F IN. **MM**	0.120 3.04	0.140 3.56
G IN. **MM**	0.090 2.29	0.110 2.79
H IN. **MM**	0.320 8.13	0.370 9.40
J IN. **MM**	0.008 0.20	0.012 0.30
K IN. **MM**	0.015 0.38	0.021 0.53

DS1307Z 64 X 8 SERIAL REAL-TIME CLOCK
8-PIN SOIC (150-MIL) MECHANICAL DIMENSIONS



PKG	8-PIN (150 MIL)	
DIM	MIN	MAX
A IN. MM	0.188 4.78	0.196 4.98
B IN. MM	0.150 3.81	0.158 4.01
C IN. MM	0.048 1.22	0.062 1.57
E IN. MM	0.004 0.10	0.010 0.25
F IN. MM	0.053 1.35	0.069 1.75
G IN. MM	0.050 BSC 1.27 BSC	
H IN. MM	0.230 5.84	0.244 6.20
J IN. MM	0.007 0.18	0.011 0.28
K IN. MM	0.012 0.30	0.020 0.51
L IN. MM	0.016 0.41	0.050 1.27
phi	0°	8°

56-G2008-001

```
; Function Discreption Template
;*******************************************************************************
;       "":
;       Func:
;       Input:
;       Output:
;       Affect:
;       Runtime:
;*******************************************************************************


    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 <common.inc>

    extern  LCD_init, LCD_clear, LCD_line2, LCD_out, LCD_wt
    extern  delay50us, delay5ms, delayX5msm, delay100ms, delayX100msm, delay1sl

;******************VARIABLES****************************************************
    cblock      0x20
    phase               ; 0 = realtime, 1 = report, 2 = run
    report_num
    line_num
    LCDline     : .17   ; 16 + null
    temp                ; only local use: must be discard before calling/jumping
    temp2
    temp3
    temp4
    temp5
    temp6
    temp7
    arg                 ; argument
    arg2
    literal_addr
    rowleft             ; how many rows left to inspect
    result_addr         ; adress to store result of current row in "layout"
    rt_year             ; realtime
    rt_month
    rt_day
    rt_hour
    rt_min
    rt_sec
    st_year             ; start
    st_month
    st_day
    st_hour
    st_min
    st_sec
    end_hour
    end_min
    end_sec
    runtime
    cl_total
    cl_pass
    cl_fail
    layout      : .20   ; result
    smotor_dir          ; direction of stepper motor
    log_total           ; total numbers of reports available
    log_next            ; index of where next report will be written
    arith_temp          ; arithmetic temp
    arith_temp2
    newsec              ; a new second occur (Bool from interrupt)
```

```
    endc

    cblock      0xB0
    light_bg    : 5     ; background light intensity of current row
    light_pos   : 5     ; position reflected light intensity of current row
    light_cl    : 5     ; closet light LEDs light intensity of current row
    light_off   : 5     ; closet light off light intensity of current row
    endc

    udata_shr
w_temp      res 1
status_temp res 1
;pclath_temp    res 1
FSR_temp    res 1
rt_counter  res 1       ; real time counter
table_temp  res 1

;*****************MACROS*****************************************************
DIVLW       macro   L
    movwf       arith_temp
    movlw       L
    call        divfn
    endm

MODLW       macro   L
    movwf       arith_temp
    movlw       L
    call        divfn
    movf        arith_temp, w
    endm

MULLW       macro   L
    movwf       arith_temp
    movlw       L
    call        mulfn
    endm

COPY_STRING macro   string_table
    movlw       string_table
    call        copystring
    endm

COPY_LAYOUT macro   rownum
    movlw       rownum
    call        copylayout
    endm

COPY_DEC1   macro   decnumber
    movf        decnumber, w
    movwf       arg
    movlw       0x01
    movwf       arg2
    call        copydec
    endm

COPY_DEC2   macro   decnumber
    movf        decnumber, w
    movwf       arg
    movlw       0x02
    movwf       arg2
    call        copydec
    endm

COPY_DEC3   macro   decnumber
```

```
    movf        decnumber, w
    movwf       arg
    movlw       0x03
    movwf       arg2
    call        copydec
    endm

TABLE       macro
    local   tablename
    movwf       table_temp
    movlw       HIGH tablename
    movwf       PCLATH
    movf        table_temp, w
    addlw       LOW tablename
    btfsc       STATUS, C
    incf        PCLATH, f
    movwf       PCL
tablename
    endm

STORE_LIGHT macro   addr
    movlw       addr
    call        lightsensor
    endm

IRLED_ON    macro
    movlw       0x01
    call        ledcontrol
    endm

IRLED_OFF   macro
    movlw       0x00
    call        ledcontrol
    endm

CALC_POS    macro   threshold
    movwf       temp
    movlw       threshold
    movwf       temp4
    call        calcpos
    endm

PRINT_DOT   macro
    MOVLF       FSR, LCDline
    COPY_STRING str_dot
    call        writeline
    endm

;*******************************************************************************
;*******************************************************************************
    org         0x0000
#IFNDEF     DEBUG
    goto        main
#ELSE
    goto        mac_test
#ENDIF

#IFNDEF     DEBUG
    org         0x0004
    goto        interrupt
#ENDIF

;*******************************************************************************
;       "literal":  String Literal Function
```

```
;       Func:       Store all string literal in this project here, it return
;                   any char wanted
;       Input:      W = the program memory address of the wanted char
;       Output:     W = the corresponding char
;*******************************************************************************
    org         0x0005          ; make sure literal table is in first 256 lines
literal
    movwf       temp
    clrf        PCLATH
    movf        temp, w
    movwf       PCL
; max length 16 dt  "0123456789ABCDEF", 0
str_init        dt  "INITIALIZING...", 0
str_noreport    dt  "NO REPORT", 0
str_enterymd    dt  "ENTER 20YYMMDD:",0
str_enterhms    dt  "ENTER hhmmss:", 0
str_yearhead    dt  "20", 0     ; all years like 20XX
str_dash        dt  "-", 0
str_colon       dt  ":", 0
str_entry       dt  "LOG ENTRY: ",0
str_start       dt  "START:  ", 0
str_finish      dt  "FINISH: ", 0
str_runtime     dt  "RUNTIME:    ", 0
str_s           dt  "s", 0      ; i.e. second
str_total       dt  "TOTAL: ", 0
str_pass        dt  "PASS: ", 0
str_fail        dt  " FAIL: ", 0
str_layout      dt  "LAYOUT:", 0
str_layout1     dt  " TOP  1  ", 0
str_layout2     dt  "      2  ", 0
str_layout3     dt  "      3  ", 0
str_layout4     dt  " BOT  4  ", 0
str_running     dt  "RUNNING", 0
str_dot         dt  ".", 0
str_insp_fin    dt  "INSPECTION FIN", 0
str_emerstop    dt  "EMERGENCY STOP", 0
str_null        dt  0
; corresponds to the layout byte
char_layout     dt  "XFFPXFFP"  ;"XF2P4567"; "XFEPEEEE"
; corresponds to keypad
char_keynumber dt  "123", 0, "456", 0, "789", 0, 0, "0", 0, 0
char_slash      dt  "/", 0
;????!!!!copy spaceX function

    code
#IFNDEF     DEBUG
;*******************************************************************************
;       "main":     Main Function
;       Func:       Call initialization and then keep polling the keypad for
;                   input (or interrupt)
;       Input:      None
;       Output:     None
;       Affect:     *
;*******************************************************************************
main
    call        init

keypoll
    btfsc       KEYPAD_DA
    call        keyresp
    call        keypad_timeout
    goto        keypoll

#ELSE
```

```
;***************************************************************************
;   "mac_test":     Machenical System Test Function
;       Func:       Test the machenical system, each botton on keypad
;                   corresponds to a machine function (i.e. driver)
;       Input:      None
;       Output:     None
;       Affect:     *
;***************************************************************************
mac_test
    call        init_pic
    call        reset_software
;   call        reset_hardware
    ; PORT Reset/Initialization
;   BANK0
    clrf        PORTA           ; RA4 (Reserved) output 0
    clrf        PORTB           ; RB0, 2, 3 (Reserved) output 0
    clrf        PORTC           ; VMOTOR disable, LED off
    clrf        PORTD           ; clear S0, S1 of SMOTOR
    clrf        PORTE           ; SMOTOR disable, clear S2, S3 of SMOTOR

    ; LCD Reset/Initialization
    call        LCD_init
    call        display

mac_test_loop
    call        analogtest2
    btfsc       KEYPAD_DA
    call        keyresp_ma
    movlw       0x64
    call        delayX5msm
    goto        mac_test_loop

keyresp_ma
    swapf       PORTB, w
    andlw       0x0F
    TABLE
;keyresp_switch_table
    goto        kma0            ; keypressed = 0   "1" = SM forward
    goto        kma1            ; keypressed = 1   "2" = SM backward
    goto        kma2            ; keypressed = 2   "3" = SM step
    goto        kma3            ; keypressed = 3   "A"
    goto        kma4            ; keypressed = 4   "4" = DC up
    goto        kma5            ; keypressed = 5   "5" = DC down
    goto        kma6            ; keypressed = 6   "6" = DC stop
    goto        kma7            ; keypressed = 7   "B"
    goto        kma8            ; keypressed = 8   "7" = IRLED on
    goto        kma9            ; keypressed = 9   "8" = IRLED off
    goto        kmaA            ; keypressed = A   "9"
    goto        kmaB            ; keypressed = B   "C"
    goto        kmaC            ; keypressed = C   "*"
    goto        kmaD            ; keypressed = D   "0"
    goto        kmaE            ; keypressed = E   "#"
    goto        kmaF            ; keypressed = F   "D"
kma_next
kma3
kma7
kmaA
kmaB
kmaC
kmaD
kmaE
kmaF
kma_release
    btfsc       KEYPAD_DA       ;Wait until key is released
```

```
	goto		kma_release
	return

kma0
	movlw		0x00
	movwf		smotor_dir
	call		advancerow
	goto		kma_next
kma1
	movlw		0x01
	movwf		smotor_dir
	call		advancerow
	goto		kma_next
kma2
	bcf		SMOTOR_EN
	goto		kma_next
kma4
	call		moveup
	goto		kma_next
kma5
	call		movedown
	goto		kma_next
kma6
	bcf		VMOTOR_C0
	bcf		VMOTOR_C1
	goto		kma_next
kma8
	IRLED_ON
	goto		kma_next
kma9
	IRLED_OFF
	goto		kma_next

;*******************************************************************************
;	"analogtest2": Analog-to-Digital Test Function
;		Func:		Convert RA0 to digital and display its value every second
;		Input:
;		Output:
;		Affect:
;*******************************************************************************
analogtest2
	STORE_LIGHT light_bg

	MOVLF		temp5, COLS	; counter
	MOVLF		temp2, LCDline	; LCDline addr
	MOVLF		temp3, light_bg; storage addr

	MOVFF		FSR, temp3
	MOVFF		temp4, INDF	; temp storage
	MOVFF		FSR, temp2
	COPY_DEC3	temp4
	COPY_STRING str_dot
	COPY_STRING str_dot
	COPY_STRING str_dot
	MOVFF		temp2, FSR
	incf		temp3, f
	incf		temp3, f

	MOVFF		FSR, temp3
	MOVFF		temp4, INDF	; temp storage
	MOVFF		FSR, temp2
	COPY_DEC3	temp4
	COPY_STRING str_dot
	COPY_STRING str_dot
```

```
    COPY_STRING str_dot
    MOVFF       temp2, FSR
    incf        temp3, f
    incf        temp3, f

    MOVFF       FSR, temp3
    MOVFF       temp4, INDF     ; temp storage
    MOVFF       FSR, temp2
    COPY_DEC3   temp4
    COPY_STRING str_dot
    COPY_STRING str_dot
    COPY_STRING str_dot
    MOVFF       temp2, FSR

    call        LCD_clear       ; clear LCD display
    call        writeline
    call        LCD_line2

    MOVLF       temp2, LCDline ; LCDline addr
    MOVLF       temp3, light_bg; storage addr
    incf        temp3, f

    MOVFF       FSR, temp3
    MOVFF       temp4, INDF     ; temp storage
    MOVFF       FSR, temp2
    COPY_STRING str_dot
    COPY_STRING str_dot
    COPY_STRING str_dot
    COPY_DEC3   temp4
    COPY_STRING str_dot
    COPY_STRING str_dot
    COPY_STRING str_dot
    MOVFF       temp2, FSR
    incf        temp3, f
    incf        temp3, f

    MOVFF       FSR, temp3
    MOVFF       temp4, INDF     ; temp storage
    MOVFF       FSR, temp2
    COPY_DEC3   temp4
    MOVFF       temp2, FSR

    call        writeline
    call        LCD_out
;ADCtest2_loop2
;   MOVFF       FSR, temp3
;   MOVFF       temp4, INDF     ; temp storage
;   MOVFF       FSR, temp2
;   COPY_DEC3   temp4
;;  COPY_STRING str_dot
;   MOVFF       temp2, FSR
;   incf        temp3, f
;   decfsz      temp5,f
;   goto        ADCtest2_loop2
;
;   call        LCD_clear       ; clear LCD display
;   call        writeline
    return

;*******************************************************************************
;   "analogtest":  Analog-to-Digital Test Function
;       Func:      Convert RA0 to digital and display its value every second
;       Input:
;       Output:
```

```
;       Affect:
;*******************************************************************************
analogtest
ADCtest_loop
    call        delay50us       ; require 2Tosc + Tacq = 28us
    bsf         ADCON0, GO
analog_poll                     ; about 40us
    btfsc       ADCON0, GO
    goto        analog_poll
    movf        ADRESH, w

    MOVLF       FSR, LCDline
    COPY_DEC1   ADRESH
    call        LCD_clear       ; clear LCD display
    call        writeline

    ; delay 0.5s
    movlw       0x64
    movwf       temp
ADCtest_simpledelay
    call        delay5ms
    decfsz      temp, f
    goto        ADCtest_simpledelay

    goto        ADCtest_loop
#ENDIF

#IFNDEF    DEBUG
;*******************************************************************************
;   "interrupt":    Interrupt Handle Function
;       Func:       Handle all interrupt that occurs in runtime
;       Input:      INTCON
;       Output:     INTCON, rt_*, newsec
;       Affect:     None (w_temp, status_temp)
;*******************************************************************************
interrupt
    movwf       w_temp          ; save W
    swapf       STATUS, w       ; save STATUS, note swapf will not affect STATUS
    BANK00
    movwf       status_temp
;   movf        PCLATH, w       ; save page information
;   movwf       pclath_temp
;   clrf        PCLATH
    movf        FSR, w          ; save FSR
    movwf       FSR_temp

    ; Timer0 Interrupt Handle
    ; T0IE always on, no test
    btfss       INTCON, T0IF
    goto        int_tmr0_skip
    incf        rt_counter, f
    movf        rt_counter, w   ; test with 98h, correponds to 996,147.2us
    sublw       0x98
    btfss       STATUS, Z
    goto        int_tmr0_end    ; rt_counter <> 98h
    clrf        rt_counter      ; rt_counter == 98h
    movlw       rt_sec
    call        addsec
    movlw       rt_day
    btfsc       STATUS, C
    call        addday          ; addday if carry from addsec
    MOVLF       newsec, TRUE
int_tmr0_end
    bcf         INTCON, T0IF
```

```
int_tmr0_skip

    ; Keypad (PORTB) Change Interrupt
    btfss       INTCON, RBIE    ; interrept must be enabled first
    goto        int_rb_skip
    btfss       INTCON, RBIF
    goto        int_rb_skip
    swapf       PORTB, w        ; test PORTB<7:4> against stop button
    andlw       0x0F
    sublw       STOP_BUT
    btfss       STATUS, Z
    goto        int_rb_end
    ; Emergency Stop!!!!
    bcf         IRLED           ; turn off IRLEDs
    bcf         VMOTOR_C0       ; turn off v motor
    bcf         VMOTOR_C1
    bcf         SMOTOR_EN       ; turn off s motor
    MOVLF       FSR, LCDline    ; display emergency stop
    COPY_STRING str_emerstop
    call        LCD_clear           ; clear LCD display
    call        writeline
    call        LCD_out
stop        goto        stop    ; hang the program
int_rb_end
    bcf         INTCON, RBIF
int_rb_skip

    movf        FSR_temp, w     ; restore FSR
    movwf       FSR
;   movf        pclath_temp, w ; restore page information
;   movwf       PCLATH
    swapf       status_temp, w ; restore STATUS
    movwf       STATUS
    swapf       w_temp, f       ; restore W, not affecting STATUS
    swapf       w_temp, w
    retfie

;*******************************************************************************
;       "init":     Initialization Function
;       Func:       Initialize chip settings, variables and reset hardware
;                   position
;       Input:      None
;       Output:     *
;       Affect:     W, STATUS, temp, delaytemp, delaycount, delaytemp2,
;                   delaycount2, delaytemp3, delaycount3
;*******************************************************************************
init
    call        init_pic
    call        reset_software
    call        reset_hardware
    call        reset_realtime
    clrf        TMR0            ; start timing
    bsf         INTCON, T0IE
    bsf         INTCON, GIE
    movlw       PHASE_REALTIME
    movwf       phase
    call        display
    return
#ENDIF

;*******************************************************************************
;   "init_pic":     PIC Initialization Function
;       Func:       Initialize chip settings: Interrupt, TMR0, PORT, ADC
;       Input:      None
```

```
;       Output:     INTCON, TRISA, TRISB, TRISC, TRISD, TRISE, ADCCON0, ADCCON1
;       Affect:     W, STATUS
;*******************************************************************************
init_pic
    ; Interrupt Initialization
    ; Disable Global interrupt, diable peripheral interrupt, enable timer0 and
    ; PORTB interrupton change (keypad), diable RB0 interrupt
    ; #define   INITVAL_INTCON B'00101000'
    ; movlw     INITVAL_INTCON
    clrf        INTCON

    ;
    ;clrf       SSPBUF
    ;BANK1
    ;clrf       TXSTA
    ;clrf       PIE1
    ;clrf       PIE2

    ; Timer0 Initialization
    BANK0
    clrf        TMR0
    clrf        rt_counter
    BANK1
    movlw       INITVAL_OPTREG
    movwf       OPTION_REG

    ; PORT Initialzation
;   BANK1
    movlw       INITVAL_TRISA
    movwf       TRISA
    movlw       INITVAL_TRISB
    movwf       TRISB
    movlw       INITVAL_TRISC
    movwf       TRISC
    movlw       INITVAL_TRISD
    movwf       TRISD
    movlw       INITVAL_TRISE
    movwf       TRISE

    ; Analog to Digital Convertor Initialization
;   BANK1
    movlw       INITVAL_ADCON1
    movwf       ADCON1
    BANK0
    movlw       INITVAL_ADCON0
    movwf       ADCON0

    return

;*******************************************************************************
;"reset_software": Software Reset/Initialization Function
;       Func:       Reset all variables
;       Input:      None
;       Output:     phase, report_num, line_num, log_total, log_next, smotor_dir
;       Affect:     STATUS
;*******************************************************************************
reset_software
    movlw       PHASE_HDINIT
    movwf       phase
    clrf        report_num
    clrf        line_num
    clrf        log_total
    clrf        log_next
    MOVLF       smotor_dir, 1
```

```
    return

;*******************************************************************************
;"reset_hardware": Hardware Reset/Initialization Function
;       Func:       Reset/Initialize hardwares to their default position:
;                   Pos Sensor Off, V DC Motor at top, S Motor at "Row 1",
;                   LCD, (RTC)
;       Input:      None
;       Output:     PORTA, PORTB, PORTC, PORTD, PORTE
;       Affect:     W, STATUS, temp, delaytemp, delaycount, delaytemp2,
;                   delaycount2, delaytemp3, delaycount3
;*******************************************************************************
reset_hardware
    ; PORT Reset/Initialization
;   BANK0
    clrf        PORTA           ; RA4 (Reserved) output 0
    clrf        PORTB           ; RB0, 2, 3 (Reserved) output 0
    clrf        PORTC           ; VMOTOR disable, LED off
    clrf        PORTD           ; clear S0, S1 of SMOTOR
    clrf        PORTE           ; SMOTOR disable, clear S2, S3 of SMOTOR

    ; LCD Reset/Initialization
    call        LCD_init
    call        display

    ; Position Sensor(IRLED) Reset/Initialization: all off
    ; Already done with PORT reset

    ; Vertical DC Motor Reset/Initialization: move to top
    call        moveup

    ; Stepper Motor Reset/Initialization: move to "Row 4"
    clrf        smotor_dir
    call        advancerow      ; 4 advance row to ensure to init pos
    call        advancerow
    call        advancerow
    call        advancerow
    MOVLF       smotor_dir, 1

    return

;*******************************************************************************
;"reset_realtime": Real Time Clock Reset/Initialization Function
;       Func:       Reset Real Time Clock
;       Input:      None (from Keypad)
;       Output:     rt_year, rt_month, rt_day, rt_hour, rt_min, rt_sec
;       Affect:
;*******************************************************************************
reset_realtime
    movlw       PHASE_RTCINIT
    movwf       phase
    clrf        rt_year
    clrf        rt_month
    clrf        rt_day
    clrf        rt_hour
    clrf        rt_min
    clrf        rt_sec

    ; YYMMDD
    call        LCD_clear       ; clear LCD display
    MOVLF       FSR, LCDline
    COPY_STRING str_enterymd
    call        writeline
```

```
    call        LCD_line2
    MOVLF       FSR, LCDline
    COPY_STRING str_yearhead
    call        writeline

    clrf        temp3          ; number of valid numbers entered
reset_rt_ymd
    btfss       KEYPAD_DA      ; Wait until data is available from the keypad
    goto        reset_rt_ymd

    swapf       PORTB, W       ; Read PortB<7:4> into W<3:0>
    andlw       0x0F
    addlw       char_keynumber
    call        literal        ; Convert keypad value to LCD character (value is still held in W)
    addlw       0x00
    btfsc       STATUS, Z      ; test for valid input (number)
    goto        reset_rt_ymdrl

    movwf       temp2          ; hold the value
    call        LCD_wt         ; Write the value in W to LCD
    movlw       0x30
    subwf       temp2, f       ; convert ASCII to number
    movf        temp3, w
    TABLE                      ; switch (temp)
    goto        reset_rt_ymd0
    goto        reset_rt_ymd1
    goto        reset_rt_ymd2
    goto        reset_rt_ymd3
    goto        reset_rt_ymd4
    goto        reset_rt_ymd5
reset_rt_ymd0
    movf        temp2, w
    MULLW       .10
    addwf       rt_year, f
    incf        temp3, f
    goto        reset_rt_ymdrl
reset_rt_ymd1
    movf        temp2, w
    addwf       rt_year, f
    incf        temp3, f
    goto        reset_rt_ymdrl
reset_rt_ymd2
    movf        temp2, w
    MULLW       .10
    addwf       rt_month, f
    incf        temp3, f
    goto        reset_rt_ymdrl
reset_rt_ymd3
    movf        temp2, w
    addwf       rt_month, f
    incf        temp3, f
    goto        reset_rt_ymdrl
reset_rt_ymd4
    movf        temp2, w
    MULLW       .10
    addwf       rt_day, f
    incf        temp3, f
    goto        reset_rt_ymdrl
reset_rt_ymd5
    movf        temp2, w
    addwf       rt_day, f
    incf        temp3, f

reset_rt_ymdrl
```

```
    btfsc       KEYPAD_DA      ; Wait until key is released
    goto        reset_rt_ymdrl
    movlw       0x06           ; 6 chars entered
    subwf       temp3, w
    btfss       STATUS, C
    goto        reset_rt_ymd

    ; hhmmss
    call        LCD_clear      ; clear LCD display
    MOVLF       FSR, LCDline
    COPY_STRING str_enterhms
    call        writeline

    call        LCD_line2
    clrf        temp3          ; number of valid numbers entered
reset_rt_hms
    btfss       KEYPAD_DA      ; Wait until data is available from the keypad
    goto        reset_rt_hms

    swapf       PORTB, W       ; Read PortB<7:4> into W<3:0>
    andlw       0x0F
    addlw       char_keynumber
    call        literal        ; Convert keypad value to LCD character (value is still held in W)
    addlw       0x00
    btfsc       STATUS, Z      ; test for valid input (number)
    goto        reset_rt_hmsrl

    movwf       temp2          ; hold the value
    call        LCD_wt         ; Write the value in W to LCD
    movlw       0x30
    subwf       temp2, f       ; convert ASCII to number
    movf        temp3, w
    TABLE                      ; switch (temp)
    goto        reset_rt_hms0
    goto        reset_rt_hms1
    goto        reset_rt_hms2
    goto        reset_rt_hms3
    goto        reset_rt_hms4
    goto        reset_rt_hms5
reset_rt_hms0
    movf        temp2, w
    MULLW       .10
    addwf       rt_hour, f
    incf        temp3, f
    goto        reset_rt_hmsrl
reset_rt_hms1
    movf        temp2, w
    addwf       rt_hour, f
    incf        temp3, f
    goto        reset_rt_hmsrl
reset_rt_hms2
    movf        temp2, w
    MULLW       .10
    addwf       rt_min, f
    incf        temp3, f
    goto        reset_rt_hmsrl
reset_rt_hms3
    movf        temp2, w
    addwf       rt_min, f
    incf        temp3, f
    goto        reset_rt_hmsrl
reset_rt_hms4
    movf        temp2, w
    MULLW       .10
```

```
    addwf       rt_sec, f
    incf        temp3, f
    goto        reset_rt_hmsrl
reset_rt_hms5
    movf        temp2, w
    addwf       rt_sec, f
    incf        temp3, f

reset_rt_hmsrl
    btfsc       KEYPAD_DA       ; Wait until key is released
    goto        reset_rt_hmsrl
    movlw       0x06            ; 6 chars entered
    subwf       temp3, w
    btfss       STATUS, C
    goto        reset_rt_hms

    return

;*******************************************************************************
;       "keyresp":  Key Response Function
;       Func:       Display information or run an inspection according to the
;                   pressed key
;       Input:      W: Index of the key that being pressed
;       Output:     All actions
;       Affect:
;*******************************************************************************
keyresp
    swapf       PORTB, w
    andlw       0x0F
    TABLE
;keyresp_switch_table
    goto        realtime        ; keypressed = 0   "1" = "real time"
    goto        report          ; keypressed = 1   "2" = "report"
    goto        unused_key      ; keypressed = 2   "3"
    goto        unused_key      ; keypressed = 3   "A"
    goto        unused_key      ; keypressed = 4   "4"
    goto        unused_key      ; keypressed = 5   "5"
    goto        unused_key      ; keypressed = 6   "6"
    goto        unused_key      ; keypressed = 7   "B"
    goto        scroll_up       ; keypressed = 8   "7" = "scroll up"
    goto        report_last     ; keypressed = 9   "8" = "report last"
    goto        unused_key      ; keypressed = A   "9"
    goto        stoprun         ; keypressed = B   "C" = "stop"
    goto        scroll_down     ; keypressed = C   "*" = "scroll down"
    goto        report_next     ; keypressed = D   "0" = "scroll up"
    goto        unused_key      ; keypressed = E   "#"
    goto        startrun        ; keypressed = F   "D" = "run"
unused_key
keyresp_next
wait_release
    btfsc       KEYPAD_DA       ; Wait until key is released????
    goto        wait_release    ; !!!!!!!!!!time

    call        display
    ; reset timeout!!!!!!!!!!
    return

realtime
    movlw       PHASE_REALTIME
    movwf       phase
    clrf        line_num
    goto        keyresp_next
report
    movlw       PHASE_REPORT    ; assuem log_total <> 0
```

```
    movf        log_total, f
    btfsc       STATUS, Z
    movlw       PHASE_NOREPORT ; log_total == 0, no report
    movwf       phase
    clrf        line_num
    goto        keyresp_next
report_last
; !!!! some ideas: must in REPORT phase; no change in line_num if at last report;
; do not go through report; if enter from other phase same fn to report;
; display report# and can go across upper/lower limit
    movf        log_total, f
    btfsc       STATUS, Z
    goto        report         ; log_total == 0, no report!!!!
    ; lower bound = (log_next - log_total + MAXLOG) MOD MAXLOG
    movf        log_total, w   ; get lower bound
    subwf       log_next, w
    addlw       MAXLOG
    MODLW       MAXLOG
    subwf       report_num, w
    btfsc       STATUS, Z
    goto        report         ; current report at lower bound!!!!"This is the last report"!!!!
    decf        report_num, w  ; get last report
    addlw       MAXLOG         ; make sure report_num between 0 and MAXLOG - 1
    MODLW       MAXLOG
    movwf       report_num
    call        readlog
    goto        report
report_next
    movf        log_total, f
    btfsc       STATUS, Z
    goto        report         ; log_total == 0, no report!!!!
    ; upper bound = (log_next - 1 + MAXLOG) MOD MAXLOG
    decf        log_next, w ; get upper bound
    addlw       MAXLOG
    MODLW       MAXLOG
    subwf       report_num, w
    btfsc       STATUS, Z
    goto        report         ; current report at upper bound!!!!
    incf        report_num, w  ; get next report
    MODLW       MAXLOG         ; make sure report_num between 0 and MAXLOG - 1
    movwf       report_num
    call        readlog
    goto        report
scroll_up
    movlw       PHASE_REPORT   ; test phase == PHASE_REPORT
    subwf       phase, w
    btfss       STATUS, Z
    goto        keyresp_next   ; phase <> PHASE_REPORT, do nothing
    movf        line_num, f    ; phase == PHASE_REPORT, test line_num == 0
    btfss       STATUS, Z
    decf        line_num, f    ; line_num <> 0, decrease line_num (scroll up)
    goto        keyresp_next
scroll_down
    movlw       PHASE_REPORT   ; test phase == PHASE_REPORT
    subwf       phase, w
    btfss       STATUS, Z
    goto        keyresp_next   ; phase <> PHASE_REPORT, do nothing
    movlw       MAXLINE        ; phase == PHASE_REPORT, test line_num == MAXLINE????
    subwf       line_num, w
    btfss       STATUS, Z
    incf        line_num, f    ; line_num <> MAXLINE, increase line_num (scroll down)
    goto        keyresp_next
startrun
    movlw       PHASE_RUN
```

```
    movwf       phase
    clrf        line_num
;!!!!  call        display
    call        LCD_clear       ; clear LCD display
    MOVLF       FSR, LCDline
    COPY_STRING str_running
    call        writeline

    call        run

    MOVLF       phase, PHASE_FINISH
    clrf        line_num
    call        display

;   call        delay1sl

    MOVLF       phase, PHASE_REPORT
    clrf        line_num
    call        display
    ; assume run time is very long, key has been released
    return
stoprun
    goto        keyresp_next

;*******************************************************************************
;"keypad_timeout":
;       Func:
;       Input:
;       Output:
;       Affect:
;*******************************************************************************
keypad_timeout
    movf        newsec, f
    btfsc       STATUS, Z
    goto        kp_to_nonewsec  ; newsec == 0(FALSE), skip
    call        display         ; newsec == TRUE, display the new sec
    clrf        newsec
kp_to_nonewsec
; SLEEP test goes here!!!!
    return

;*******************************************************************************
;       "display":  Display Function (User Interface)
;       Func:       First make up the content to be displayed into LCDline
;                   according to phase and line_num, then print the string
;                   to the LCD
;       Input:      phase, line_num
;       Output:     None (to LCD)
;       Affect:     W, STATUS, FSR, temp2, table_temp, arg, arg2,
;                   delaytemp, delaycount, lcd_temp
;*******************************************************************************
display
    ; run phase does not use general display function
    movf        phase, w
    sublw       PHASE_RUN
    btfsc       STATUS, Z
    return

    movf        line_num, w     ; current line#
    call        makeline

    call        LCD_clear       ; clear LCD display
    call        writeline
```

```
    incf        line_num, w     ; next line# in W, but not inc line#
    call        makeline

    call        LCD_line2
    call        writeline

    call        LCD_out         ; move the cursor out of screen
    return

;*******************************************************************************
;   "makeline":     Displayable Line Make & Copy Function
;       Func:       Make lines to be displayed according to the phase and
;                   line number (W), copy it to the LCDline array
;       Input:      W = line # to be displayed, phase
;       Output:     FSR = point to the null termination of the line made &
;                   copied, (Copied line in LCDline)
;       Affect:     W, STATUS, temp2, temp3, table_temp, arg, arg2, literal_addr
;*******************************************************************************
makeline
    movwf       temp2           ; line#
    MOVLF       FSR, LCDline    ; start from LCDline
    movf        phase, w
    TABLE                       ; switch (phase)
    goto        ML_hdinit       ; phase == 0
    goto        ML_rtcinit      ; phase == 1
    goto        ML_realtime     ; phase == 2
    goto        ML_noreport     ; phase == 3
    goto        ML_report       ; phase == 4
    goto        ML_run          ; phase == 5
    goto        ML_finish       ; phase == 6

ML_hdinit
    movf        temp2, w
    TABLE                       ; switch (temp2(line#))
    goto        ML_hdinit_0     ; line# == 0
    goto        ML_hdinit_1     ; line# == 1
ML_rtcinit      ; display do not use this function
    COPY_STRING str_null        ; null termination
    return
ML_realtime
    movf        temp2, w
    TABLE                       ; switch (temp2(line#))
    goto        ML_realtime_0   ; line# == 0
    goto        ML_realtime_1   ; line# == 1
ML_noreport
    movf        temp2, w
    TABLE                       ; switch (temp2(line#))
    goto        ML_noreport_0   ; line# == 0
    goto        ML_noreport_1   ; line# == 1
ML_report
    movf        temp2, w
    TABLE                       ; switch (temp2(line#))
    goto        ML_report_0     ; line# == 0
    goto        ML_report_1     ; line# == 1
    goto        ML_report_2     ; line# == 2
    goto        ML_report_3     ; line# == 3
    goto        ML_report_4     ; line# == 4
    goto        ML_report_5     ; line# == 5
    goto        ML_report_6     ; line# == 6
    goto        ML_report_7     ; line# == 7
    goto        ML_report_8     ; line# == 8
    goto        ML_report_9     ; line# == 9
    goto        ML_report_10    ; line# == 10
ML_run
```

```
    movf        temp2, w
    TABLE                       ; switch (temp2(line#))
    goto        ML_run_0        ; line# == 0
    goto        ML_run_1        ; line# == 1
ML_finish
    movf        temp2, w
    TABLE                       ; switch (temp2(line#))
    goto        ML_finish_0     ; line# == 0
    goto        ML_finish_1     ; line# == 1

ML_hdinit_0
    COPY_STRING str_init
    return
ML_hdinit_1
    COPY_STRING str_null
    return

ML_realtime_0
    COPY_STRING str_yearhead
    COPY_DEC2   rt_year
    COPY_STRING str_dash
    COPY_DEC2   rt_month
    COPY_STRING str_dash
    COPY_DEC2   rt_day
    return
ML_realtime_1
    COPY_DEC2   rt_hour
    COPY_STRING str_colon
    COPY_DEC2   rt_min
    COPY_STRING str_colon
    COPY_DEC2   rt_sec
    return

ML_noreport_0
    COPY_STRING str_noreport
    return
ML_noreport_1
    COPY_STRING str_null
    return


ML_report_0
    COPY_STRING str_entry
    ; log# = (log_total - log_next + report_num + 1 + MAXLOG) MOD MAXLOG
    ; also log# = MAXLOG if result == 0
    movf        log_next, w
    subwf       log_total, w
    addwf       report_num, w
    addlw       0x01
    addlw       MAXLOG
    MODLW       MAXLOG
    btfsc       STATUS, Z
    movlw       MAXLOG          ; result == 0, log# = MAXLOG
    movwf       temp3
    COPY_DEC2   temp3
    COPY_STRING char_slash
    COPY_DEC2   log_total
    return
ML_report_1
    COPY_STRING str_start
    COPY_DEC2   st_hour
    COPY_STRING str_colon
    COPY_DEC2   st_min
    COPY_STRING str_colon
```

```
    COPY_DEC2   st_sec
    return
ML_report_2
    COPY_STRING str_finish
    COPY_DEC2   end_hour
    COPY_STRING str_colon
    COPY_DEC2   end_min
    COPY_STRING str_colon
    COPY_DEC2   end_sec
    return
ML_report_3
    COPY_STRING str_runtime
    COPY_DEC1   runtime
    COPY_STRING str_s
    return
ML_report_4
    COPY_STRING str_total
    COPY_DEC1   cl_total
    return
ML_report_5
    COPY_STRING str_pass
    COPY_DEC1   cl_pass
    COPY_STRING str_fail
    COPY_DEC1   cl_fail
    return
ML_report_6
    COPY_STRING str_layout
    return
ML_report_7
    COPY_STRING str_layout1
    COPY_LAYOUT 0
    return
ML_report_8
    COPY_STRING str_layout2
    COPY_LAYOUT 1
    return
ML_report_9
    COPY_STRING str_layout3
    COPY_LAYOUT 2
    return
ML_report_10
    COPY_STRING str_layout4
    COPY_LAYOUT 3
    return

ML_run_0
    COPY_STRING str_running
    return
ML_run_1
    COPY_STRING str_null
    return

ML_finish_0
    COPY_STRING str_insp_fin
    return
ML_finish_1     ; ALL PASS????!!!!
    COPY_STRING str_runtime
    COPY_DEC1   runtime
    COPY_STRING str_s
    return

;*******************************************************************************
;   "copystring":  String Copy Function
;       Func:      Copy the string literal (null terminated) pointed by W
```

```
;                  to position pointed by FSR (indirect pointer)
;       Input:     W = address of the string literal wanted,
;                  FSR = adress of destination
;       Output:    FSR = adress of the null terminator of the copyed string
;       Affect:    W, STATUS, literal_addr
;*******************************************************************************
copystring
    movwf       literal_addr
copystring_loop
    movf        literal_addr, w
    call        literal
    movwf       INDF
    movf        INDF, f         ; test INDF(last char) == 0(NULL)
    btfsc       STATUS, Z
    return                      ; if end of string is reached (NULL)
    incf        FSR, f
    incf        literal_addr, f
    goto        copystring_loop

;*******************************************************************************
;   "copylayout":  Layout Row Translate & Copy Function
;       Func:      Translate a row in layout array into printable format,
;                  copy it to position pointed by FSR,
;                  and add a null termination after the copied charactor
;       Input:     W = raw number, FSR = adress of destination
;       Output:    FSR = adress of the null terminator after copied layout
;       Affect:
;*******************************************************************************
copylayout
    movwf       temp            ; raw number, later hold translated layout byte
    movlw       layout
    movf        temp, f         ; test if temp == 0
    btfsc       STATUS, Z
    goto        copylayout_next; temp == 0, starting position = layout
copylayout_startloop
    addlw       COLS
    decfsz      temp, f
    goto        copylayout_startloop
copylayout_next
    movwf       temp2           ; address of layout byte
    movf        FSR, w
    movwf       temp3           ; address of destination

    movlw       COLS
    movwf       temp4           ; colume counter
copylayout_charloop
    movf        temp2, w        ; get layout byte
    movwf       FSR
    movlw       char_layout     ; get translated layout byte address
    addwf       INDF, w
    call        literal         ; translate char
    movwf       temp            ; save the translated layout byte into temp
    movf        temp3, w        ; get destination
    movwf       FSR
    movf        temp, w         ; copy translated byte to destination
    movwf       INDF
    incf        temp2, f
    incf        temp3, f
    decfsz      temp4, f
    goto        copylayout_charloop

    movf        temp3, w
    movwf       FSR             ; FSR will now have the address after last byte
    movlw       NULL            ; add null terminator
```

```
    movwf       INDF
    return

;*******************************************************************************
;       "copydec":  Byte Display Conversion & Copy Function
;       Func:       Convert a number store in a byte to a printable decimal
;                   ASCII string with null termination and copy it to a position
;                   pointed by FSR
;       Input:      arg = number to be converted,
;                   arg2 = minimum number of digits displayed
;                   FSR = adress of destination
;       Output:     FSR = adress of the null terminator of the decimal display
;       Affect:     W, STATUS, arith_temp, arith_temp2
;*******************************************************************************
copydec
    movf        arg, w
    DIVLW       0x64            ; 100
    btfss       STATUS, Z       ; test if quotient is 0
    goto        copydec_copyhundreds   ; quotient <> 0, normal display
    movlw       0x03            ; quotient == 0, depends on arg2
    subwf       arg2, w
    btfss       STATUS, C       ; test if arg2 < 3
    goto        copydec_tens    ; arg2 < 3, skip 0 hundred
    movlw       0x00            ; arg2 >= 3, display 0
copydec_copyhundreds
    addlw       0x30            ; num+0x30 = its ASCII
    movwf       INDF
    incf        FSR, f
    movlw       0x03            ; hundreds already displayed, set arg2 to 3
    movwf       arg2            ; because all following digit shall be seen
copydec_tens
    movf        arg, w
    MODLW       0x64            ; 100
    DIVLW       0x0A            ; 10
    btfss       STATUS, Z       ; test if quotient is 0
    goto        copydec_copytens   ; quotient <> 0, normal display
    movlw       0x02            ; quotient == 0, depends on arg2
    subwf       arg2, w
    btfss       STATUS, C       ; test if arg2 < 2
    goto        copydec_ones    ; arg2 < 2, skip 0 tens
    movlw       0x00            ; arg2 >= 2, display 0
copydec_copytens
    addlw       0x30            ; num+0x30 = its ASCII
    movwf       INDF
    incf        FSR, f
;       movlw       0x02            ; tens already displayed, set arg2 to 2
;       movwf       arg2            ; because all following digit shall be seen
copydec_ones
    movf        arg, w
    MODLW       0x0A            ; 10
                                ; ones shall be displayed anyways
    addlw       0x30            ; num+0x30 = its ASCII
    movwf       INDF
    incf        FSR, f
copydec_ending                  ; write a null ending
    movlw       NULL
    movwf       INDF
    return

;*******************************************************************************
;   "writeline":   Write Displayable Line to LCD Function
;       Func:       Send LCDline string to LCD charactor by charactor
;       Input:      None (string prepared in LCDline)
;       Output:     None (to LCD)
```

```
;       Affect:     W, STATUS, FSR, delaytemp, delaycount
;       Runtime:    (3.2 + 216.0 * N) us, N = # of char, not including NULL
;*******************************************************************************
writeline
    MOVLF       FSR, LCDline
writeline_loop
    movf        INDF, w         ; test INDF(char pointer to the string)==0(NULL)
    btfsc       STATUS, Z
    return                      ; if end of string is reached (NULL)
    call        LCD_wt
    incf        FSR, f
    goto        writeline_loop

;*******************************************************************************
;       "Run":      Run Inspection Function
;       Func:       Control the entire process of inspection
;       Input:      None
;       Output:     st_year, st_month, st_day, st_hour, st_min, st_sec,
;                   end_hour, end_min, end_sec, runtime, cl_total, cl_pass,
;                   cl_fail, layout, smotor_dir, log!!!!
;       Affect:
;*******************************************************************************
run
    ; store sarting time
    MOVFF       st_year, rt_year
    MOVFF       st_month, rt_month
    MOVFF       st_day, rt_day
    MOVFF       st_hour, rt_hour
    MOVFF       st_min, rt_min
    MOVFF       st_sec, rt_sec

    ; reset all layouts
    MOVLF       FSR, layout
    MOVLF       temp, MAXPOS
run_clearlayoutloop
    clrf        INDF
    incf        FSR, f
    decfsz      temp, f
    goto        run_clearlayoutloop

    ; initialize variables
    movlw       layout
    movf        smotor_dir, f
    btfsc       STATUS, Z
    goto        run_init_smotornext; smotor_dir == 0, "row1" to "row4"
    addlw       MAXPOS          ; smotor_dir <> 0, "row4" to "row1"
    movwf       temp            ; temperary storage
    movlw       COLS
    subwf       temp, w         ; layout + MAXPOS - COLS, at last row


run_init_smotornext
    movwf       result_addr
    MOVLF       rowleft, ROWS
    clrf        cl_total
    clrf        cl_pass
    clrf        cl_fail

    bcf         INTCON, RBIF
    bsf         INTCON, RBIE    ; enable keypad interrept

    STORE_LIGHT light_bg
    PRINT_DOT                   ; !!!!
run_loop
```

```
    IRLED_ON
    STORE_LIGHT light_pos
    IRLED_OFF
    movf        result_addr, w
    CALC_POS    THD_IRLED
    btfsc       STATUS, Z
    goto        run_noextrarow ; return value == 0, no lights in this row

    ;PRESS_CL
    call        movedown
    STORE_LIGHT light_pos
    CALC_POS    THD_CL3LED
    call        moveup
    STORE_LIGHT light_cl
    PRINT_DOT                  ; !!!!
    ;PRESS_CL
    call        movedown
    call        moveup
    STORE_LIGHT light_off
    PRINT_DOT                  ; !!!!
    movf        result_addr, w
    call        calcfunc
    addlw       0x00
    btfsc       STATUS, Z
    goto        run_noextrarow ; return value == 0, no extra row

    ; return value <> 0, advance extra row
    ; advance to next RAM location
    movlw       COLS
    movf        smotor_dir, f
    btfss       STATUS, Z
    goto        run_nextRAM_reverse; smotor_dir<>0, "row4" to "row1", sub COLS
    addwf       result_addr, f ; smotor_dir == 0, "row1" to "row4", add COLS
    goto        run_nextRAM_next
run_nextRAM_reverse
    subwf       result_addr, f
run_nextRAM_next
    decf        rowleft, f
    btfsc       STATUS, Z
    goto        run_end        ; no row left, end run
    call        advancerow     ; advance to next machine location
    PRINT_DOT                  ; !!!!

run_noextrarow
    ; advance to next RAM location
    movlw       COLS
    movf        smotor_dir, f
    btfss       STATUS, Z
    goto        run_nextRAM_reverse2   ; smotor_dir <> 0, "row4" to "row1"
    addwf       result_addr, f ; smotor_dir == 0, "row1" to "row4", add COLS
    goto        run_nextRAM_next2
run_nextRAM_reverse2
    subwf       result_addr, f
run_nextRAM_next2
    decf        rowleft, f
    btfsc       STATUS, Z
    goto        run_end        ; no row left, end run
    call        advancerow     ; advance to next machine location
    PRINT_DOT                  ; !!!!
    goto        run_loop

run_end
    bcf         INTCON, RBIE   ; disable keypad interrept
    movlw       0x01           ; mask last bit
```

```
    xorwf       smotor_dir, f  ; logic NOT last digit, reverse direction

    ; store end time
    MOVFF       end_hour, rt_hour
    MOVFF       end_min, rt_min
    MOVFF       end_sec, rt_sec

    call        calcruntime
    call        writelog
    movwf       report_num
    return

;*******************************************************************************
;       "calcpos":  Row Position Calculation Function
;       Func:       Determine the existance of closet light at any positions in
;                   current row: result CL_POS_BIT = (light_pos >= THD_IRLED)
;       Input:      W = the starting address where the result will be stored,
;                   temp = address of the result, temp4 = threshold, light_pos[]
;       Output:     W = number of lights in current row, result bytes,
;                   STATUS is set according to W
;       Affect:     FSR, temp, temp2, temp3, temp4
;*******************************************************************************
calcpos
;   movwf       temp            ; address of the result
    clrf        temp2           ; colume number
    clrf        temp3           ; number of CLs in current row
calcpos_loop
    MOVFF       FSR, temp
    movlw       light_pos
    addwf       temp2, w
    movwf       FSR             ; get current light_pos
    movf        temp4, w
    subwf       INDF, w         ; light_pos - threshold
    btfss       STATUS, C
    goto        calcpos_next    ; light_pos < threshold, no light
    MOVFF       FSR, temp       ; light_pos >= threshold, has light
    bsf         INDF, CL_POS_BIT   ; set pos bit
    incf        temp3, f
calcpos_next
    incf        temp, f
    incf        temp2, f
    movlw       COLS
    subwf       temp2, w        ; temp2(col#) - COLS(max col#)
    btfss       STATUS, C
    goto        calcpos_loop    ; temp2(col#) < COLS(max col#)

    movf        temp3, w
    return

;*******************************************************************************
;   "calcfunc":     Row Functionality Calculation Function
;       Func:       Determine the functionality of closet light   at ANY positions
;                   (not only those maked pos) of current row, also determine
;                   whether next row can physically have any CL
;       Input:      W = the starting address where the result will be stored,
;                   light_bg[], light_cl[], light_off[], result bytes
;       Output:     W = whether the machine need to advance one more row
;                   (0 = advance one row; 1 = advance two rows),
;                   cl_total, cl_pass, cl_fail, layout[]
;       Affect:     STATUS, FSR, temp, temp2, temp3, temp4, temp5, temp6, temp7
;*******************************************************************************
calcfunc
    movwf       temp            ; address of the result
    clrf        temp2           ; colume number
```

```
    clrf        temp3           ; C0 = light_pos >= THD_IRLED
    clrf        temp4           ; C1 = light_cl >= THD_CL3LED
    clrf        temp5           ; C2 = light_off >= (lihgt_bg + THD_BG)
    clrf        temp6           ; number of CLs in current row
    clrf        temp7           ; temperaty storage
calcfunc_loop
    ; get C0 = CL_POS_BIT from calcpos
    MOVFF       FSR, temp
    movlw       FALSE
    btfsc       INDF, CL_POS_BIT
    movlw       TRUE            ; CL_POS_BIT set, C0 = true
    movwf       temp3
    ; get C1 = light_cl >= THD_CL3LED
    movlw       light_cl
    addwf       temp2, w
    movwf       FSR             ; get current light_cl
    movlw       THD_CL3LED
    subwf       INDF, w         ; light_cl - THD_CL3LED
    btfss       STATUS, C
    goto        calcfunc_C1false   ; light_pos < THD_IRLED, C1 = false
    movlw       TRUE            ; light_pos >= THD_IRLED, C1 = ture
    goto        calcfunc_C1next
calcfunc_C1false
    movlw       FALSE
calcfunc_C1next
    movwf       temp4
    ; get C2 = light_off >= (lihgt_bg + THD_BG)
    movlw       light_bg
    addwf       temp2, w
    movwf       FSR             ; get current light_bg
    movlw       THD_BG
    addwf       INDF, w         ; lihgt_bg + THD_BG
    movwf       temp7           ; temperally save
    movlw       light_off
    addwf       temp2, w
    movwf       FSR             ; get current light_off
    movf        temp7, w        ; put (lihgt_bg + THD_BG) back
    subwf       INDF, w         ; light_off - (lihgt_bg + THD_BG)
    btfss       STATUS, C
    goto        calcfunc_C2false   ; light_off < (lihgt_bg - THD_BG), C2 = false
    movlw       TRUE            ; light_off >= (lihgt_bg - THD_BG), C2 = true
    goto        calcfunc_C2next
calcfunc_C2false
    movlw       FALSE
calcfunc_C2next
    movwf       temp5

    ; determine the functionality of CL:
    MOVFF       FSR, temp
    clrf        INDF            ; reset layout byte
    ; CL_POS_BIT = C0 IOR C1 IOR C2
    movf        temp3, w        ; W = C0
    iorwf       temp4, w        ; W = C0 IOR C1
    iorwf       temp5, w        ; W = C0 IOR C1 IOR C2
    btfsc       STATUS, Z
    goto        calcfunc_POSfalse  ; W == 0, false
    bsf         INDF, CL_POS_BIT   ; W <> 0, true
    incf        cl_total, f
    incf        temp6, f
calcfunc_POSfalse
    ; CL_FN_BIT = (C1 AND (NOT C2)) IOR (C0 AND (NOT C1) AND C2)
    movf        temp5, w        ; W = C2
    xorlw       0xFF            ; W = NOT C2
    andwf       temp4, w        ; W = C1 AND (NOT C2)
```

```
    movwf       temp7           ; temperally save
    movf        temp4, w        ; W = C1
    xorlw       0xFF            ; W = NOT C1
    andwf       temp3, w        ; W = C0 AND (NOT C1)
    andwf       temp5, w        ; W = C0 AND (NOT C1) AND C2
    iorwf       temp7, w        ; W = (C1 AND(NOT C2))IOR(C0 AND(NOT C1)AND C2)
    btfsc       STATUS, Z
    goto        calcfunc_FNfalse    ; W == 0, false
    bsf         INDF, CL_FN_BIT     ; W <> 0, true
    incf        cl_pass, f
calcfunc_FNfalse
    ; CL_ERR_BIT = (C1 AND (NOT C0)) IOR (C2 AND (NOT C1))
    movf        temp4, w        ; W = C1
    xorlw       0xFF            ; W = NOT C1
    andwf       temp5, w        ; W = C2 AND (NOT C1)
    movwf       temp7           ; temperally save
    movf        temp3, w        ; W = C0
    xorlw       0xFF            ; W = NOT C0
    andwf       temp4, w        ; W = C1 AND (NOT C0)
    iorwf       temp7, w        ; W = (C1 AND (NOT C0)) IOR (C2 AND (NOT C1))
    btfss       STATUS, Z
    bsf         INDF, CL_ERR_BIT    ; W <> 0, true

    incf        temp, f
    incf        temp2, f
    movlw       COLS
    subwf       temp2, w        ; temp2(col#) - COLS(max col#)
    btfss       STATUS, C
    goto        calcfunc_loop   ; temp2(col#) < COLS(max col#)

    ; cl_fail = cl_total - cl_pass
    movf        cl_pass, w
    subwf       cl_total, w     ; cl_total - cl_pass
    movwf       cl_fail

    ; advance two rows if "CLs in this row" >= MAXCLINROW
    movlw       MAXCLINROW
    subwf       temp6, w        ; "CLs in this row" - MAXCLINROW
    btfss       STATUS, C
    retlw       0x00            ; "CLs in this row" < MAXCLINROW
    retlw       0x01            ; "CLs in this row" >= MAXCLINROW

;******************************************************************************
;   "calcruntime": Runtime Calculation Function
;       Func:       Calculate the running time of the run = endtime - starttime
;       Input:      st_min, st_sec, end_min, end_sec
;       Output:     runtime
;       Affect:     W, STATUS, temp
;******************************************************************************
calcruntime
    movf        st_min, w       ; temp = end_min - st_min
    subwf       end_min, w
    btfsc       STATUS, C
    goto        calcruntime_next1
    addlw       .60             ; borrow occur, +60min
calcruntime_next1
    movwf       temp

    movf        st_sec, w       ; W = end_sec - st_sec
    subwf       end_sec, w
    btfsc       STATUS, C
    goto        calcruntime_next2
    addlw       .60             ; borrow occur, +60sec
    decf        temp, f         ; -1min
```

```
calcruntime_next2

    movf        temp, f
    btfsc       STATUS, Z
    goto        calcruntime_next3  ; temp == 0, runtime = W
calcruntime_loop               ; runtime = W + .60*temp
    addlw       .60
    btfsc       STATUS, C
    goto        calcruntime_overflow   ; W>255 overflow
    decfsz      temp, f
    goto        calcruntime_loop
calcruntime_next3
    movwf       runtime
    return
calcruntime_overflow
    movlw       0xFF           ; !!!!
    movwf       runtime
    return

;*******************************************************************************
;   "lightsensor": Light Sensor Read, A/D Convert, and Store Function
;       Func:       Read in analog signal from light sensor, convert it to
;                   digital, and store the most significant 8-bit result to
;                   designated address, convert voltage reading to intensity,
;                   take average of LIGHTAVGX readings,
;                   repeat COLS times for a row
;       Input:      W = the staring address where the readings will be stored
;       Output:     readings store into designated bytes
;       Affect:     W, STATUS, FSR, ADCON0, ADRESH, ADRESL, temp, temp2, temp3
;                   delaytemp, delaycount, arith_temp, arith_temp2
;*******************************************************************************
lightsensor
    movwf       FSR
    clrf        temp           ; CHS (Channel Select) bits
    MOVLF       temp2, COLS    ; number of colume left
lightsensor_loop_col
    movlw       B'11000111'    ; mask CHS bits
    andwf       ADCON0, f      ; clear CHS bits
    movf        temp, w
    iorwf       ADCON0, f      ; set CHS bits

    ; avergae = (X1/n) + (X2/n) + ... + (Xn/n)
    MOVLF       temp3, LIGHTAVGX   ; count of sample light sensor reading
    clrf        INDF
lightsensor_loop_avg
    call        delay50us      ; require 2Tosc + Tacq = 28us
    bsf         ADCON0, GO
lightsensor_poll               ; about 40us
    btfsc       ADCON0, GO
    goto        lightsensor_poll
    movf        ADRESH, w      ; the most significant 8-bit result
    xorlw       0xFF           ; invert result, since 5V = 0 intensity!!!!
    DIVLW       LIGHTAVGX
    addwf       INDF, f
    decfsz      temp3, f
    goto        lightsensor_loop_avg

    movlw       B'00001000'
    addwf       temp, f        ; advance CHS
    incf        FSR, f
    decfsz      temp2, f
    goto        lightsensor_loop_col
    return
```

```
;***************************************************************************
;   "ledcontrol":  IR LEDs Control Function
;       Func:      Turn on or off the infrared LEDs according to W
;       Input:     W = turn on or off the LED (0=off or else=on)
;       Output:    None
;       Affect:    STATUS
;       Runtime:   3.2 us
;***************************************************************************
ledcontrol
    bcf        IRLED
    addlw      0x00
    btfss      STATUS, Z
    bsf        IRLED          ; if w <> 0, set IRLED
    return

;***************************************************************************
;   "movedown":    Arm Move Down Control Function
;       Func:      Order the test arm to move down to the Closet Lights
;       Input:     None
;       Output:    None
;       Affect:    VMOTOR_C0, VMOTOR_C1
;       Runtime:   ???? us
;***************************************************************************
movedown
    bsf        SMOTOR_EN      ; power s_motor to hold position
    bcf        VMOTOR_C1      ; to be safe
    bsf        VMOTOR_C0
;movedown_poll
;   btfsc      FB_BOT
;   goto       movedown_poll
    movlw      DCDOWNDELAY
    call       delayX100msm
    bcf        VMOTOR_C0
    bcf        SMOTOR_EN
    return

;***************************************************************************
;       "moveup":  Arm Move Up Control Function
;       Func:      Order the test arm to move up to its default position
;       Input:     None
;       Output:    None
;       Affect:    VMOTOR_C0, VMOTOR_C1
;       Runtime:   ???? us
;***************************************************************************
moveup
    bsf        SMOTOR_EN      ; power s_motor to hold position
    bcf        VMOTOR_C0      ; to be safe
    bsf        VMOTOR_C1
moveup_poll
    btfsc      FB_TOP
    goto       moveup_poll
    bcf        VMOTOR_C1
    bcf        SMOTOR_EN
    return

;***************************************************************************
;   "advancerow":  Arm Advance-to-Next-Row Control Function
;       Func:      Order the test arm to the next(determined by smotor_dir) row
;       Input:     smotor_dir = direction of stpper motor
;                  (0 = "Row 1" to "Row 4"; 1 = "Row 4" to "Row 1")
;       Output:    None
;       Affect:    W, STATUS, temp, SMOTOR_EN, SMOTOR_S0, SMOTOR_S1, SMOTOR_S2,
;                  SMOTOR_S3, delaytemp, delaycount, delaytemp2, delaycount2,
;                  delaytemp3, delaycount3
```

```
;*******************************************************************************
advancerow
    bcf         SMOTOR_S1           ; state init
    bsf         SMOTOR_S3
    bsf         SMOTOR_EN           ; start motor
    movlw       SMOTOR_STEPS
    movwf       temp
    movlw       SMOTOR_SPD_F        ; note: w must keep its value until very end
    movf        smotor_dir, f
    btfss       STATUS, Z
    goto        advancerow_backwardloop
advancerow_forwardloop              ; smotor_dir == 0, "Row 1" to "Row 4"
    bcf         SMOTOR_S0
    bsf         SMOTOR_S2
    btfss       FB_ROW4
    goto        advancerow_end
    call        delayX5msm
    bcf         SMOTOR_S3
    bsf         SMOTOR_S1
    btfss       FB_ROW4
    goto        advancerow_end
    call        delayX5msm
    bcf         SMOTOR_S2
    bsf         SMOTOR_S0
    btfss       FB_ROW4
    goto        advancerow_end
    call        delayX5msm
    bcf         SMOTOR_S1
    bsf         SMOTOR_S3
    btfss       FB_ROW4
    goto        advancerow_end
    call        delayX5msm
    decfsz      temp, f
    goto        advancerow_forwardloop
    goto        advancerow_end
advancerow_backwardloop                 ; smotor_dir <> 0, "Row 4" to "Row 1"
    bcf         SMOTOR_S3
    bsf         SMOTOR_S1
    call        delayX5msm
    bcf         SMOTOR_S0
    bsf         SMOTOR_S2
    call        delayX5msm
    bcf         SMOTOR_S1
    bsf         SMOTOR_S3
    call        delayX5msm
    bcf         SMOTOR_S2
    bsf         SMOTOR_S0
    call        delayX5msm
    decfsz      temp, f
    goto        advancerow_backwardloop
advancerow_end
    bcf         SMOTOR_EN
    return

;*******************************************************************************
;   "writelog":     Log Write Function
;       Func:       Write the result of current run to log
;       Input:      st_year, st_month, st_day, st_hour, st_min, st_sec,
;                   runtime, layout[], log_total, log_next
;       Output:     W = current log index, log_total, log_next, Log Entry
;       Affect:     STATUS, FSR, temp, temp2, temp3, temp4,
;                   arith_temp, arith_temp2
;*******************************************************************************
writelog
```

```
    ; starting address (indirect) offset = (log_next / MAXLOGBANK) * 0x80 + 0x10
    movf        log_next, w
    DIVLW       MAXLOGBANK
    MULLW       0x80
    addlw       0x10
    movwf       temp            ; save in temp

    ; starting address (indirect) = (log_next MOD MAXLOGBANK)*LOGLENGTH + offset
    movf        log_next, w
    MODLW       MAXLOGBANK      ; W = log_next mod MAXLOGBANK
    MULLW       LOGLENGTH
    addwf       temp, w

    ; store easy-access variables
    bsf         STATUS, IRP     ; BANK2&3 indirect access
    movwf       FSR             ; log addr + 0
    MOVFF       INDF, runtime
    incf        FSR, f          ; log addr + 1
    swapf       st_year, w
    addwf       st_month, w
    movwf       INDF
    incf        FSR, f          ; log addr + 2
    MOVFF       INDF, st_day
    incf        FSR, f          ; log addr + 3
    MOVFF       INDF, st_hour
    incf        FSR, f          ; log addr + 4
    MOVFF       INDF, st_min
    incf        FSR, f          ; log addr + 5
    MOVFF       INDF, st_sec
    incf        FSR, f          ; log addr + 6

    ; store layout
    MOVFF       temp, FSR       ; the address of log entry (start at + 6)
    MOVLF       temp2, layout   ; address of layout bytes
    MOVLF       temp3, MAXPOS   ; position counter
writelog_layoutloop             ; copy layout bytes in pairs
    ; write upper ribble
    MOVFF       FSR, temp2      ; get first layout byte
    bcf         STATUS, IRP     ; BANK0&1 indirect access
    swapf       INDF, w
    movwf       temp4           ; temp4 hold the swaped layout byte temperally
    MOVFF       FSR, temp       ; log entry
    bsf         STATUS, IRP     ; BANK2&3 indirect access
    MOVFF       INDF, temp4
    incf        temp2, f
    decf        temp3, f
    btfsc       STATUS, Z
    goto        writelog_layoutnext

    ; write lower ribble
    MOVFF       FSR, temp2      ; get second layout byte
    bcf         STATUS, IRP     ; BANK0&1 indirect access
    MOVFF       temp4, INDF     ; temp4 hold the layout byte temperally
    MOVFF       FSR, temp       ; log entry
    bsf         STATUS, IRP     ; BANK2&3 indirect access
    movf        temp4, w
    addwf       INDF, f
    incf        temp2, f
    incf        temp, f
    decfsz      temp3, f
    goto        writelog_layoutloop

writelog_layoutnext
    ; calculate new log_next = (log_next + 1) MOD MAXLOG
```

```
    MOVFF       temp, log_next ; save current log index in temp
    incf        log_next, f
    movlw       MAXLOG          ; test if max log reached
    subwf       log_next, w     ; log_next + 1 - MAXLOG
    btfss       STATUS, C
    addlw       MAXLOG          ; log_next + 1 < MAXLOG, add MAXLOG back
    movwf       log_next        ; log_next = (log_next + 1) MOD MAXLOG

    ; calculate new log_total
    movlw       MAXLOG
    subwf       log_total, w    ; log_total - MAXLOG
    btfss       STATUS, C
    incf        log_total, f    ; log_total < MAXLOG, increase total number

    movf        temp, w         ; resume current log entry index
    bcf         STATUS, IRP     ; back to BANK0&1 indirect access
    return

;*******************************************************************************
;       "readlog":  Log Read Function
;       Func:       Read the log of specified run
;       Input:      W = index of wanted report
;       Output:     st_year, st_month, st_day, st_hour, st_min, st_sec,
;                   end_hour, end_min, end_sec, runtime, cl_total, cl_pass,
;                   cl_fail, layout[]
;       Affect:     W, STATUS, FSR, temp, temp2, temp3, temp4,
;                   arith_temp, arith_temp2
;*******************************************************************************
readlog
    movwf       temp2           ; save the index in temp2
    clrf        cl_total
    clrf        cl_pass
    ; starting address (indirect) offset = (index / MAXLOGBANK) * 0x80 + 0x10
    DIVLW       MAXLOGBANK
    MULLW       0x80
    addlw       0x10
    movwf       temp            ; save in temp

    ; starting address (indirect) = (index MOD MAXLOGBANK)*LOGLENGTH + offset
    movf        temp2, w        ; resume index
    MODLW       MAXLOGBANK      ; W = index mod MAXLOGBANK
    MULLW       LOGLENGTH
    addwf       temp, w

    ; read easy-access variables
    bsf         STATUS, IRP     ; BANK2&3 indirect access
    movwf       FSR             ; log addr + 0
    MOVFF       runtime, INDF
    incf        FSR, f          ; log addr + 1
    swapf       INDF, w         ; swap upper 4 bits and lower 4 bits
    andlw       0x0F            ; mask the lower 4 bits only
    movwf       st_year
    movf        INDF, w
    andlw       0x0F            ; mask the lower 4 bits only
    movwf       st_month
    incf        FSR, f          ; log addr + 2
    MOVFF       st_day, INDF
    incf        FSR, f          ; log addr + 3
    MOVFF       st_hour, INDF
    incf        FSR, f          ; log addr + 4
    MOVFF       st_min, INDF
    incf        FSR, f          ; log addr + 5
    MOVFF       st_sec, INDF
    incf        FSR, f          ; log addr + 6
```

```
    ; read layout[], cl_total and cl_pass
    MOVFF       temp, FSR       ; the address of log entry (start at + 6)
    MOVLF       temp2, layout   ; address of layout bytes
    MOVLF       temp3, MAXPOS   ; position counter
readlog_layoutloop              ; read layout bytes in pairs
    ; read upper ribble
    MOVFF       FSR, temp       ; log entry
    bsf         STATUS, IRP     ; BANK2&3 indirect access
    swapf       INDF, w
    andlw       0x0F            ; mask the lower 4 bits only (not 3 bits!!!!)
    movwf       temp4           ; temp4 hold the swaped layout byte temperally
    MOVFF       FSR, temp2      ; get first layout byte
    bcf         STATUS, IRP     ; BANK0&1 indirect access
    MOVFF       INDF, temp4
    btfsc       INDF, CL_POS_BIT
    incf        cl_total, f
    btfsc       INDF, CL_FN_BIT
    incf        cl_pass, f
    incf        temp2, f
    decf        temp3, f
    btfsc       STATUS, Z
    goto        readlog_layoutnext

    ; read lower ribble
    MOVFF       FSR, temp       ; log entry
    bsf         STATUS, IRP     ; BANK2&3 indirect access
    movf        INDF, w
    andlw       0x0F            ; mask the lower 4 bits only (not 3 bits!!!!)
    movwf       temp4           ; temp4 hold the layout byte temperally
    MOVFF       FSR, temp2      ; get second layout byte
    bcf         STATUS, IRP     ; BANK0&1 indirect access
    MOVFF       INDF, temp4
    btfsc       INDF, CL_POS_BIT
    incf        cl_total, f
    btfsc       INDF, CL_FN_BIT
    incf        cl_pass, f
    incf        temp2, f
    incf        temp, f
    decfsz      temp3, f
    goto        readlog_layoutloop

readlog_layoutnext
    ; calculate end time
    MOVFF       end_hour, st_hour
    MOVFF       end_min, st_min
    MOVFF       end_sec, st_sec
    movf        runtime, w
    DIVLW       .60
    addwf       end_min, f      ; end_min = st_min + runtime / .60
    movf        runtime, w
    MODLW       .60
    addwf       end_sec, f      ; end_sec = st_sec + runtime MOD .60
    movlw       .60             ; test if 60s
    subwf       end_sec, w      ; w = end_sec - 60
    btfss       STATUS, C
    goto        readlog_endsec_next; end_sec < 60, end_sec OK
    movlw       .60             ; end_sec >= 60, +1min, -60s
    subwf       end_sec, f
    incf        end_min, f
readlog_endsec_next
    movlw       .60             ; test if 60min
    subwf       end_min, w
    btfss       STATUS, C
```

```
    goto        readlog_next    ; end_min < 60, end_min OK, end_hour should OK
    movlw       .60             ; end_min >= 60, +1h, -60min
    subwf       end_min, f
    incf        end_hour, f
    movlw       .24             ; test if 24h
    subwf       end_hour, w
    btfss       STATUS, C
    goto        readlog_next
    movlw       .24             ; >24h, -24h
    subwf       end_hour, f
readlog_next

    ; cl_fail = cl_total - cl_pass
    movf        cl_pass, w
    subwf       cl_total, w
    movwf       cl_fail

    bcf         STATUS, IRP     ; back to BANK0&1 indirect access
    return

;*******************************************************************************
;       "addsec":   Add One Second Function
;       Func:       Add one second to the second byte of a timer,!!!!!!!!!!!!!
;                   increase minute and hour bytes if necessary,
;                   carry out if day increment occur
;       Input:      W = Address of the second byte, min byte and hour byte
;                   should be at address W-1 and W-2, respectively
;       Output:     set STATUS C for day carry out
;       Affect:     W, FSR
;       Runtime:    3.6us/6.4us/9.2us/10.0us
;*******************************************************************************
addsec
    movwf       FSR
    incf        INDF, f         ; +1sec
    movlw       .60
    subwf       INDF, w         ; W = INDF - 60, test if 60s
    btfss       STATUS, C
    return                      ; INDF < 60, C==0, function done
    clrf        INDF            ; >=60s, clear sec
    decf        FSR, f          ; now FSR has address of min byte
    incf        INDF, f         ; +1min
    movlw       .60
    subwf       INDF, w         ; W = INDF - 60, test if 60min
    btfss       STATUS, C
    return                      ; INDF < 60, C==0, function done
    clrf        INDF            ; >=60min, clear min
    decf        FSR, f          ; now FSR has address of hour byte
    incf        INDF, f         ; +1hour
    movlw       .24
    subwf       INDF, w         ; W = INDF - 24, test if 24hour
    btfss       STATUS, C
    return                      ; INDF < 24, C==0, function done
    clrf        INDF            ; >=24h, clear hour
    return                      ; C has been set

;*******************************************************************************
;       "addday":   Add One Day Function
;       Func:       Add one day to the day byte of a timer,
;                   increase month and year bytes if necessary, !!!!unfinish
;       Input:      W = Address of the day byte, month byte and year byte
;                   should be at address W-1 and W-2, respectively
;       Output:     None
;       Affect:     W, STATUS, FSR
;       Runtime:    3.6us/6.4us/8.0us
```

```
;*****************************************************************************
addday
    movwf       FSR
    incf        INDF, f         ; +1day
    movlw       .31             ; assume 1 month always= 30 days !!!!
    subwf       INDF, w         ; W = INDF - 31, test if over 30days
    btfss       STATUS, C
    return                      ; INDF < 31, C==0, function done
    clrf        INDF            ; >=31s, clear day
    incf        INDF            ; day default at 1
    decf        FSR, f          ; now FSR has address of month byte
    incf        INDF, f         ; +1month
    movlw       .13
    subwf       INDF, w         ; W = INDF - 13, test if over 12month
    btfss       STATUS, C
    return                      ; INDF < 13, C==0, function done
    clrf        INDF            ; >=13month, clear month
    incf        INDF            ; month default at 1
    decf        FSR, f          ; now FSR has address of year byte
    incf        INDF, f         ; +1year
    return

;*****************************************************************************
;       "divfn":    Byte Integer Division Function
;       Func:       Devide temp by W and store result in W, not efficient
;       Input:      Temp = Dividend, W = Dividor
;       Output:     W = Quotient, arith_temp = reminder,
;                   set STATUS Z for zero quotient, C for error
;       Affect:     arith_temp2
;*****************************************************************************
divfn
    addlw       0x00
    bsf         STATUS, C       ; set carry in case of error
    btfsc       STATUS, Z       ; if zero
    return                      ;   return (error C,Z)
    clrf        arith_temp2
divfn_loop
    subwf       arith_temp, f
    btfss       STATUS, C
    goto        divfn_next
    incf        arith_temp2, f
    goto        divfn_loop
divfn_next
    addwf       arith_temp, f
    movf        arith_temp2, w
    return

;*****************************************************************************
;       "mulfn":    Byte Integer Multiplication Function
;       Func:       Multiple W by temp and return result in W, not efficient
;       Input:      Temp, W
;       Output:     W = Result, set STATUS Z for zero, C for overflow
;       Affect:     arith_temp, arith_temp2
;*****************************************************************************
mulfn
    bcf         STATUS, C       ; clr C bit for arith_temp == 0
    movwf       arith_temp2         ; store W in arith_temp2
    movlw       0x00            ; W = 0 + arith_tempp * arith_temp2
    movf        arith_temp, f
    btfsc       STATUS, Z
    return                      ; arith_temp == 0, return 0
mulfn_loop
    addwf       arith_temp2, w
    btfsc       STATUS, C
```

```
    goto        mulfn_overflow ; W>255 overflow
    decfsz      arith_temp, f
    goto        mulfn_loop
mulfn_overflow
    return

    end
```

```
; for LCD series 162A

    #include <p16f877.inc>
    #include <common.inc>

    extern  delay50us, delay5ms, delayX5msm, delay1sl

    ;Only these functions are visible to other asm files
    global  LCD_init, LCD_clear, LCD_line2, LCD_out, LCD_wt

    ;Declare unbanked variables (at 0x70 and on)
    UDATA_SHR
lcd_tmp    res 1

;******************************************************************************
;       "ClkLCD":  LCD Enable Click Function
;       Func:      Pulse the E line low
;       Input:     None
;       Output:    None
;       Affect:    STATUS, delaytemp, delaycount
;       Runtime:   102.4 us
;******************************************************************************
ClkLCD     macro
    call    delay50us
    bcf     LCD_E
    call    delay50us
    bsf     LCD_E
    endm

;******************************************************************************
;       "MovMSB":  Most Significant Bits Move Function
;       Func:      Move MSB of W to LCD_PORT<4:7>, without disturbing LSB
;       Input:     W
;       Output:    LCD_PORT
;       Affect:    STATUS
;       Runtime:   3.2 us
;******************************************************************************
MovMSB     macro
    andlw   0xF0
    iorwf   LCD_PORT,f
    iorlw   0x0F
    andwf   LCD_PORT,f
    endm

    code
;******************************************************************************
;   "LCD_init":    LCD Initialization Function
;       Func:      Initialize LCD after reset
;       Input:     None
;       Output:    None
;       Affect:    delaytemp, delaycount, delaytemp2, delaycount2,
;       Runtime:   37,074.8 us
;******************************************************************************
LCD_init
    BANK0
    bsf     LCD_E          ; E default high

    ; Wait for more than 15ms after VDD rises to 4.5V (20ms)
    call    delay5ms
    call    delay5ms
    call    delay5ms
    call    delay5ms

    ;Ensure 8-bit mode first (no way to immediately guarantee 4-bit mode)
```

```
    ; -> Send b'0011' 3 times
    bcf     LCD_RS          ; Instruction mode
    movlw   B'00110000'
    MovMSB
    ; Finish last 4-bit send (if reset occurred in middle of a send)
    ClkLCD
    call    delay5ms        ; Wait for more than max instruction time 4.1ms(5ms)
    ClkLCD          ; Assuming 4-bit mode, set 8-bit mode
    call    delay50us       ; Wait for more than 100us
    call    delay50us
    ClkLCD

    ; (note: if it's in 8-bit mode already, it will stay in 8-bit mode)
    ; Now that we know for sure it's in 8-bit mode, set 4-bit mode.
    movlw   B'00100000'
    MovMSB
    ClkLCD

    ; Give LCD init instructions
    movlw   B'00101000'     ; 4 bits, 2 lines,5X8 dot
    call    LCD_wt
    movlw   B'00001111'     ; display on,cursor,blink
    call    LCD_wt
    movlw   B'00000110'     ; Increment,no shift
    call    LCD_wt
    ; Ready to display characters
    call    LCD_clear
    bsf     LCD_RS          ; Character mode
    return

;*******************************************************************************
;   "LCD_clear":    LCD Clear Function
;       Func:       Clear the LCD display using clear command
;       Input:      None
;       Output:     None
;       Affect:     W, STATUS, delaytemp, delaycount, lcd_temp
;       Runtime:    2,264.8 us
;*******************************************************************************
LCD_clear
    bcf     LCD_RS          ;Instruction mode
    movlw   B'00000001'
    call    LCD_wt
    ; expected excution time: 1.64ms (~2ms)
    movlw   .40
    movwf   lcd_tmp
    call    delay50us
    decfsz  lcd_tmp, f
    goto    $-2
    bsf     LCD_RS          ; Character mode
    return

;*******************************************************************************
;   "LCD_line2":    LCD Move to Seocnd Line Function
;       Func:       Move the LCD cursor to second Line
;       Input:      None
;       Output:     None
;       Affect:     W, STATUS, delaytemp, delaycount
;       Runtime:    216.4 us
;*******************************************************************************
LCD_line2
    bcf     LCD_RS          ; Instruction mode
    movlw   B'11000000'     ; shift position to 40h : second line
    call    LCD_wt
    bsf     LCD_RS          ; Character mode
```

```
    return

;*******************************************************************************
;       "LCD_out":  LCD Move Out of Screen Function
;       Func:       Move the LCD cursor out of screen (to 50h)
;       Input:      None
;       Output:     None
;       Affect:     W, STATUS, delaytemp, delaycount
;       Runtime:    216.4 us
;*******************************************************************************
LCD_out
    bcf     LCD_RS          ; Instruction mode
    movlw   B'11010000'     ; shift position to 50h : out of screen
    call    LCD_wt
    bsf     LCD_RS          ; Character mode
    return

;*******************************************************************************
;       "LCD_wt":   LCD Write Function
;       Func:       Clock MSB and LSB of W to LCD_PORT<7:4> in two cycles
;       Input:      W
;       Output:     None
;       Affect:     W, STATUS, delaytemp, delaycount
;       Runtime:    213.6 us
;*******************************************************************************
LCD_wt
    movwf   lcd_tmp         ; store original value
    MovMSB          ; move MSB to PORTD
    ClkLCD
    swapf   lcd_tmp,w       ; Swap LSB of value into MSB of W
    MovMSB          ; move to PORTD
    ClkLCD
    return

    end
```

```
    #include <p16f877.inc>
    #include <common.inc>

    cblock      0xA0
    delaytemp
    delaycount
    delaytemp2
    delaycount2
    delaytemp3
    delaycount3
    endc

    code
    global  delay50us, delay5ms, delayX5msm, delay100ms, delayX100msm, delay1sl

;*******************************************************************************
;   "dalay50us":   50us Delay Function
;       Func:      Precisely dalay 50 microsecond for 10Mz (125cycles*4)
;                  including the time this function being called
;       Input:     None
;       Output:    None
;       Affect:    STATUS, delaytemp, delaycount
;       Runtime:   50.0 us
;*******************************************************************************
delay50us
    ; call function cost 2 cycles
    ;BANK1                      ; 2 cycles
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    movwf       delaytemp       ; protect the data in w, 1 cycle
    movlw       0x26            ; 38, 1 cycle
    movwf       delaycount      ; 1 cycle
delay50us_loop                  ; ((1+2)*38)-1 = 113 cycles
    decfsz      delaycount, f
    goto        delay50us_loop
    movf        delaytemp, w    ; resume w, 1 cycle
    ;BANK0                      ; 2 cycles
    bcf         STATUS, RP0
    nop
    return                      ; 2 cycles

;*******************************************************************************
;   "dalay5ms":    5ms Delay Function
;       Func:      Precisely dalay 5 millisecond for 10Mz (100*125cycles*4)
;                  including the time this function being called
;       Input:     None
;       Output:    None
;       Affect:    STATUS, delaytemp, delaycount, delaytemp2, delaycount2
;       Runtime:   5,000.0 us
;*******************************************************************************
delay5ms
    ; call function cost 2 cycles
    ;BANK1                      ; 2 cycles
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    movwf       delaytemp2      ; protect the data in w, 1 cycle
    ; 5 cycles to this point

    movlw       0x60            ; 96, 1 cycle
    movwf       delaycount2     ; 1 cycle
delay5ms_loop                   ; (130*96)-1 = 12479 cycles

    ;call       delay50us       ; (125 cycles)
    nop
```

```
    ;BANK1                      ; 2 cycles
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    movwf       delaytemp       ; protect the data in w, 1 cycle
    movlw       0x27            ; 39, 1 cycle
    movwf       delaycount      ; 1 cycle
delay5ms_50us_loop              ; ((1+2)*39)-1 = 116 cycles
    decfsz      delaycount, f
    goto        delay5ms_50us_loop
    movf        delaytemp, w    ; resume w, 1 cycle
    ;BANK0                      ; 2 cycles
    bcf         STATUS, RP0
    nop

    ;BANK1                      ; delay50us will reset Bank to 0, (2 cycles)
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    decfsz      delaycount2, f ; (1(2) cycle)
    goto        delay5ms_loop  ; (2 cycle)
    ; 12486 cycles to this point

    movlw       0x02            ; 2, 1 cycle
    movwf       delaycount2     ; 1 cycle
delay5ms_loop2                  ; (3*2)-1 = 5 cycles
    decfsz      delaycount2, f
    goto        delay5ms_loop2
    ; 12493 cycles to this point

    nop                         ; 1 cycle
    nop                         ; 1 cycle
    movf        delaytemp2, w   ; resume w, 1 cycle
    ;BANK0                      ; 2 cycles
    bcf         STATUS, RP0
    nop
    return                      ; 2 cycles

;********************************************************************************
;   "dalayX5msm":  Multiple of 5ms More Delay Function (less than 0.1% error)
;       Func:       Delay slightly more than 5*W millisecond for 10Mz
;                   precisely dalay (5.002*W+0.0032) ms ((12,505*W+8) cycles)
;                   including the time this function being called
;       Input:      W = numbers of 5ms to delay
;       Output:     None
;       Affect:     STATUS, delaytemp, delaycount, delaytemp2, delaycount2,
;                   delaycount3
;       Runtime:    (5,002.0*W + 3.2) us
;********************************************************************************
delayX5msm
    ; call function cost 2 cycles
    ;BANK1                      ; 2 cycles
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    movwf       delaycount3     ; 1 cycle

delayX5msm_loop                 ; (12505*W-1) cycles
    call        delay5ms
    ;BANK1                      ; delay5ms will reset Bank to 0
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    decfsz      delaycount3, f
    goto        delayX5msm_loop

    ;BANK0                      ; 2 cycles
    bcf         STATUS, RP0
```

```
    nop
    return                      ; 2 cycles

;*******************************************************************************
;   "dalay100ms":  100ms Delay Function
;       Func:      Precisely dalay 100 millisecond for 10Mz (250,004cycles)
;                  including the time this function being called
;       Input:     None
;       Output:    None
;       Affect:    STATUS, delaytemp, delaycount, delaytemp2, delaycount2
;       Runtime:   100,001.6 us
;*******************************************************************************
delay100ms
    ; call function cost 2 cycles
    ;BANK1                      ; 2 cycles
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    movwf       delaytemp2      ; protect the data in w, 1 cycle
    ; 5 cycles to this point

    movlw       0xF9            ; 249, 1 cycle
    movwf       delaycount2     ; 1 cycle
delay100ms_loop                 ; (1004*250)-1 = 249,995 cycles
    movlw       0xFA            ; 250, 1 cycle
    movwf       delaycount      ; 1 cycle
delay100ms_loop2                ; (4*250)-1 = 999 cycles
    nop
    decfsz      delaycount, f
    goto        delay100ms_loop2

    decfsz      delaycount2, f ; (1(2) cycle)
    goto        delay100ms_loop; (2 cycle)
    ; 250,000 cycles to this point

    movf        delaytemp2, w   ; resume w, 1 cycle
    ;BANK0                      ; 1 cycles
    bcf         STATUS, RP0
    return                      ; 2 cycles

;*******************************************************************************
;   "dalayX100msm": Multiple of 100ms More Delay Function(less than 0.01% error)
;       Func:      Delay slightly more than 100*W millisecond for 10Mz
;                  precisely dalay (0.100002*W+0.0000032) s
;                  ((250,009*W+8) cycles)
;                  including the time this function being called
;       Input:     W = numbers of 100ms to delay
;       Output:    None
;       Affect:    STATUS, delaytemp, delaycount, delaytemp2, delaycount2,
;                  delaycount3
;       Runtime:   (100,003.6*W + 3.2) us
;*******************************************************************************
delayX100msm
    ; call function cost 2 cycles
    ;BANK1                      ; 2 cycles
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    movwf       delaycount3     ; 1 cycle

delayX100msm_loop               ; (250,009*W-1) cycles
    call        delay100ms
    ;BANK1                      ; delay100ms will reset Bank to 0
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    decfsz      delaycount3, f
```

```
    goto        delayX100msm_loop

    ;BANK0                      ; 2 cycles
    bcf         STATUS, RP0
    nop
    return                      ; 2 cycles

;******************************************************************************
;   "dalay1sl":     1s Less Delay Function
;       Func:       Dalay slightly less than 1 second for 10Mz,
;                   precisely delay 999.0428ms (2,497,607 cycles)
;                   including the time this function being called
;       Input:      None
;       Output:     None
;       Affect:     STATUS, delaytemp, delaycount, delaytemp2, delaycount2,
;                   delaytemp3, delaycount3
;       Runtime:    999,042.8 us
;******************************************************************************
delay1sl
    ; call function cost 2 cycles
    ;BANK1                      ; 2 cycles
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    movwf       delaytemp3      ; protect the data in w, 1 cycle
    ; 5 cycles to this point

    movlw       0xC7            ; 199, 1 cycle
    movwf       delaycount3     ; 1 cycle
delay1sl_loop                   ; (12505*199)-1 = 2,488,494 cycles
    call        delay5ms        ; (12500 cycles)
    ;BANK1                      ; delay5ms will reset Bank to 0, (2 cycles)
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    decfsz      delaycount3, f  ; (1(2) cycle)
    goto        delay1sl_loop   ; (2 cycle)
    ; 2,488,501 cycles to this point

    movlw       0x46            ; 70, 1 cycle
    movwf       delaycount3     ; 1 cycle
delay1sl_loop2                  ; (130*69)-1 = 9099 cycles
    call        delay50us       ; (125 cycles)
    ;BANK1                      ; delay50us will reset Bank to 0, (2 cycles)
    bcf         STATUS, RP1
    bsf         STATUS, RP0
    decfsz      delaycount3, f
    goto        delay1sl_loop2
    ; 2,497,602 cycles to this point

    movf        delaytemp3, w   ; resume w, 1 cycle
    ;BANK0                      ; 2 cycles
    bcf         STATUS, RP0
    nop
    return                      ; 2 cycles

    end
```

```
    #define DEBUG

    #define NULL            0x00    ; must not change
    #define FALSE           0x00    ; must not change
    #define TRUE            0xFF    ; must not change
    #define KEYPAD_DA       PORTB, 1
    #define SMOTOR_S0       PORTD, 0
    #define SMOTOR_S1       PORTD, 1
    #define SMOTOR_S2       PORTE, 0
    #define SMOTOR_S3       PORTE, 1
    #define SMOTOR_EN       PORTE, 2
    #define VMOTOR_C0       PORTC, 0
    #define VMOTOR_C1       PORTC, 1
    #define IRLED           PORTC, 2
    #define FB_TOP          PORTC, 5
;   #define FB_BOT          PORTC, 6
    #define FB_ROW4         PORTC, 7
    #define LCD_RS          PORTD, 2
    #define LCD_E           PORTD, 3
    #define LCD_PORT        PORTD
    #define CL_POS_BIT      0
    #define CL_FN_BIT       1
    #define CL_ERR_BIT      2
    #define  STOP_BUT       0x0B       ; stop buttom
    #define PHASE_HDINIT    0       ; Phase 0: Hardware Initialization/Reset
    #define PHASE_RTCINIT   1       ; Phase 1: RTC Initialization/Reset
    #define PHASE_REALTIME  2       ; Phase 2: Real Time
    #define PHASE_NOREPORT  3       ; Phase 3: No Report
    #define PHASE_REPORT    4       ; Phase 4: Report Review
    #define PHASE_RUN       5       ; Phase 5: Inspection Run
    #define PHASE_FINISH    6       ; Phase 6: Inspection Finish
    ; pull-up diable, Timer0 internal clk, pos edge, 1:64
    #define INITVAL_OPTREG B'10000101'
    #define INITVAL_TRISA  B'00101111'
    #define INITVAL_TRISB  B'11110010'
    #define INITVAL_TRISC  B'10100000' ;B'11100000'!!!!
    #define INITVAL_TRISD  B'00000000'
    #define INITVAL_TRISE  B'00000000'
    #define INITVAL_ADCON0 B'10000001' ; 32Tosc, AN0, ADC on
    #define INITVAL_ADCON1 B'00000010' ; left justified, 5 channel, 0 Vref

;****************CONSTANTS***************************************************
MAXLINE         EQU .9      ; (max "line_num" - 1) in report phase
MAXPOS          EQU .20     ; total positions = ROWS*COLS
ROWS            EQU .4      ; total 4 rows to move
COLS            EQU .5      ; total 5 cols in a row
MAXCLINROW      EQU .3      ; max number of CLs possible in a row
DCDOWNDELAY     EQU .30     ; * 0.1s == elapse time for DC motor to drive down
SMOTOR_STEPS    EQU .28     ; * 4 == # of steps that stepper motor need to turn
SMOTOR_SPD_F    EQU .6      ; speed fact: speed = 1 step / (SMOTOR_SPD_F * 5ms)
THD_CL3LED      EQU .80     ; threshold to distinguish between 2 LEDs and 3LEDS
THD_IRLED       EQU .28     ; threshold for infrared reflected light
THD_BG          EQU .5      ; threshold for background variation (add to light_bg)
MAXLOG          EQU .12     ; max number of log
MAXLOGBANK      EQU .6      ; max number of log per bank = MAXLOG/2
LOGLENGTH       EQU .16     ; length of each log entry
LIGHTAVGX       EQU .4      ; number of sample light sensor reading to be averaged

;****************MACROS******************************************************
BANK0       macro
    bcf         STATUS, RP0
    endm
```

```
BANK1       macro
    bsf         STATUS, RP0
    endm

BANK00      macro
    bcf         STATUS, IRP
    bcf         STATUS, RP1
    bcf         STATUS, RP0
    endm

;BANK02     macro
;   bsf         STATUS, IRP
;   bsf         STATUS, RP1
;   bcf         STATUS, RP0
;   endm
;
;BANK03     macro
;   bsf         STATUS, IRP
;   bsf         STATUS, RP1
;   bsf         STATUS, RP0
;   endm

; Affect W!
MOVFF       macro   dest, orig
    movf        orig, w
    movwf       dest
    endm

; Affect W!
MOVLF       macro   dest, literal
    movlw       literal
    movwf       dest
    endm
```