RIO-MP USER'S GUIDE

Version 1.2
November 2002

Rigel Corporation
P.O. Box 90040, Gainesville, Florida
(352) 384-3766
www.rigelcorp.com
tech@rigelcorp.com

HARDWARE WARRANTY

Limited Warranty. Rigel Corporation warrants, for a period of sixty (60) days from your receipt, that READS software, RROS, hardware assembled boards and hardware unassembled components shall be free of substantial errors or defects in material and workmanship which will materially interfere with the proper operation of the items purchased. If you believe such an error or defect exists, please call Rigel Corporation at (352) 373-4629 to see whether such error or defect may be corrected, prior to returning items to Rigel Corporation. Rigel Corporation will repair or replace, at its sole discretion, any defective items, at no cost to you, and the foregoing shall constitute your sole and exclusive remedy in the event of any defects in material or workmanship. Although Rigel Corporation warranty covers 60 days, Rigel shall not be responsible for malfunctions due to customer errors, this includes but is not limited to, errors in connecting the board to power or external circuitry. This warranty does not apply to products which have been subject to misuse (including static discharge), neglect, accident or modification, or which have been soldered or altered during assembly and are not capable of being tested.

DO NOT USE PRODUCTS SOLD BY RIGEL CORPORATION AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS!
Products sold by Rigel Corporation are not authorized for use as critical components in life support devices or systems. 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.

THE LIMITED WARRANTIES SET FORTH HEREIN ARE IN LIEU OF ALL OTHER WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

YOU ASSUME ALL RISKS AND LIABILITY FROM OPERATION OF ITEMS PURCHASED AND RIGEL CORPORATION SHALL IN NO EVENT BE LIABLE FOR DAMAGES CAUSED BY USE OR PERFORMANCE, FOR LOSS PROFITS, PERSONAL INJURY OR FOR ANY OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES. RIGEL CORPORATION'S LIABILITY SHALL NOT EXCEED THE COST OF REPAIR OR REPLACEMENT OF DEFECTIVE ITEMS.

IF THE FOREGOING LIMITATIONS ON LIABILITY ARE UNACCEPTABLE TO YOU, YOU SHOULD RETURN ALL ITEMS PURCHASED TO RIGEL CORPORATION PRIOR TO USE.

Return Policy. This policy applies only when product purchased directly from Rigel Corporation. If you are not satisfied with the items purchased, **prior to usage**, you may return them to Rigel Corporation within thirty (30) days of your receipt of same and receive a full refund from Rigel Corporation. This does not apply to books. Books are non-returnable.
Please call (352) 373-4629 to receive an RMA (Returned Merchandise Authorization) number prior to returning product. You will be responsible for shipping costs.
All returns must be made within 30 days of date of invoice and be accompanied by the original invoice number and a brief explanation of the reason for the return.
Return merchandise in original packaging.
All returned products are subject to a $15 restocking charge. "Custom Items" are not returnable.

Repair Policy. If you encounter problems with your board or software after the 60 day warranty period, please call Rigel Corporation at (352) 373-4629 or email tech@rigelcorp.com for advice and instruction.

Rigel Corporation will test and attempt to repair any board. You will be responsible for shipping costs and repair fees. If you send a detailed report of the problems you encountered while operating the board, Rigel Corporation will inspect and test your board to determine what the problem is free of charge. Rigel Corporation will then contact you with an estimated repair bill. You will have the choice of having the board fixed, returned to you as is, or purchasing a new board at a reduced price. Rigel Corporation charges repair fees based on an hourly rate of $50.00. Any parts that need to be replaced will be charged as separate items. Although Rigel Corporation will test and repair any board, it shall not be responsible for malfunctions due to customer errors, this includes but is not limited to, errors in connecting the board to power or external circuitry.

Board Kit. If you are purchasing a board kit, you are assumed to have the skill and knowledge necessary to properly assemble same. Please inspect all components and review accompanying instructions. If instructions are unclear, please return the kit unassembled for a full refund or, if you prefer, Rigel Corporation will send you an assembled and tested board and bill you the price difference. You shall be responsible for shipping costs. The foregoing shall apply only where the kit is unassembled. In the event the kit is partially assembled, a refund will not be available, however, Rigel Corporation can, upon request, complete assembly for a fee based on an hourly rate of $50.00. Although Rigel Corporation will replace any defective parts, it shall not be responsible for malfunctions due to errors in assembly. If you encounter problems with assembly, please call Rigel Corporation at (352) 373-4629 for advice and instruction. In the event a problem cannot be resolved by telephone, Rigel Corporation will perform repair work, upon request, at the foregoing rate of $50.00 per hour.

Governing Law. This agreement and all rights of the respective parties shall be governed by the laws of the State of Florida.

Rigel Corporation's Software License Agreement

This Software License Agreement ("Agreement") covers all software products copyrighted to Rigel Corporation, including but not limited to: Reads51, rLib51, RbHost, RitaBrowser, FLASH, rChpSim, Reads166, and rFLI.

This Agreement is between an individual user or a single entity and Rigel Corporation. It applies to all Rigel Corporation software products. These Products ("Products") includes computer software and associated electronic media or documentation "online" or otherwise.

Our software, help files, examples, and related text files may be used without fee by students, faculty and staff of academic institutions and by individuals for non-commercial use. For distribution rights and all other users, including corporate use, please contact:
Rigel Corporation, PO Box 90040, Gainesville, FL 32607
or e-mail tech@rigelcorp.com

Terms and Conditions of the Agreement

1. These Products are protected by copyright laws, intellectual property laws, and international treaties. Rigel Corporation owns the title, copyright, and all other intellectual property rights in these Products. We grant you a personal, non-transferable, and non-exclusive license to use the Products. These Products are not transferred to you, given away to you or sold to you.

   Non-commercial use: These Products are licensed to you free of charge.

   Commercial use: You must contact Rigel Corporation to find out if a licensing fee applies before using these Products.

2. You may install and use an unlimited number of copies of these Products.

3. You may store copies of these Products on a storage device or a network for your own use.

4. You may not reproduce and distribute these Products to other parties by electronic means or over computer or communication networks. You may not transfer these Products to a third party. You may not rent, lease, or lend these Products.

5. You may not modify, disassemble, reverse engineer, or translate these Products.

6. These Products are provided by Rigel Corporation "as is" with all faults.

7. In no event shall Rigel Corporation be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use of or inability to use the Product, even if Rigel Corporation has been advised of the possibility of such damages. Because some states do not allow the exclusion or limitations of consequential or incidental damages, the above limitations may not apply to you.

8. Rigel Corporation makes no claims as to the applicability or suitability of these Products to your particular use, application, or implementation.

9. Rigel Corporation reserves all rights not expressly granted to you in this Agreement.

10. If you do not abide by or violate the terms and conditions of this Agreement, without prejudice to any other rights, Rigel Corporation may cancel this Agreement. If Rigel Corporation cancels this Agreement; you must remove and destroy all copies of these Products.

11. If you acquired this Product in the United States of America, this Agreement is governed by the laws of the Great State of Florida. If this Product was acquired outside the United States of America all pertinent international treaties apply.

TABLE OF CONTENTS

1. OVERVIEW

The RIO-MP is a general purpose I/O board designed for use with Rigel Corporations' family of 8 and 16-bit embedded controllers. The development and implementation of application-specific microcontroller based prototype circuits is significantly simplified with the RIO-MP breadboard area, the three terminal strips, and the user input/output devices (UIOD). These provide flexibility for connecting prototyping components to the microcontroller lines, and for developing and debugging user-designed analog and digital application circuits.

There are 4 IDC headers at the top of the RIO-MP board correspond to the I/O headers found on the various controllers. The IDC headers are used to connect the controllers to the RIO board. The IDC header marked JP3 is to be used with the 16-bit boards, JP4 is to be used with the R-31/R-31JP boards, and JP1 and JP2 are for use with the R-535J board. In addtion there is a header marked JP7 Auxiliary, which may be used when single lines from the embedded controllers are needed. Only one embedded controller should be plugged into the RIO board at a time.

There are a variety of I/O devices available on the RIO board including LEDs, push buttons, seven segment displays, potentiometers, toggle switches, a speaker with amplifier, LCD display, keypad, and stepper motor ICs. There is also an IC (ULN803) which may be used as an open collector for high current / high voltage inputs and outputs to the board. These components are referred to as the User Input Output Devices (UIOD).

1.1 Hardware Overview

Designed to work with current and future 8 and 16- bit microcontrollers boards
On board 5-volt power regulator
Machine screw sockets under all IC's
Power supplied to the board by way of a 2 position terminal block
Power on LED
Operating temperature 0 to 70C
Large solderless breadboard
Two solderless terminal strips allow access to all microcontroller ports and UIOD devices
Prototyping components

- 8 - LEDs
- 10 - Pushbuttons
- 8 - Dip Switches
- 8 - Potentiometers
- 2 - Seven Segment Displays
- 1 - Speaker with Amplifier
- 2 - Stepper Motor Drivers
- 8 - Open Collectors
- 1 - Keypad
- 1 - LCD Display

1.2 Software Overview

Demo programs which may be run on the board are given for 8 and 16-bit processor in C and assembly languages.

1.3 Parts List

RIO-MP Board
User's Manual on WEB
Demo software on included in Reads51, downloadable from the WEB

2. OPERATING NOTES

2.1 Power (JP6)

A 9 to 18 volt alternating current (AC) or direct current (DC) supply should be connected to the terminal block marked JP6. The RIO board has a switching regulator power supply to generate the 5volt VCC used by the digital circuitry. The IDC headers connecting the RIO board to the embedded controllers have VCC and GND lines so that the embedded controllers may receive their VCC current from the RIO board. **It is recommended that the embedded controllers be powered from the RIO board via the IDC headers and ribbon cables in this manner.**

It is possible to connect the embedded controllers to a well-regulated 5-volt supply and power the RIO board from the embedded controllers. Any power circuitry you build on the RIO board will then be limited to the power received from the embedded controllers.

Care should be taken not to power the RIO board and the embedded controller from two separate power supplies. Although not recommended, it is possible to connect two power supplies, one to the embedded controller, and one to the RIO board. In this case, the two supplies must either be floating or have the same ground voltage.

2.2 IDC Headers (JP1, JP2, JP3, JP4, JP7)

The IDC headers found across the top of the RIO board are used to connect the various I/O headers on the embedded controllers to the RIO board. When connecting the embedded controller I/O headers to the RIO board it is important to connect pin 1 on the controller to pin 1 on the RIO board. Pin 1 on both boards is marked with a square box. All of the I/O lines on the IDC headers are made available on JP5 for interfacing to the RIO UIODs and to external circuitry built on the breadboard.

JP1 and JP2 are used for the R-535J board. JP1 on the RIO board corresponds to the I/O header JP2 on the R-535J. JP2 on the RIO board corresponds to the A/D header JP3 on the R-535J.

JP3 is used for the 16-bit family of embedded controllers. The I/O headers on all of these boards have the same pin-out and use the single JP3 header on the RIO board. If additional lines are needed for the 16-bit boards the auxiliary header JP7 on the RIO board may be used.

JP4 is used with the R-31J and R-31JP boards.

JP7 is the auxiliary header and may be used when single additional lines of the embedded controller are needed. An example of this would be implementing a reset button on the RIO board and using it instead of the reset found on the embedded controllers. The reset line from the controller would be wire wrapped to a pin of JP7 and then a jumper wire connected from the corresponding post of JP5 to one of the push buttons marked 0-9 on JP8. Every time the corresponding button is pressed on the RIO board the embedded controller will reset.

2.3 JP5

All microcontroller I/O ports on the embedded controllers are available on the 54-post solderless terminal strip, JP5, located above the breadboard. The posts used for the I/O ports of the 8-bit and 16-bit boards are the same though they are labeled separately. The posts as they appear from left to right are listed below.

POST #	LABEL AND USE	
1	**VAREF** is connected to the microcontrollers VAREF pin. It provides the higher reference voltage to the analog-to-digital converter.	
2	**VAGND** is connected to the microcontrollers VAGND pin. It provides the lower reference voltage to the analog-to-digital converter.	
3-12	**Analog In** labeled 0-9 are connected to the analog to digital port bits of the microcontrollers. (9-0 left to right on the board)	
	8-BIT	**16-BIT**
13-20	**P5.7-P5.0** are connected to the bits of Port 5.	**P3.15-P3.8** are connected to the bits of Port 3
21-28	**P4.7-P4.0** are connected to the bits of Port 4.	**P3.9-3.0** are connected to the bits of Port 3
29-32	**Not Used**	**P2.15-P2.12** are connected to the bits of Port 2
33-36	**P3.5-P3.2** are connected to the corresponding bits of Port 3.	**P2.11-P2.8** are connected to the bits of Port 2
37-44	**P1.7-P1.0** are connected to the bits of Port 1.	**P2.7-P2.0** are connected to the bits of Port 2
45-51	**Auxiliary** labeled 6-0 is connected to JP7	
52	**VSS** is the 9-12 volts unregulated DC supply	
53	**VCC i**s the +5 Volt supply.	
54	**GND** is the ground (0 Volts).	

2.4 JP8, JP9

JP8 and JP9 are used to access the UIODs on the RIO board. UIODs include LEDs, push buttons, seven segment displays, potentiometers, toggle switches, a speaker with amplifier, LCD display, keypad, and stepper motor ICs. There is also an IC (ULN803) which may be used as an open collector switch for high current / high voltage inputs and outputs to the board. The UIODs may be accessed on the two 54 post terminal strips marked JP8 and JP9.

2.4.1 JP8

The terminal strip JP8 has the following UIODs connected to it, 8 light emitting diodes (LED) marked 7 to 0, 10 push buttons marked 9-0, a keypad marked 7-0, LCD display marked E, R/W, RS, Data 7-0, and 2 seven segment displays marked HIGH DIGIT and LOW DIGIT

2.4.2 JP9

The terminal strip JP9 is connected to 8 potentiometers marked 7-0, 8 toggle switches marked 7-0, a speaker with amplifier marked IN, OUT, and V+, 2 stepper motor drivers marked Stepper A and Stepper B, and open collector switches for high current / high voltage inputs and outputs, marked OC drivers In 0-7 and Out 0-7.

TABLE FOR JP5, JP8, AND JP9 POST ASSIGNMENTS

POST #	JP5		JP8	JP9
1	VAREF		LEDs 7	POTS 7
2	VAGND		LEDs 6	POTS 6
3	ANALOG 9		LEDs 5	POTS 5
4	ANALOG 8		LEDs 4	POTS 4
5	ANALOG 7		LEDs 3	POTS 3
6	ANALOG 6		LEDs 2	POTS 2
7	ANALOG 5		LEDs 1	POTS 1
8	ANALOG 4		LEDs 0	POTS 0
9	ANALOG 3		PUSH BUTTON 9	DIP SW 7
10	ANALOG 2		PUSH BUTTON 8	DIP SW 6
11	ANALOG 1		PUSH BUTTON 7	DIP SW 5
12	ANALOG 0		PUSH BUTTON 6	DIP SW 4
13	**8-BIT** P5.7	**16-BIT** P3.15	PUSH BUTTON 5	DIP SW 3
14	P5.6	P3.14	PUSH BUTTON 4	DIP SW 2
15	P5.5	P3.13	PUSH BUTTON 3	DIP SW 1
16	P5.4	P3.12	PUSH BUTTON 2	DIP SW 0
17	P5.3	P3.11	PUSH BUTTON 1	AMP IN
18	P5.2	P3.10	PUSH BUTTON 0	AMP OUT
19	P5.1	P3.9	KEYPAD 7	AMP V+
20	P5.0	P3.8	KEYPAD 6	STEPPER A IN 3
21	P4.7	P3.7	KEYPAD 5	STEPPER A IN 2
22	P4.6	P3.6	KEYPAD 4	STEPPER A IN 1
23	P4.5	P3.5	KEYPAD 3	STEPPER A IN 0
24	P4.4	P3.4	KEYPAD 2	STEPPER A COM
25	P4.3	P3.3	KEYPAD 1	STEPPER A OUT 3
26	P4.2	P3.2	KEYPAD 0	STEPPER A OUT 2
27	P4.1	P3.1	LCD E	STEPPER A OUT 1
28	P4.0	P3.0	LCD R/W	STEPPER A OUT 0
29	Not Used	P3.15	LCD RS	STEPPER B IN 3
30	Not Used	P3.14	LCD DATA 7	STEPPER B IN 2
31	Not Used	P3.13	LCD DATA 6	STEPPER B IN 1
32	Not Used	P3.12	LCD DATA 5	STEPPER B IN 0
33	P3.5	P3.11	LCD DATA 4	STEPPER B COM
34	P3.4	P3.10	LCD DATA 3	STEPPER B OUT 3
35	P3.3	P3.9	LCD DATA 2	STEPPER B OUT 2
36	P3.2	P3.8	LCD DATA 1	STEPPER B OUT 1
37	P1.7	P3.7	LCD DATA 0	STEPPER B OUT 0
38	P1.6	P3.6	7-SEG HIGH G	OC DRIVERS IN 7
39	P1.5	P3.5	7-SEG HIGH F	OC DRIVERS IN 6
40	P1.4	P3.4	7-SEG HIGH E	OC DRIVERS IN 5
41	P1.3	P3.3	7-SEG HIGH D	OC DRIVERS IN 4
42	P1.2	P3.2	7-SEG HIGH C	OC DRIVERS IN 3
43	P1.1	P3.1	7-SEG HIGH B	OC DRIVERS IN 2
44	P1.0	P3.0	7-SEG HIGH A	OC DRIVERS IN 1
45	AUXILIARY 6		7-SEG LOW G	OC DRIVERS IN 0
46	AUXILIARY 5		7-SEG LOW F	OC DRIVERS COM
47	AUXILIARY 4		7-SEG LOW E	OC DRIVERS OUT 7
48	AUXILIARY 3		7-SEG LOW D	OC DRIVERS OUT 6
49	AUXILIARY 2		7-SEG LOW C	OC DRIVERS OUT 5
50	AUXILIARY 1		7-SEG LOW B	OC DRIVERS OUT 4
51	AUXILIARY 0		7-SEG LOW A	OC DRIVERS OUT 3
52	VSS		VSS	OC DRIVERS OUT 2
53	+5V		+5V	OC DRIVERS OUT 1
54	GND		GND	OC DRIVERS OUT 0

2.5 LCD Potentiometer (PT1)

PT1 is used to control the contrast on the LCD display. Rotating the potentiometer will change the contrast making the LCD display either lighter or darker.

2.6 GAIN/IN Potentiometers (R15, R16)

The potentiometers labeled GAIN and IN are used with the speaker and amplifier. The GAIN potentiometer sets the gain of the amplifier. and IN attenuates the input.

2.7 JP12 and JP13

JP12 is used to access P3.12 the BHE# line on the 16-bit controllers. JP13 is used to access P3.13 the WR# line on the 16-bit controllers. JP12 and JP13 are not used with the 8-bit boards.

3. UIOD - User Input/Output Devices

3.1 LEDs

The output LEDs are in a common anode configuration, meaning that their anodes are connected to +5 Volts (VCC) by current-limiting resistors. There are several ways the LEDs may be turned on and off. The LED may be turned on and off by connecting the corresponding post on the solderless terminal strip to the corresponding post for a push button and then pressing the button. The LED may be turned on by connecting the corresponding post on the solderless terminal strip to the ground (0 Volts). The ground is available on the JP8 at the position marked GND. The LED may also be switched on by applying a logic LOW signal to the LED posts 0-7.

3.2 Push Buttons, Dip Switches

There are 18 switches, as part of the UIODs, to be used as digital input devices. Each switch has one terminal connected to ground, and the other available on one of the 54-post terminal strips. The 10 push button switches terminate at the set of posts marked Push Buttons 0-9 on JP8. Similarly, the 8 toggle switches which are implemented by an 8 contact DIP switch marked DIP SW, are terminated on JP9 at the posts marked DIP-SW 7-0. **Care should be taken not to connect these output posts to the power post VCC, or VSS, since closing (making contact) the corresponding switch will short power to ground**. Output devices sink current, and are therefore appropriate to drive the microcontrollers input ports. As a simple demonstration, connect LED 0 to Pushbutton 9. Observe that LED 0 lights when push button 9 is pressed.

3.3 Seven Segment Displays

The 14 posts for the seven segment displays on JP8 are grouped into 2 sets marked HIGH DIGIT and LOW DIGIT. Each display has segments with the standard designations A to G. The output devices are in a common anode configuration, meaning that their anodes are connected to +5 Volts (VCC) by current-limiting resistors. An output device, say segment D of DISPLAY LOW, may be turned on by connecting the corresponding post on the solderless terminal strip to ground (0 Volts).

3.4 Potentiometers

In addition to the digital input devices, the UIODs contain 8 potentiometers, marked POTs 0 to 7. Each potentiometer has one fixed terminal connected to +5 Volts (VCC), the other fixed terminal to 0 Volts (the ground GND), and its wiper terminal available at the 54-post terminal strip marked POTs.

3.5 Speaker and Amplifier

There is a speaker with amplifier built into the RIO board. It may be used to produce sound either by pulsing controller pins or pulsing low level signals from external sources.

3.6 Stepper Motor Drivers

There are 2 stepper motor ICs on the RIO board. Each has 4 transistors built in for driving the stepper motors. See section 5 for details on how to use these ICs.

3.7 Open Collectors

There are 8 open collector outputs available to drive loads upto 50volts or 500Ma using TTL outputs of the microcontroller. See section 6 for details.

3.8 Keypad

The RIO board is designed to be used with standard matrix type keypads.
See section 7 on working with keypads.

3.9 LCD Display

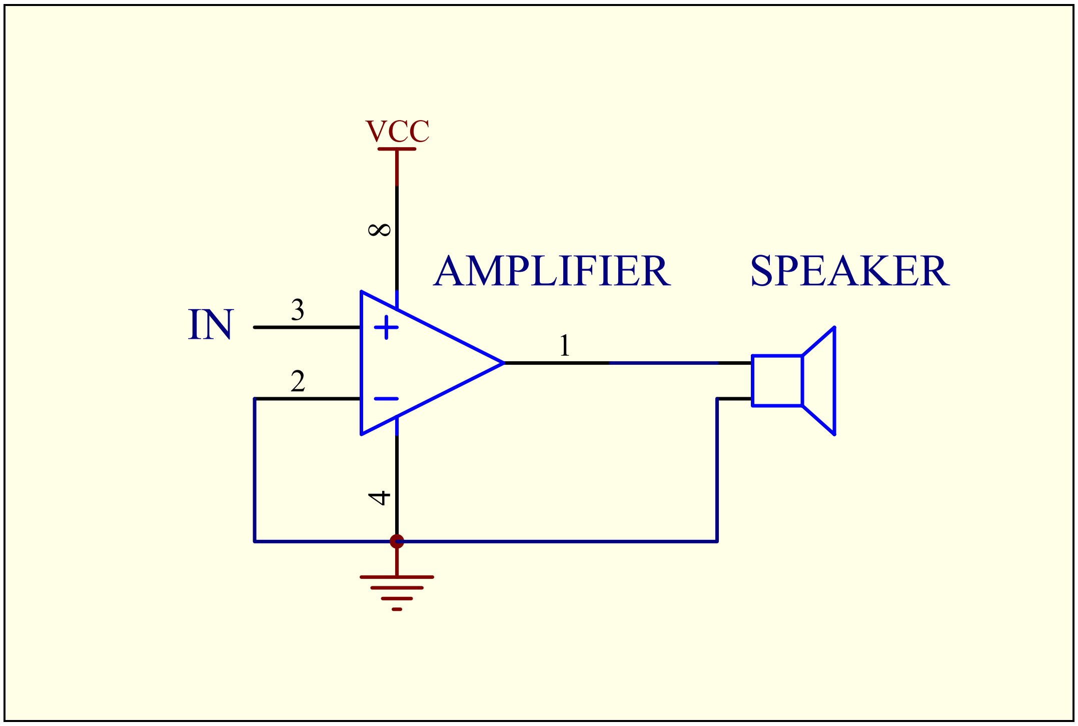
A variety of LCD displays will work on the RIO board. The LCDs' have 2 standard pin configurations, a 2x7 or a 1x14. The pin-outs are the same in either configuration and the RIO board is designed to allow the use of both configurations.
See section 8 on working with LCD displays.

4. WORKING WITH THE SPEAKER AND AMPLIFIER

Background

Microcontrollers can be used to generate different tones by pulsing an output line at different frequencies. This may be observed on the RIO board using the speaker and amplifier provided. Connect an output line to the IN post under AMP on JP9. The OUT post can be used to power the speaker on the RIO board. V+ should be connected to the ground on JP8. LOW level signals can then be pulsed using the microcontroller to generate tones.

Another use of the speaker and amplifier would involve connecting V+ to Vcc on the RIO board. The AMP IN terminal can then be connected to other ICs like sound recording and playback ICs.



5. WORKING WITH STEPPER MOTORS

5.1 Background

Stepper motors are convenient for applications where a high degree of positional control is required. Many of the features required for precise motion control, such as direction, half or full stepping sequences, speed, acceleration, and position tracking may be accomplished by software, decreasing the amount of hardware needed and reducing the cost.
Printers, tape drives, disk drives, robot joints, for example, are typical application areas for stepper motors. In the simplest form, a stepper motor has a permanent magnet rotor and a stator consisting of two coils. The rotor aligns with the stator coil that is energized, as depicted below. By changing which coil that is energized, as illustrated in the following figures, the rotor is turned.

5.2 Procedure

In the figure below, the permanent magnet rotor lines up with the coil pair that is energized. The direction of the current determines the polarity of the magnetic field, and thus, the angular position of the rotor.

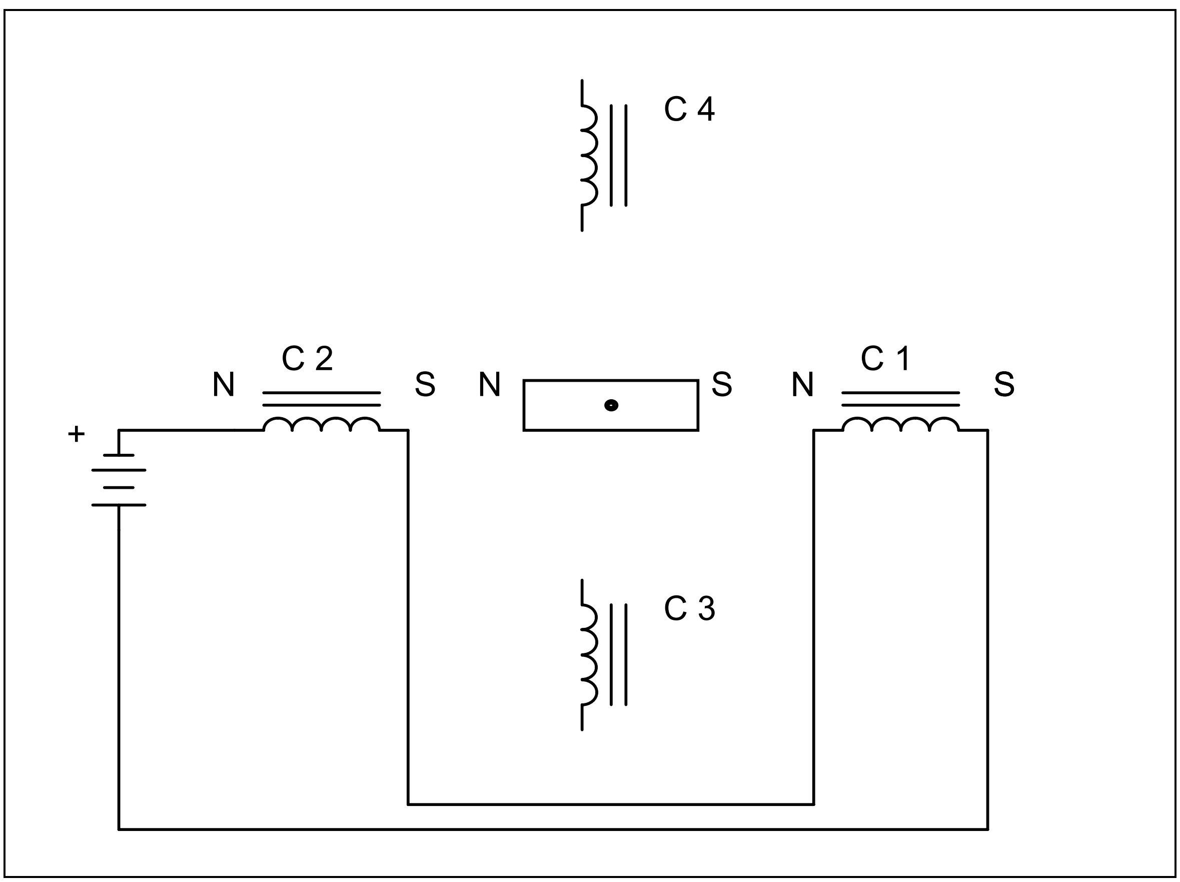


Figure 5.1 Full Step 1

Energizing coil pair C3-C4 causes the rotor to rotate 90 degrees. Again, the direction of the current determines the magnetic polarity, and thus, the angular position of the rotor. In this example, the direction of the current causes the rotor to move in the clockwise direction, as shown in Figure 5.2.

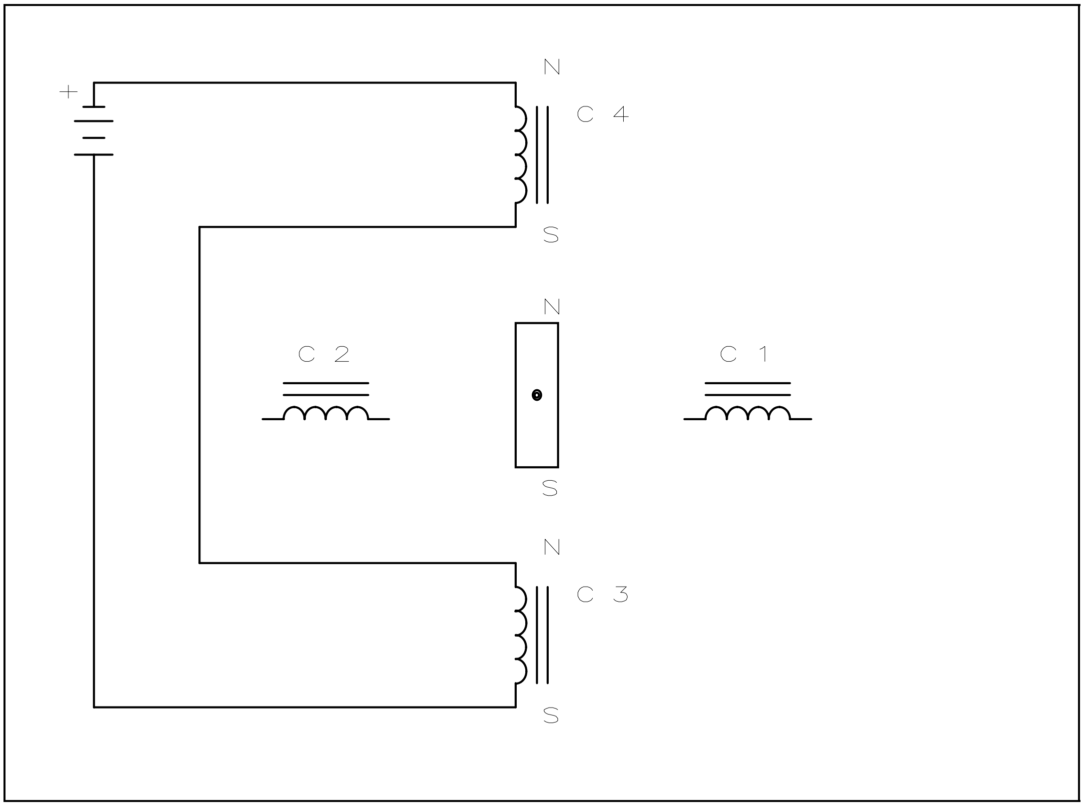


Figure 5.2 Full Step 2

The coil pair C1-C2 is energized again, but with a current opposite to that of in Step 1. The rotor moves 90 degrees in the clockwise direction as shown in Figure 5.3.

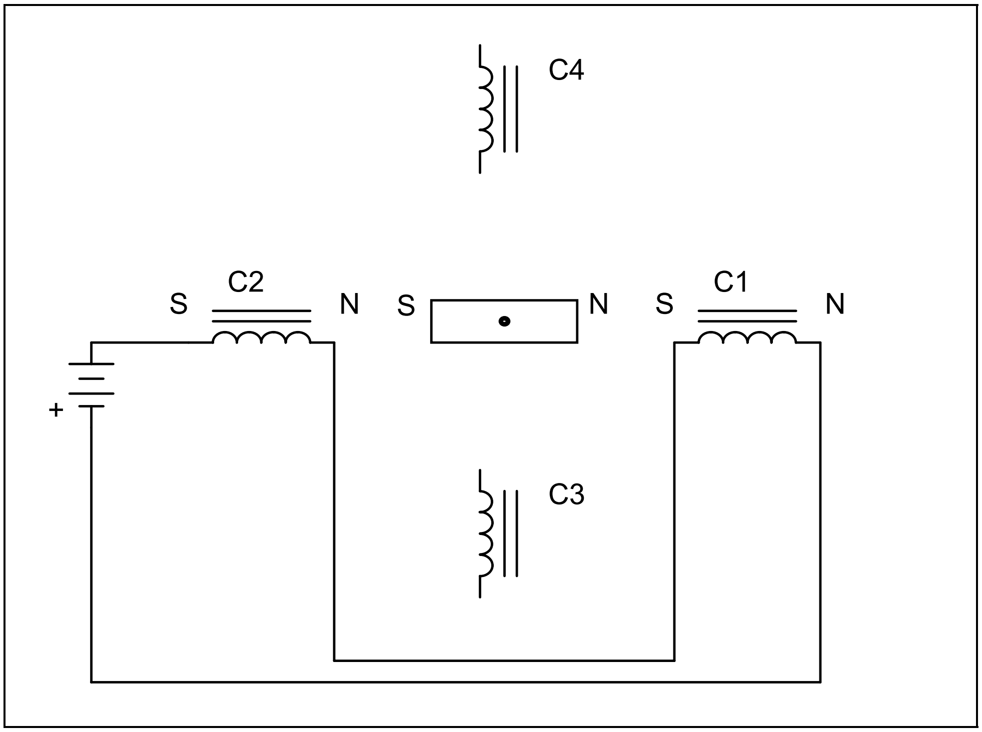


Figure 5.3 Full Step 3

The final full step moves the rotor another 90 degrees in the clockwise direction. Note that again the coil pair C3-C4 is energized, but with a current opposite to that of in Step 2.

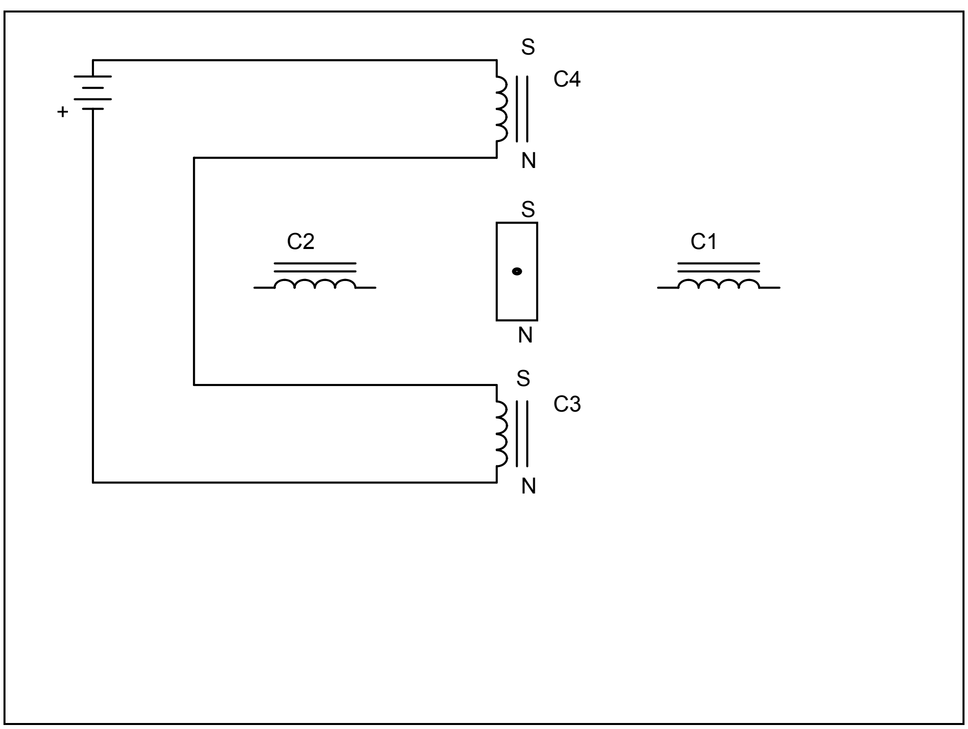


Figure 5.4 Full Step 4

The magnetic polarity of a coil may be reversed by winding two coils on the same bobbin, in opposite directions, and energizing only one winding at a time. In the example shown below, closing switch 1 or switch 2 creates a magnetic field, but with opposite polarities. The mechanical switches may be replaced with switching transistors and controlled by digital signals from a microcontroller.

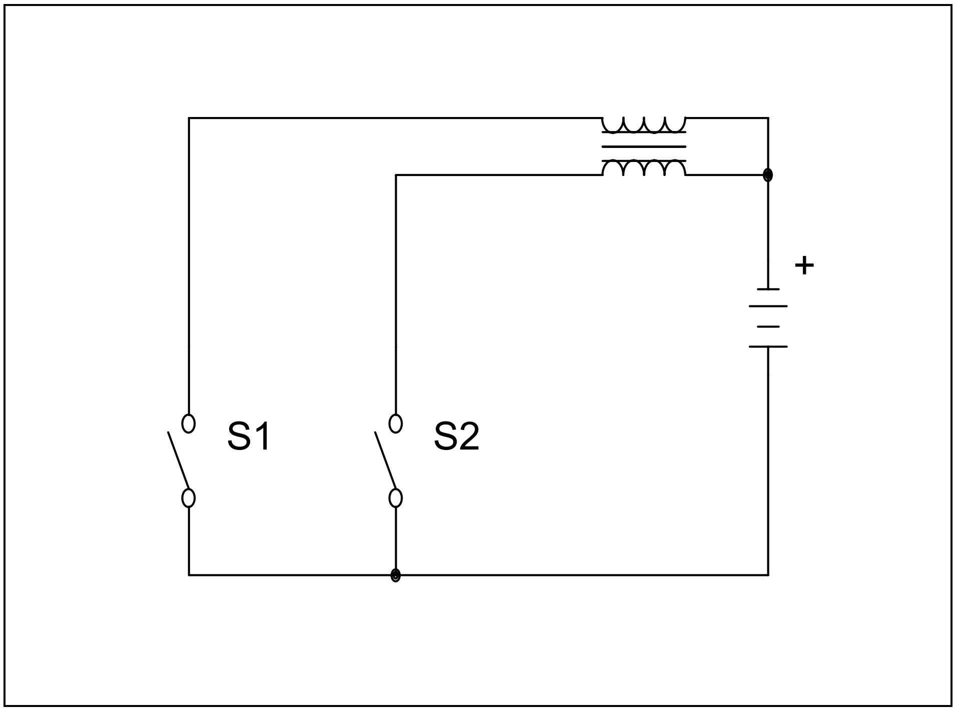


Figure 5.5 Reversing the Magnetic Polarity of Coils

A microcontroller can easily switch currents to the coils and hence control the position of the rotor. Moreover, a microcontroller can time the duration the coil is energized, and hence control the speed of the stepper motor in a precise manner.

Read the manufacturer's specifications on the stepper motor you plan to use with the RIO board. We recommend a stepper motor that uses a 5 volt supply so that VCC may be used to power the motor. The example below uses the unipolar mode of operation, since this mode uses only 4 switching transistors which are available on the ULN2066 populated on the RIO board.

The stepper motor drivers can handle upto 50V - 1.5amps only.

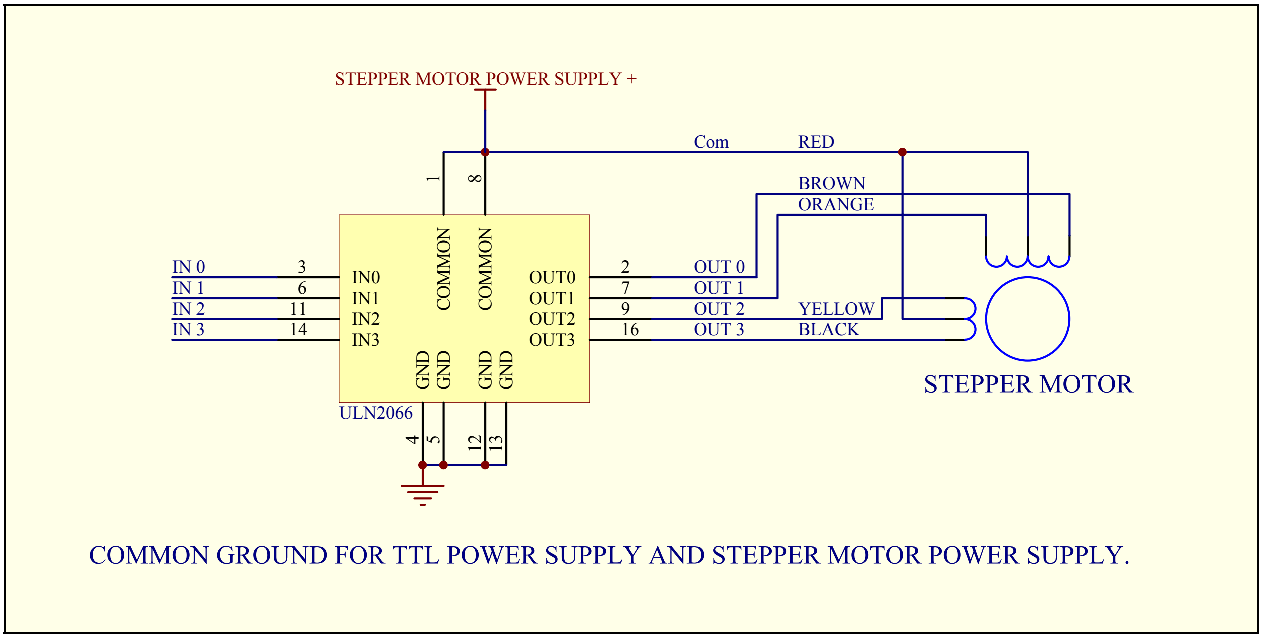


Table 5.1 Full step sequence for clockwise rotation.

Step	Q1 P1.0, Black	Q2 P1.1, Yellow	Q3 P1.2, Orange	Q4 P1.3, Brown	Value
1	ON	OFF	ON	OFF	1010 = 10
2	ON	OFF	OFF	ON	1001 = 9
3	OFF	ON	OFF	ON	0101 = 5
4	OFF	ON	ON	OFF	0110 = 6
1	ON	OFF	ON	OFF	1010 = 10

Table 5.2 Half step Sequence for Clockwise Rotation.

Step	Q1 P1.0, Black	Q2 P1.1, Yellow	Q3 P1.2, Orange	Q4 P1.3, Brown	Value
1	ON	OFF	ON	OFF	1010 = 10
2	ON	OFF	OFF	OFF	1000 = 8
3	ON	OFF	OFF	ON	1001 = 9
4	OFF	OFF	OFF	ON	0001 = 1
5	OFF	ON	OFF	ON	0101 = 5
6	OFF	ON	OFF	OFF	0100 = 4
7	OFF	ON	ON	OFF	0110 = 6
8	OFF	OFF	ON	OFF	0010 = 2
1	ON	OFF	ON	OFF	1010 = 10

In the tables above, ON refers to a conducting transistor, i.e., the base of the transistor must be high. Similarly, OFF indicates that the base of the transistor is grounded. Thus, we put a logic 1 signal to the port driving the transistor to switch the corresponding coil ON, and put a logic 0 to switch the coil OFF. The digital signals which are used to switch the coils on and off are contiguous bits of port 1. The 4-bit value of the lower nibble of Port 1 which would generate the desired pattern is given for each step of the sequence in the last column.

It is a good idea to connect the collectors of the transistors to LEDs L0 to L3. With this arrangement, when a motor coil is energized, the corresponding LED will light up.

6. USING THE OPEN COLLECTORS

6.1 Background

A microcontroller is ideally suited for switching devices like lamps, relays, hammers, or other similar loads. However the TTL levels of the microcontrollers are not capable of driving any load higher than a few milli-amps. The ULN2803 on the board is an octal high voltage, high current Darlington transistor array and can interface logic level TTL circuitry with devices not exceeding 50 volts or 500mA.

6.2 Procedure

ULN2803 is an inverting switch, i.e., a logic LOW switches it on and a logic HIGH switches if off. Figure 6.1 shows how different devices can be interfaced to the microcontroller using the ULN2803

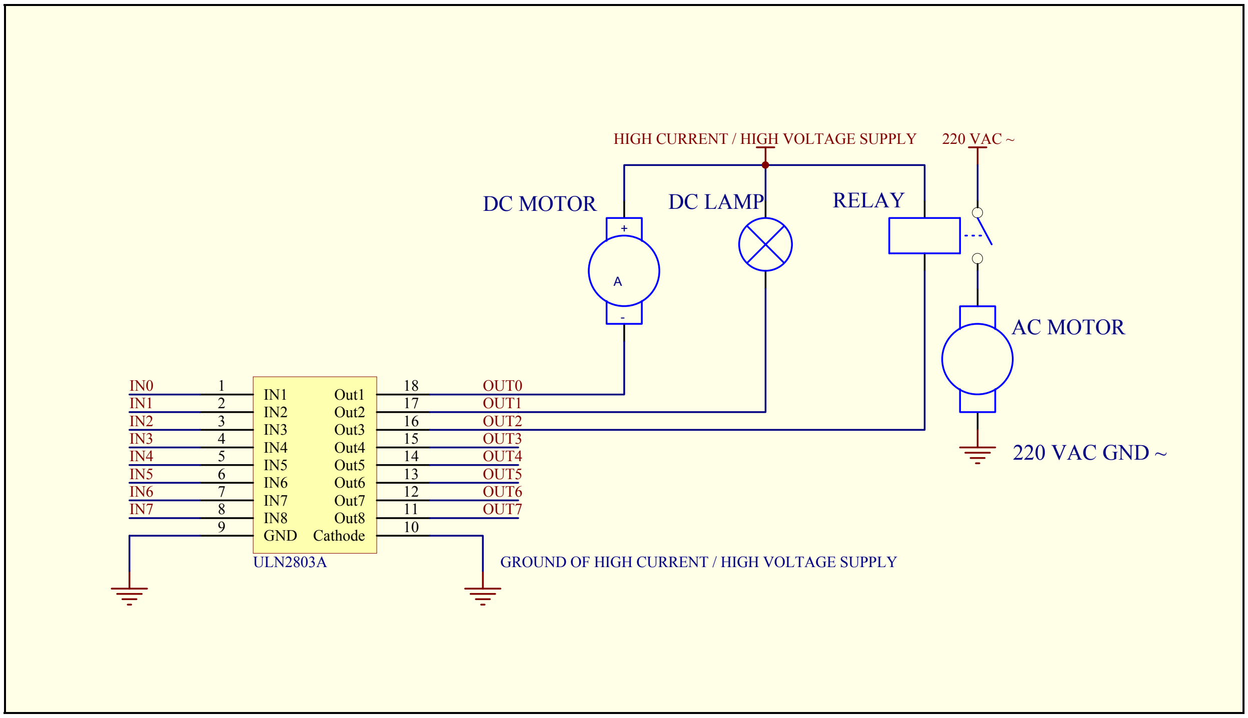


Diagram for the common collector darlington pair chip.

7. WORKING WITH KEYPADS

7.1 Background

Matrix-type keypads consist of a rectangular array of momentary push buttons. Each row and each column of push buttons are connected to a common rail. Suppose a 4 by 4 array of pushbuttons are used. A 4 by 4 array is often used to input hexadecimal numbers. There are 4 column rails and 4 row rails. Each push button has two terminals, one connected to its column rail, and the other, to its row rail. The row and column rails are connected to the microcontroller ports. The columns are driven low by output ports. The rows are then read into the input ports. If no key is pressed, the rows read 1. When a row is detected to be 0, it indicates that a key in that row is pressed. The task now is to detect which key of the row is actually pressed. The microcontroller loops through each column, driving only one column low at a time, as it inspects the row. The microcontroller needs to poll the rows to see if a key is pressed. Only when the column in which the pressed key resides is driven low, the row rail is grounded, and thus the voltage is low. The rows and columns are interchangeable, that is, the rows may be driven low as the columns are read by the input ports.

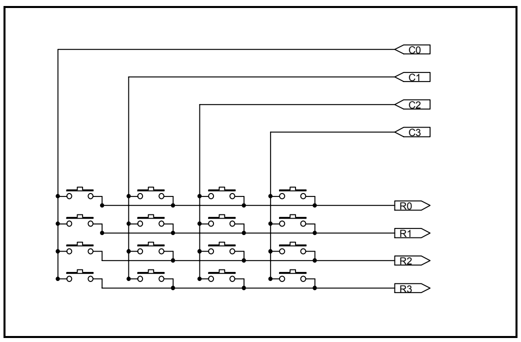


Figure 7.1 A 4-by-4 Matrix Keypad.

The keypad may be implemented to invoke an interrupt, thus reducing the load on the CPU. The interrupt should be invoked whenever a key is pressed. Typically, the rows are combined into an AND gate. With no key pressed, all rows are at level 1, thus the AND gate produces a 1. Whenever a key is pressed, one of the rows becomes 0. This action also causes the output of the AND gate to be 0. A 1-to-0 transition at the output of the AND gate indicates a key has just been pressed. This signal may be used as the external interrupt of the microcontroller. With this arrangement, the CPU detects key press actions in an interrupt service routine, called only upon the event of a key press.

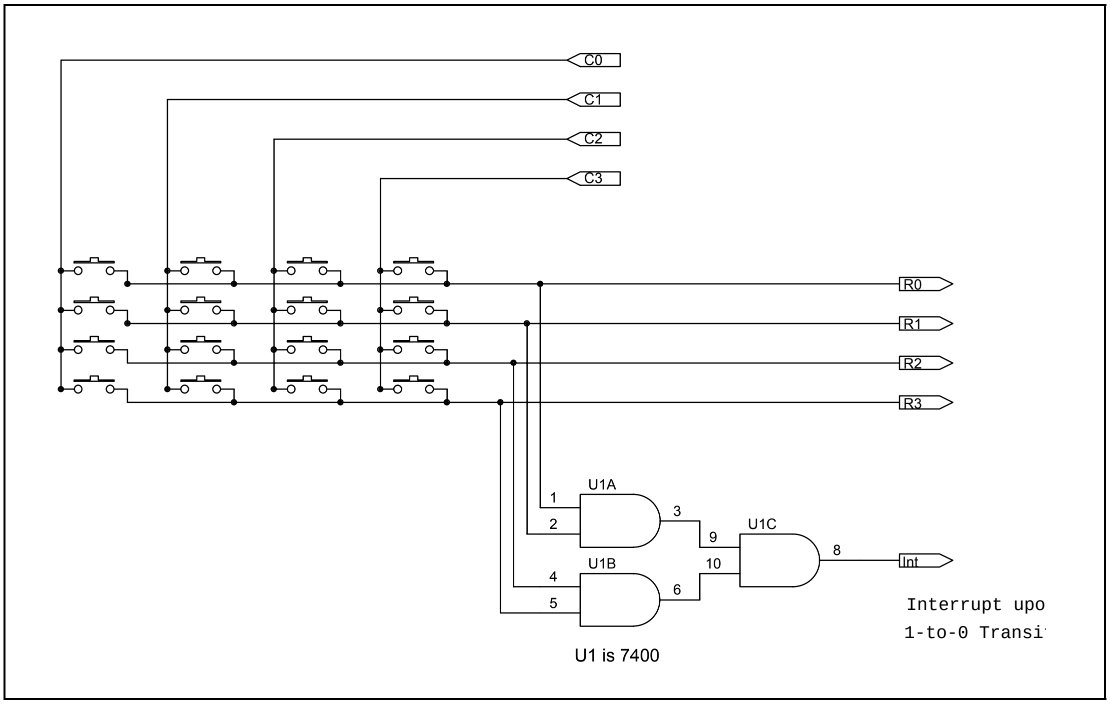


Figure 7.2 Generating an Interrupt when a Key is Pressed.
(Optional Circuitry)

Many push buttons, when pressed, make a series of contacts and breaks before a solid contact is established. These pulses are due to the mechanical motion of the two conductors engaging to make the final contact. These pulses are short in duration and appear as spikes on an oscilloscope. However, since microcontrollers are fast, there is a possibility that a single key press is registered as a few separate key presses, as the key bounces. A "debouncing" scheme is called for. Debouncing may be accomplished in hardware or in software. The simplest software debouncing routine is a short delay loop to wait out the pulses.

8. WORKING WITH LIQUID CRYSTAL DISPLAYS

8.1 Background

A series of Liquid Crystal Display (LCD) modules, commonly available from various manufacturers, provide effective information presentation with very little effort and low program overhead in small computer or embedded controller systems. The LCDs come in various sizes, with 1 to 4 lines, 16 to 40 characters per line, and 5x7 or 5x10 dot display fonts. The LCDs are mounted on printed circuit board along with a character generator and a system controller. Although the LCDs may vary in size, the controller, or more specifically, the protocol in interacting with the controller, is standard. The interface is TTL compatible. The controller may be connected to a 4- or 8-bit data bus. Three control lines are required. The LCD requires a 5 Volt supply for the logic and a separate supply for the liquid crystal driver.

8.2 LCD Module Pins

There are 14 pins to the LCD as listed below. Pins 1-3 are power pins, 4-6 are for the 3 control lines, and the remaining 8 are for the data bus. The levels H and L refer to TTL levels high (logic 1) and low (logic 0).

Pin Number	Symbol	Level	Function
1	VSS	0 Volts	(Ground)
2	VCC	+5 Volts for logic circuit supply	
3	VEE	LCD drive supply	
4	RS	H/L	Register Select Level H selects the Data Register for data input or output. Level L selects the Control Register for instruction input.
5	R/W	H/L	Read Write Select Level H indicates a data read operation (from LCD module to the CPU). Level L Indicates data write operation (from the CPU to the LCD module).
6	E	Pulse L-H-L	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

8.3 LCD Options and Supply Voltages

The 3 separate power pins of the LCD module are Vss, Vcc, and Vee. Vss and Vcc are the standard 0 and 5 Volt TTL-level supply voltages. Vee supplies the liquid crystal display. Vee is held below Vcc. The voltage difference Vcc-Vee is commonly denoted by V0. V0 may be more than 5 Volts, implying that Vee may have a potential less than Vss. That is Vee may be negative with respect to ground. The potential difference V0 needs to be increased (Vee needs to be made more negative) as the ambient temperature decreases. There is little need for a temperature compensation circuit in a laboratory environment. However, if the module is to be operated outdoors, the manufacturers recommend circuitry that increases V0 as the temperature decreases.

Typically there are two distinct versions of the LCD, the so-called normal temperature type and the extended temperature type. Similarly, there are two types of fluid used, referred to as the TN and the NTN fluid definitions. The TN definition modules contain LCD fluid that is twisted approximately 90 degrees which provides a viewing angle of about 40 degrees with a contrast ratio of 4:1. The NTN definition modules contain LCD fluid that is twisted approximately 180 degrees which provides a viewing angle of about 70 degrees with a contrast ration of 8:1.

The extended temperature NTN definition modules provide better performance at the cost of requiring Vee to be at a lower voltage than ground. At room temperature, Vee may typically be -1 to -2 Volts with respect to Vss (ground).

8.4 The Display

The display may be up to 40 characters per line with up to 4 lines. The LCD controller contains Random Access Memory (RAM) to store the codes of 80 characters. This memory is referred to as Display Data RAM or DDRAM. With a display of 20 characters per line and 2 lines, one may think of two virtual lines of 40 characters each. Of these 40 characters, only 20 characters are seen at any given time. The display may be thought of as a window of width 20 over a virtual line of 40 characters. The display may be shifted to view the other characters in the virtual line. Suppose we wrote the 40-character string

"0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcd"

on line 1. The display will show only 20 characters, i.e.,

0123456789ABCDEFGHIJ

provided that the Entry Mode (explained below) specifies not to shift the display. The display may then be shifted to the right 1 notch to reveal the next character (K) as,

123456789ABCDEFGHIJK

while hiding the first character (0). In order to write a character into a given location, the address of the location in DDRAM must first be selected. This process also moves the cursor to the selected location in preparation for the next read or write operation.

8.5 The Character Set

The LCD may display preprogrammed (built-in) or user-defined characters. The LCD controller contains a character generator Read Only Memory (ROM) that contains 192 characters. Characters are selected by their code. There are 96 ASCII characters (selected by their ASCII codes), 64 Japanese characters (kana characters), and 32 special characters such as Greek letters and lower case letters with descenders. The LCD controller also contains Random Access Memory (RAM), referred to as CGRAM (Character Generator RAM) which stores 8 user-defined characters. The user-defined characters must first be loaded into CGRAM and then called to be displayed.

Table8.1. Character Set

	High Nibble												
Low Nibble	0	2	3	4	5	6	7	A	B	C	D	E	F
0	(CGRAM 1)		0	@	P	`	p						
1	(CGRAM 2)	!	1	A	Q	a	q						
2	(CGRAM 3)	"	2	B	R	b	r						
3	(CGRAM 4)	#	3	C	S	c	s						
4	(CGRAM 5)	$	4	D	T	d	t						
5	(CGRAM 6)	%	5	E	U	e	u						
6	(CGRAM 7)	&	6	F	V	f	v			Kana			
7	(CGRAM 8)	'	7	G	W	g	w			and			
8	(CGRAM 1)	(	8	H	X	h	x			Greek			
9	(CGRAM 2)	)	9	I	Y	i	y			Characters			
A	(CGRAM 3)	*	:	J	Z	j	z						
B	(CGRAM 4)	+	;	K	[	k	{						
C	(CGRAM 5)	,	<	L	¥	l	|						
D	(CGRAM 6)	-	=	M	]	m	}						
E	(CGRAM 7)	.	>	N	^	n	→						
F	(CGRAM 8)	/	?	O	_	o	←						

8.6 The Enable Line

The Enable line is the primary control line that is involved in all LCD controller operations. Each operation selects RS and R/W line levels. If the operation is an instruction to the module or a data write operation, the data is placed on the data bus. Then the Enable line is pulsed. The pulse is a positive pulse, L->H->L. For memory read instructions, the data bus is inspected during the Enable pulse, that is, while the Enable line is high. The Enable pulse must be at level H for at least 450 nanoseconds. Further details on timing are discussed below.

8.7 LCD Controller Operations

An LCD controller operation is either a control operation or a data operation. Control operations may be instructions sent to the module or a status byte received from the module. In a typical application, the instructions are sent to the module to place the module in a certain mode of operation. Then, most of the interaction with the module consists of data operations, sending character codes to the module so that the characters are displayed. Occasionally, instructions to shift the cursor or the display or to clear the display may be issued.

The operation is a control operation when the RS (Register Select) line is held at level L (logic 0). All instructions originate from the host system. The instructions are bytes written to the module. Thus, the R/W (Read/Write) line must also be held at level L. The R/W line is held at level H to interrogate the status of the LCD controller. The control operations may be classified into 5 categories: initialization instructions, cursor instructions, display instructions, CGRAM (Character Generator Ram) instructions, and the Status Inquiry.

8.7.1 Initialization Instructions

Power On Reset

The module self-initializes when power is first applied. The steps of this power-on-reset procedure may also be repeated by sending 3 bytes of instruction 30h. Actually, the last nibble of the instruction is ignored. More specifically, DB7=DB6=0 and DB5=DB4=1. This facilitates both 4 and 8 bit data busses.

Function Set

The size of data bus, the number of lines of display, and the font size are specified by the function set. The instruction in binary is,

(0 0 1 DL N F x x)

where,

DL=0 for a 4-bit data bus and DL=1 for an 8-bit data bus;

N=0 for 1 display line and N=1 for 2 display lines;

F=0 for 5x7 dot character fonts and F=1 for 5x10 dot fonts.

The x indicates that the state of this bit does not matter.

Entry Mode

The Entry Mode refers to the actions of the cursor and the display following every character read or write operation. The common mode of operation is to increment the cursor one character to the right, keeping the current contents of the display in their places. With this arrangement, the next character will be written into the location immediately to the right of the current location. The instruction in binary is,

(0 0 0 0 0 1 I/D S)

where,

I/D=0 to decrement the cursor (move one notch to the left),

I/D=1 to increment the cursor (move one notch to the right);

S=1 to shift the entire contents of the display one notch to the left or to the right, depending on the value of I/D, and S=0 otherwise.

8.7.2 Cursor Instructions

The cursor specifies the location into which the next character will be written or from which the next character will be read. The Entry Mode instruction described above determines the operating mode of the cursor.

Home Cursor

The Home position of the cursor is at the left-most character in the top line. The instruction in binary is,

(0 0 0 0 0 0 1 x)

Move (Place) Cursor

The cursor may be placed anywhere in the DDRAM. Depending on the DDRAM location and on how much the display is shifted, the cursor may or may not fall into the range of displayed locations. Moving the cursor is accomplished by selecting the DDRAM address. The instruction in binary is,

(1 A6 A5 A4 A3 A2 A1 A0)

where A0-A6 is the DDRAM address. In a two-line display, the addresses 0 to 27h (0 to 39 decimal) correspond to the first line and addresses 40h to 67h (64 to 103 decimal) correspond to the second line.

Hide/Show/Blink Cursor

The cursor may be hidden from view, although it still points to the location the next data transfer will take place. The cursor is normally an underline, called the line cursor. Alternatively, the character at the cursor location may be blinked. With a blinking character, the (line) cursor may be hidden or shown. The instruction also includes a bit to control the entire display. The instruction in binary is,

(0 0 0 0 1 D C B)

where

D=1 for display on, D=0 for display off;
C=1 for cursor on, C=0 for cursor off;
B=1 to blink the character at the cursor position, 0 otherwise.

While keeping the display on, the instructions for no cursor, blinking character, line cursor, and blinking character along with the line cursor are 0Ch, 0Dh, 0Eh, and 0Fh, respectively.

Shift Cursor

The cursor or the entire display may be shifted to the left or to the right. The instruction in binary is,

(0 0 0 1 S/C R/L x x)

where

S/C=0 to shift the cursor and S/C=1 to shift the entire display;
R/L=0 to shift to the left and R/L=1 to shift to the right.

8.7.3 Display Instructions

Display Off/On

The display may be blanked while retaining the contents of the DDRAM. The instruction also controls the cursor mode. The instruction in binary is,

(0 0 0 0 1 D C B)

where

D=1 for display on, D=0 for display off;
C=1 for cursor on, C=0 for cursor off;
B=1 to blink the character at the cursor position, 0 otherwise.

While keeping the line cursor on, the instructions to blank the display with and without the blinking character at the cursor position are 0Ah, 0Bh, respectively. In order to turn the display on with no cursor, blinking character cursor, line cursor, and line cursor along with the blinking character are 0Ch, 0Dh, 0Eh, and 0Fh, respectively.

Clear Display

Clearing the display places the space character (ASCII 20h) into all DDRAM locations and homes the cursor to the first character of the top line. The display is also shifted back to its origin, i.e., the DDRAM location 40h corresponding to the first character on the top line. The instruction is 01, or in binary,

(0 0 0 0 0 0 0 1).

Shift Display

The cursor or the entire display may be shifted to the left or to the right. The instruction in binary is,

(0 0 0 1 S/C R/L x x)

where

S/C=0 to move cursor and S/C=1 to shift the entire display;
R/L=0 to shift to the left and R/L=1 to shift to the right.

<u>CGRAM (Character Generator RAM) Instructions</u>

Set CGRAM Address
This instruction is similar to the move cursor operation. Although there is no cursor for the CGRAM, the location which will be involved in the next data transfer operation is set in a fashion similar to that of placing the cursor. The CGRAM contains 64 bytes. The instruction in binary is

(0 1 A5 A4 A3 A2 A1 A0)

where A0-A5 is the CGRAM address. The use of CGRAM is discussed below and illustrated by examples.

Status Inquiry
The LCD controller contains a status byte which contains the Busy Flag (BF) and the Address Counter (AC). The BF=1 if the LCD controller is still busy executing the last operation and BF=0 if the LCD controller is ready for the next instruction. The AC is a 7-bit address used for both DDRAM and CGRAM addresses. AC reports the current DDRAM address if the most recent data operation involved the DDRAM or if the DDRAM address was just set. Similarly, it shows the current CGRAM if the most recent data operation involved the CGRAM or if the GCRAM address was just set. The status byte is received by a read operation. The RS line is held at level L to select a control operation and the R/W line is held at level H to select a read operation. The status byte is then received through the data bus. The status byte, in binary, has the following format.

(BF AC6 AC5 AC4 AC3 AC2 AC1 AC0)

where BF=1 if the LCD controller is busy, 0 otherwise;
AC0-AC6 is the 7-bit Address Counter.

8.7.4 Data Operations
The operation is a memory read or write operation when the RS (Register Select) line is held at level H (logic 1). Data may be transferred to or from either DDRAM or CGRAM. The address of data locations in DDRAM or CGRAM must first be selected.

The cursor position indicates the DDRAM location which will be involved in the next data transfer operation. Thus, to select a DDRAM location the cursor must be moved by either of the following operations: move cursor, shift cursor, home cursor, or reset display (which also homes the cursor). Note that the cursor location determines the selected DDRAM address even when the cursor is hidden. Also note that, depending on the Entry Mode (explained above) selected, the cursor is incremented or decremented after each data transfer operation. In order to write a string into successive DDRAM locations, select the increment cursor mode and place the cursor at the beginning of the DDRAM block. The entire string may now be written to DDRAM with successive write operations, as the cursor is automatically incremented by the LCD module. Data is written to DDRAM by holding the R/W (Read/Write) line at level L (logic 0). Data is read from DDRAM by holding the R/W (Read/Write) line at level H (logic 1).

Transferring data to and from the CGRAM requires similar steps. First the CGRAM address is set. Then data may be written to or read from CGRAM. Again, the R/W line determines the direction of the data transfer operation. Also note that the CGRAM location is incremented or decremented, depending on the Entry Mode selected, following each data transfer operation that involves the CGRAM.

4-Bit Operation

The LCD module may be interfaced through either a 4-bit data bus or an 8-bit data bus. When a 4-bit data bus is used, only the higher 4 lines (DB4 to DB7) of the LCD module data bus is used. Each data or instruction is then sent in two parts, the higher order nibble first, followed by the lower order nibble.

The default configuration is the 8-bit data bus mode. The LCD module is initiated in the 8-bit mode upon power up. In order to implement the 4-bit data bus operation, the control word 20h (DB7=DB6=DB4=0 and DB5=1) must be sent. Notice that when the LCD module receives this control word, it is in the 8-bit data bus mode. Thus, the lower nibble should not be sent to the module.

Timing Considerations

The LCD controller is a relatively slow controller. Care should be taken to allow time between successive operations. The time required to complete each operation is given in the following table.

Table 8.2. Instruction Set

Instruction	RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Action
Quench Display	0	0	0	0	0	0	0	0	0	1	Collective data store, but not the character generator, is erased; cursor is set to home position; address 00 is at upper left.
Cursor Home	0	0	0	0	0	0	0	0	1	x	Cursor is set in home position; if the display was shifted, it is relocated; address 00 is at upper left; stored data remain unchanged.
Manner in which characters are displayed	0	0	0	0	0	0	0	1	ID	S	Determines into which direction the cursor will be shifted (ID) after a character has appeared, and whether simultaneously the entire display should be shifted one position
Display and Cursor On/Off	0	0	0	0	0	0	1	D	C	B	Display is switched on or off (D); cursor (line under character) is switched on or off (C); cause the character at the cursor position to blink (B)
Shift Cursor or Display	0	0	0	0	0	1	SC	RL	x	x	Cursor or entire line is shifted without any change to the memory contents
Function	0	0	0	0	1	DL	N	x	x	x	Indicates the width of the data bus and whether the upper row or both rows will be used
Character Generator Address	0	0	0	1	digits			lines			Sets the address of the memory of the character generator; the subsequent data produce the relevant character pattern
Data Memory Address	0	0	1	address							Sets the address of the data memory; the subsequent data produce the relevant ASCII characters
Busy Flag: Read Address	0	1	BF	address							Reads BF to determine whether the display is ready for the next instruction; also reads the cursor position
Write Data	1	0	data								Writes data into the data memory or into the character generator
Read Data	1	1	data								Reads data from the data memory or into the character generator

In order to obey the timing requirements, the host CPU may time the interval between successive operations, or inspect the BF (Busy Flag) and wait until BF=0 before performing the next operation.

Timing is still somewhat delicate, often requiring further experimentation. A case in point is the sample software LCD4V.ASM where the analog-to-digital conversion is implemented as an interrupt driven task, thus disturbing the timing requirements of the LCD module. The solution given is to disable interrupts (clear EAL) while interacting with the LCD module.

Hardware Considerations
The timing requirements call for a relatively slow operation. Hence, the module should be considered as a peripheral to be controlled through ports, rather than in a memory-mapped input/output configuration. That is, the LCD module data bus as well as its control lines should be connected to latched ports rather than the data bus and the address bus of the host system. If a memory-mapped input/output configuration is chosen, care must be taken to slow down the memory read and write cycles of the host system. The example software given below prescribes that the data bus and the control lines of the LCD module be connected to the ports of the 8031 family of microcontrollers.

Software Architecture
There are two aspects to the high-level LCD interface software: LCD management and message display operations. Following structured programming principles, the LCD management software, which invokes the control operations, should be kept independent of the data transfer routines.

Typically, software controls which characters are placed on display, at which locations, which source the characters are obtained from, and how long the characters stay on display. The characters displayed may be one of the following:

1. Preprogrammed messages or strings stored in the memory (more likely in ROM) of the host system. These messages may be, for instance, menu choices presented to the user.

2. Run-time messages or stings computed and stored in the RAM of the host system. On a test instrument which employs the LCD module, for instance, current readings of voltage or temperature are run-time messages.

In either case, the source is a string stored as a block of memory by the host system.

The sample software given below contains subroutines to reset the LCD module, move the cursor, select cursor type, home the cursor, clear the display, shift the cursor or display.

Sample Software, Written for the 8051 Microcontroller is illustrated below.

Sample software available on disk in the book **Programming and Interfacing the 8051 Microcontroller** by Yeralan or on RIGEL Corporation's home page at **http://www.rigelcorp.com** illustrates various subroutines that perform most of the LCD control tasks. First consider the 8-bit data bus implementation illustrated by the program LCD8.ASM, and then the 4-bit data bus implementation, illustrated by LCD4.ASM. Finally, the LCD display is used to present the Voltage measured on the AN0 port of the 80535.

<u>8-Bit Data Bus Operation, LCD8.ASM</u>

Port 1 is connected to the data lines DB0-DB7 (P1.0 to DB0, P1.1 to DB1, etc.). Bits 0-2 of Port 4 are used for the control lines RS, R/W, and E. These connections are defined by EQU pseudo ops:

```
; UIOD connections
;
; constants
 LCD_DATA  equ 090h ; port 1 is used for data
 LCD_RS    equ 0e8h ; p4.0 LCD Register Select line
 LCD_RW    equ 0e9h ; p4.1 LCD Read / Write line
 LCD_E     equ 0eah ; p4.2 LCD Enable line
```

The configuration specified during the function set operation and the entry mode are also defined.

```
; system instructions
 Config    equ 38h  ; 2 lines, 5 by 7 character matrix
 entryMode equ 6    ; increment cursor, do not shift display
```

Next, the commonly used control words are defined: the cursor and display control words.

```
; cursor control instructions
 offCur    equ  0Ch
 lineCur   equ  0Eh
 blinkCur  equ  0Dh
 combnCur  equ  0Fh
 homeCur   equ  02h
 shLfCur   equ  10h
 shRtCur   equ  14h

; display control instructions
 clrDsp    equ  01h
 offDsp    equ  0Ah
 onDsp     equ  0Eh
 shLfDsp   equ  18h
 shRtDsp   equ  1Ch
```

The two subroutines wrLCDdata and wrLCDcom are perhaps the most important routines. These routines send the contents of the R0 register to the LCD module as data or as a command, respectively. Notice that there is a good deal of overlap between the two routines. In fact, the only difference is that the RS line is set or cleared, indicating whether a data or an instruction operation is to follow. A command may be issued by first loading the corresponding control word into R0, and then calling wrLCDcom. The routine rdLCDstat returns the status byte (Busy Flag and the Address Counter) in the accumulator.

The routines crShLf, crShRt, dspShLf, and dspShRt shift the cursor or the display to the left or to the right the number of places given in the accumulator. The routine placeCur places the cursor on line a at location b where a and b are the accumulator and the b register contents. Similar to placeCur, which selects the Display Data RAM (DDRAM) address, setCGRAM selects the Character Generator RAM address. Again, the accumulator is used to specify the character code [0..7], and the b register, the font row [0..7]. Any data transfer immediately following setCGRAM will interact with the CGRAM.

The routine pulseEwait sends a high pulse on the Enable (E) line and then continuously interrogates the Busy Flag (BF) until BF is cleared. The routines initLCD and resetLCD initiate and replicate the power-on-reset operations, respectively. The initialization resets the entry mode, clears the display, and homes the cursor.

The routine prtLCD facilitates printing messages on the LCD module. This routine assumes that the string to be displayed immediately follows the call, and is terminated by a zero. For example, the segment

```
lcall prtLCD
db    "Hello !",0
```

will display the string "Hello !" starting at the cursor position. It is assumed that the entry mode is set to increment the cursor to the right.

The demonstration program LCD8.ASM first resets and initialized the LCD module by calling routines initLCD and resetLCD. It then prints a greeting message. The program waits until a character is received at the serial port. That is, the program is suspended until a key is pressed at the host. Next, the control words offDsp and onDsp are loaded into R0 and issued as commands using routine wrLCDcom. The display is first hidden and then restored.

Next, the cursor is placed on line 2, position 7. Another message is displayed on the LCD. The display is then cleared by issuing the clrDsp command.

User-defined characters are illustrated next. Font row 0 of Character 0 is selected with the setCGRAM routine. 16 rows, corresponding to 2 characters (character 0 and 1) are downloaded by the prtLCD routine. Note that prtLCD simply sends the null-terminated string that immediately follows to the LCD module as data. The LCD module will store this data in the CGRAM, since setCGRAM is the most recent operation specifying a RAM address. Also note that the string that is sent may not contain a 0, which would falsely signal the end of the string. When a font row needs to be blank, the character 20h is used. The last 5 bits of 20h are 0, blanking the font row of width 5.

Clearing the display selects DDRAM. User-defined characters may now be used in strings. Note that character 0 is specified by its alternate address 8, to prevent signaling the end of the string.

In order to illustrate the 40-character virtual line, a long message of 40 characters is displayed on line 1. The cursor is incremented after the 40-th character to the beginning of line 2. The cursor is homed, and the display is shifted to the right, to the left, and then back to the origin. Also illustrated is the fact that if the cursor is now shifted to the left, it will be out of the range of the display.

<u>4-Bit Data Bus Operation, LCD4.ASM</u>

This program is similar to LCD8,ASM, with the only difference that the 4-bit data bus option of the LCD module is implemented. The program goes through the same steps to demonstrate the various features of the LCD module. Although the subroutines are accomplish the same tasks as their counterparts in LCD8.ASM, there are occasional differences to handle the different data bus width.

The higher 4 bits of Port 1 is connected to the data lines DB4-DB7. Each data line is individually assigned to a Port bit. Thus, by changing the assignment, for example, bits from several Ports may be mixed to serve as the data bus to the LCD module. Bits 0-2 of Port 4 are used for the control lines RS, R/W, and E. These connections are defined by EQU pseudo ops:

```
; constants
 LCD_DATA  equ 090h ; port 1 is used for data
 LCD_DB4   equ 094h ; high nibble of port 1 is used for data
 LCD_DB5   equ 095h ; high nibble of port 1 is used for data
 LCD_DB6   equ 096h ; high nibble of port 1 is used for data
 LCD_DB7   equ 097h ; high nibble of port 1 is used for data
 LCD_RS    equ 0e8h ; p4.0 LCD Register Select line
 LCD_RW    equ 0e9h ; p4.1 LCD Read / Write line
 LCD_E     equ 0eah ; p4.2 LCD Enable line
```

The configuration specified during the function set operation and the entry mode are also defined.

```
; system instructions
 Config    equ 28h  ; 4 bit data, 2 lines, 5 by 7 character
                    ; matrix
 entryMode equ 6    ; increment cursor, do not shift display
```

The commonly used control words to manipulate the cursor and display are defined as in LCD8.ASM. The two subroutines wrLCDdata4 and wrLCDcom4 are the counterparts of wrLCDdata and wrLCDcom of LCD8.ASM. The major difference is that the data bits are placed one-by-one on the data bus. Furthermore, two data transfers of 4 bits each are required to complete a command or data transfer operation.

The routine rdLCDstat returns the status byte (Busy Flag and the Address Counter) in the accumulator. This routine should be revised if the Port bits assigned to the data bus are from different ports. The implementation in LCD4.ASM assumes the high nibble of Port 1 is the data bus. The operation reads the port twice, saving the first byte read. The two nibbles are then combined to construct the status byte. If bits from several Ports were involved, each Port need be read, constructing the status word bit-by-bit.

The routines crShLf4, crShRt4, dspShLf4, and dspShRt4 shift the cursor or the display to the left or to the right the number of places given in the accumulator similar to the 8-bit versions. The only difference is that these routines call wrLCDcom4 instead of wrLCDcom.

The routine pulseEwait4 sends a high pulse on the Enable (E) line and then continuously interrogates the Busy Flag (BF) until BF is cleared. It differs from its 8-bit version counterpart pulseEwait in the way it checks for the status of the Busy Flag.

The routines initLCD4, prtLCD4, setCGRAM4 and placeCur4 are the same as their 8-bit data bus version counterparts in LCD8.ASM except that they use wrLCDcom4 instead of wrLCDcom.

The routine resetLCD4 differs from its 8-bit version counterpart. The first 3 steps of reset are as in the 8-bit version. Then the control word that specifies the 4-bit option is sent to the LCD module. Note that this control word is still sent under the 8-bit data bus convention.

LCD4.ASM goes through the same steps as LCD8.ASM to demonstrate the various features of the LCD module.

Please remember all of the above sample software is written for the 8051 microcontroller. Demos for the 16-bit family are available from our WEB site, and in Rigel Corp's CD-ROM

9. RIO-MP PARTS LIST

QUANTITY	PART	DESIGNATOR
CAPACITORS		
8	100nF	C6, 7, 14-19
1	1uF 50V	C12
1	10uF 35V	C11
5	47uF 16V	C1-C5
1	220uF 16V	C13
2	330uF 16V	C8, C9
1	2200uF 35V	C10
RESISTORS		
2	220 OHM 1/2W	R3, R4
1	470 OHM 1/2W	R1
1	470 OHM (10g)	R2,
2	10K (6 gang)	R5, R6
1	10K (10 gang)	R17
9	5K POT	PT0-PT7
2	20K POT	R15, R16
DIODES		
9	RED LED	D2, D4-D11
2	7 SEG DISPLAY	D12, D13
1	BRIDGE	D3
1	1N5821	D1
MISC.		
1	DIP SWITCH	S1
10	SM P BUTTON	PB1-PB10
3	54 PT 3M PART	JP5, JP8, JP9
1	SPEAKER	SP1
1	SM T BLOK (2)	JP6
1	COIL	L1
1	KEYPAD	JP10
1	LCD DISPLAY	JP11 (or HD1)
1	BREADBOARD	BB1
HEADERS		
2	1X2	JP12, JP13
1	1X7	JP7
1	1X12	JP2
0 or 1	1X14	HD1 (or JP11)
0 or 1	2X7	JP11 (or HD1)
1	2X16	JP1
1	2X20	JP4
1	2X25	JP3
ICS		

1	LM386	U5
1	ULN2803A	U2
2	ULN2066	U3, U4
1	LM2575	U1
SOCKETS		
1	8 PIN	U5
2	14 PIN	D12, D13
3	16 PIN	S1, U3, U4
1	18 PIN	U2

10. TOP OVERLAY AND SCHEMATICS