Direct Digital Synthesis (DDS) is one of the more prevalent methods used to generate a frequency agile signal today.

The material in the following explanation was taken from www.ehb.itu.edu.tr/~eepazarc/ddstutor.html. If you want to search the Internet for more information on DDS, I suggest you use the full name as the search criteria otherwise you will get lots of hits on dentistry!

STAND-ALONE DDS UNIT

One of the ways DDS differs from both phase locked loops and mixing techniques is that the output is generated by a digital-to-analog (D/A) converter. There are four basic components to a DDS system:

1. Crystal oscillator
2. Phase accumulator
3. Look-up table
4. D/A converter

The crystal oscillator defines the highest frequency capable of being generated by the system. The DDS covers an operating range limited by sampling theory (Shannon, Nyquist). The highest practical output frequency is about 45% of the crystal oscillator frequency. One of the main advantages of a DDS system is that the method of constructing the output signal is almost entirely digital, and the precise amplitude, frequency, and phase are known and controlled at all times. Among the advantages of a DDS system are:

1. Very fine tuning steps
2. Very fast switching time
3. Phase continuous frequency change
4. Low phase noise

The November '05 issue of *QST* had an article on a DDS device controlled by a PC. After reading it, I determined that it might not be very difficult to build a stand-alone unit. I now have a DDS unit which is controlled by an RCM3610 by Rabbit Semiconductor. The DDS unit has the following characteristics:

1. A two line LCD for a menu display
2. Push button switches to step through the menus
3. Powered by a 7-15 VDC wall wart
4. The following is programmable:
   - Frequency in 1 Hz steps from 1 Hz to 10 MHz
   - Phase
   - Wave form — sine, triangle, square
   - PSK and FSK
   - Frequency sweep with pause
   - Amplitude of sine and triangle waveforms

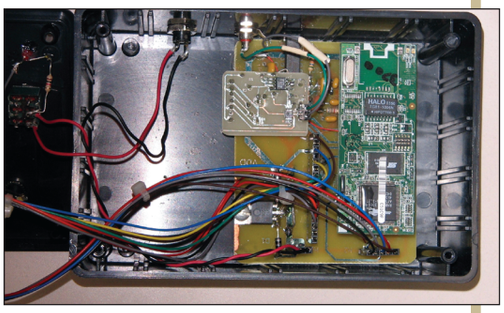
The system consists of two printed circuit boards. The first is the main board with the processor, LCD, switch interfaces, and the DDS circuit. The second board has the programmable attenuator and output buffer.

There are three push button switches in the design. Only two are used to navigate the menu system. PB3 is not used — it is available for future expansion. SW2 — a toggle switch — is used when the PSK/FSK mode is selected. It is the method of switching between the two frequencies and/or phases. This method allows you to put a connector in parallel with the switch in order to implement an external method of control.

The Microprocessor

Refer to Figure 1. I chose the RCM3610 as the control device

■These photos show the DDS unit.



mainly because I am quite familiar with it. It is programmed in both C and assembly language. It has quite a few parallel output bits, as well as several serial I/O ports. The parallel I/O is used for the LCD, switches, and Chip Select signals.

Two resistor packs terminate unused inputs, as well as providing pullups for the switches. When the microprocessor on the RCM3610 comes out of its reset state, all of the I/O pins which can be inputs are set as inputs. All inputs on a CMOS device should be terminated to either ground or VDD. Floating inputs can cause localized damage to the device.

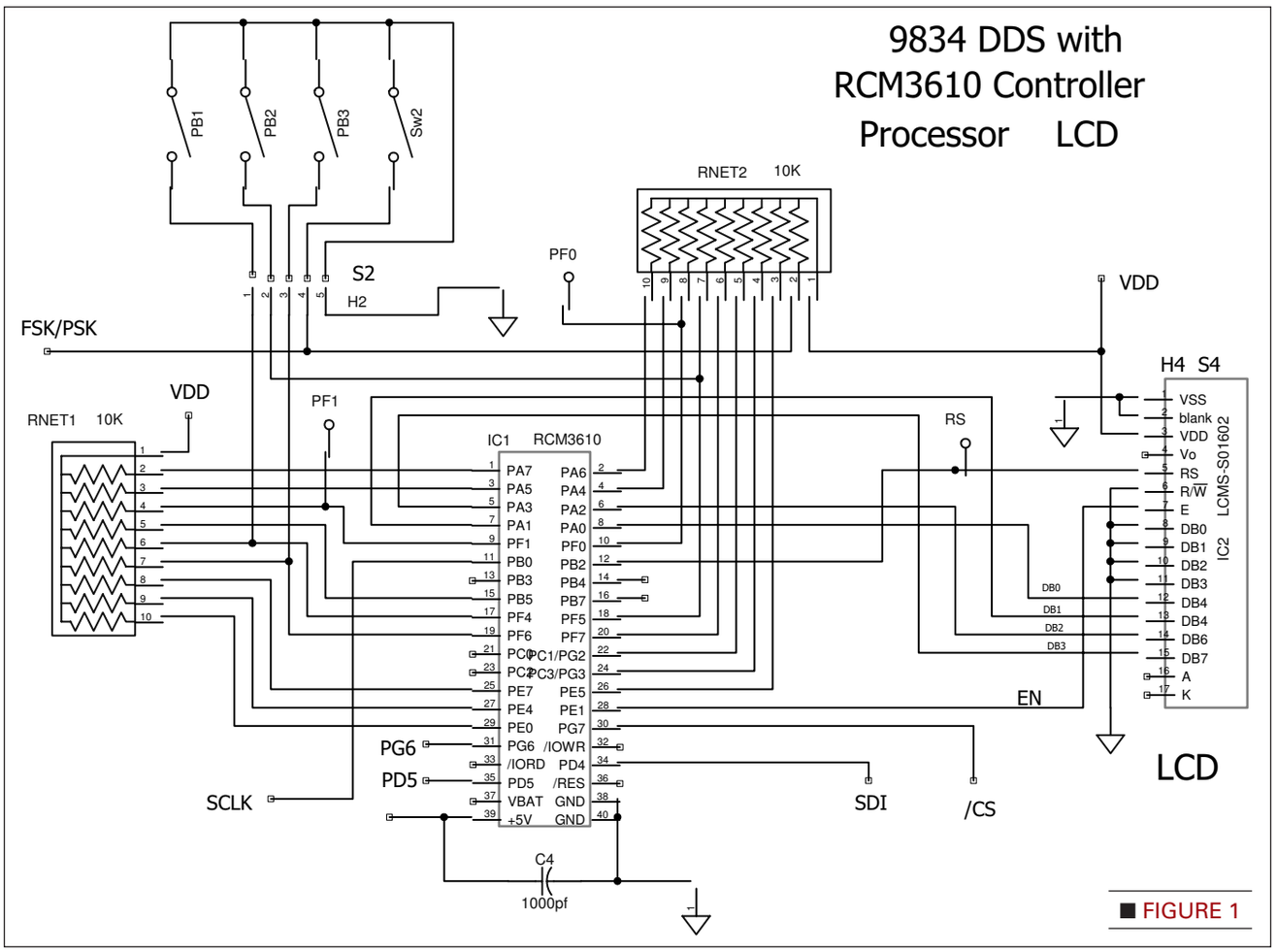
The interface to the LCD uses a feature of the processor which allows you to expand the parallel I/O capability by up to 64K bytes. This feature enables the program to access the external devices in essentially the same way as you would access the internal parallel I/O ports. Parallel Port A becomes the data bus, Parallel Port B bits 2-7 become address bits, and bits from Parallel Port E are I/O strobes. Since I am never reading anything from the LCD, its interface only needs to use a single address bit (PB2) in order to select between either the control register or the data register.

The interface to both the DDS and attenuator ICs use SPI (Serial Peripheral Interface). This is a relatively high speed serial communications method which is used by many peripheral ICs. You will find many other ICs which use it — A/Ds, D/As, memories, etc. The April '06 issue of *NV* has a good article discussing SPI.

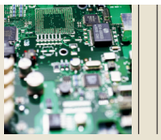
In the general case, each IC requires four signals: Clock, Data In, Data Out, and Chip Select. Neither of the SPI devices in this system sends any data back to the processor so they do not have a Data Out. Each device must have its own Chip Select, the other signals are shared. Only the device with the active Chip Select will communicate with the controlling processor.

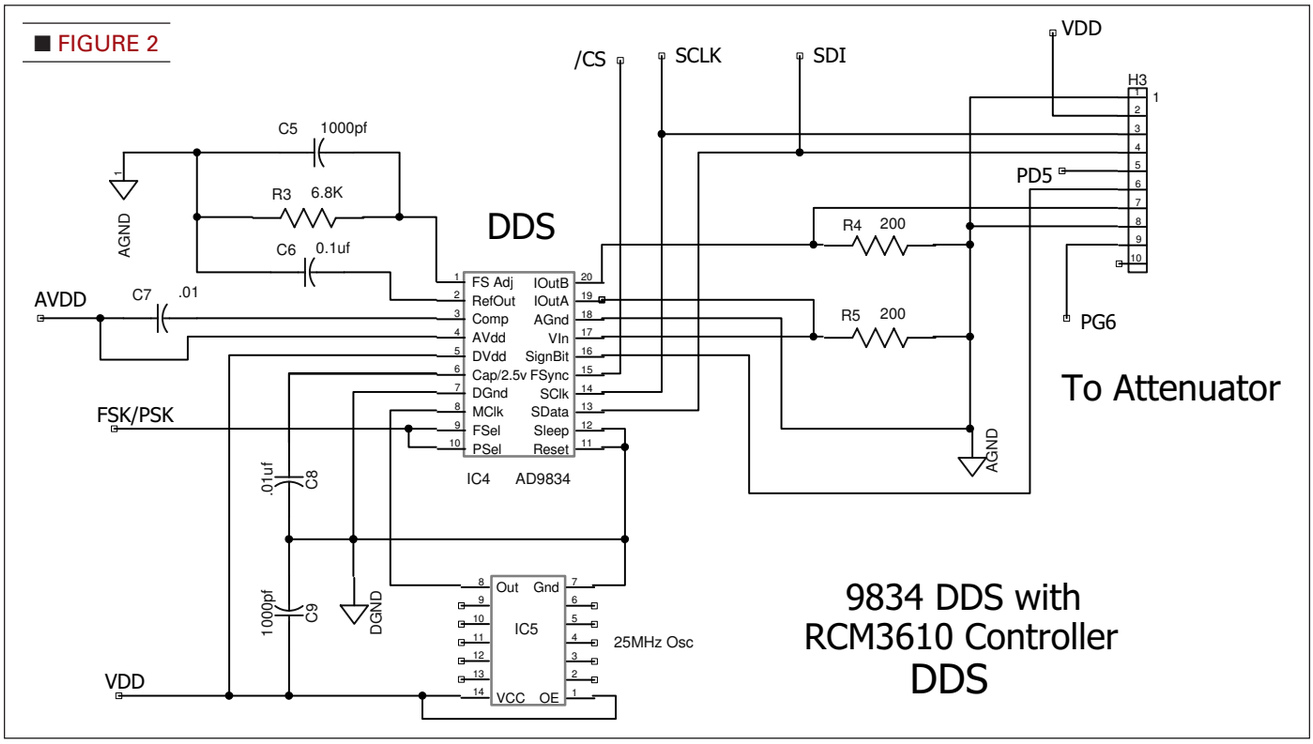
The LCD

The LCD has an S6A0069 controller which is compatible with the industry standard HD44780. I chose to use the four-bit interface to the LCD because I eventually want to



■ FIGURE 1

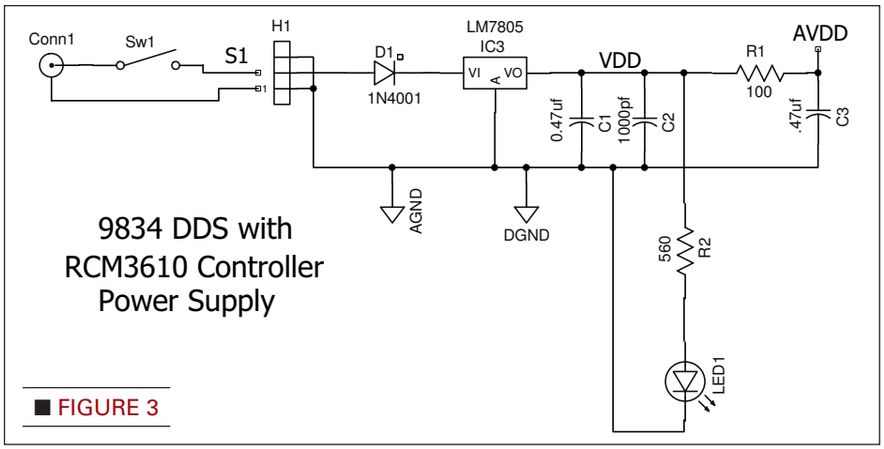




■ FIGURE 2

try to build the device using a low pin-count PIC. You can see from the schematic that there are two control lines being used: Enable and Register Select. The Control Strobe from Port E bit 1 is set up as a Write Strobe to drive the Enable input. The Register Select is driven from Port B bit 2 which is I/O address bit 0. I am not using the read capability so I have the Read/Write grounded to permanently enable its Write operation.

I chose a two-line, 16 character device in order to get a low-cost display. I also wanted to choose an LCD which can be easily obtained by anyone who wants to build the unit. However, I have since found some two-line, 20 character LCDs which are relatively inexpensive. For those who want to enhance the menu system, it would be fairly easy to use a four-line LCD instead. The pin-out for most character LCDs is common so you can probably use almost any unit, as long as it uses an HD44780, or equivalent, and is at least two lines of 16 characters each.



■ FIGURE 3

The DDS

Refer to Figure 2. The main output of the circuit comes from IOutB which goes to the Attenuator board. This output can be either a sine or triangle wave signal. The output of the internal D/A is 3 mA full scale. With a 200 ohm load (the recommended value), this yields a 600 mV peak-to-peak signal.

IOutA drives an internal comparator, the output of which can be steered to the SignBit output. This output is a square wave which can be programmed to select from among three signals: comparator output, sign bit of the internal D/A, or sign bit divided by two. The current system design does not make use of this signal, but it is available on the interface connector to the Attenuator board for anyone who wants to use it.

The main clock — which is used to generate the output signal — can be any frequency up to 50 MHz. I chose 25 MHz mainly because I was originally going to use an AD9833, a lower frequency device. However, I had trouble obtaining one from Digi-Key. The 9834 was readily available, so I

changed my design to use it instead.

The DDS can generate any frequency up to one half the main clock (Fosc) in steps of Fosc/2^{28}. This is because the DDS uses a 28 bit phase accumulator. With the 25 MHz oscillator, this yields steps of about 0.1 Hz. The highest practical sine wave output is about 1 MHz. Higher frequency signals do not look much like sine waves.

The Attenuator Board

My initial design did not have any means to control the amplitude of the output signal. I decided to implement an amplitude control circuit after showing my prototype to a few friends at work (Figure 4). The circuit consists of three sections: an inverting amplifier to get the signal up to a higher level, a programmable attenuator, and a voltage follower.

The inverting amplifier has a gain of approximately five — fixed by the ratio of R8/R7. Since the signal is applied to the inverting input, I had to add an offset circuit — R9 and R10 — to insure that the output signal stays in the linear range of the op-amp. With these component values, the output voltage ranges from about one volt to four volts. To keep the design simple, I decided to not implement any kind of adjustments.

The attenuator consists of a programmable potentiometer and voltage follower. The AD5200 is programmable in 256 equal steps with an end-to-end resistance of 10K. The program allows you to set the "wiper" to any of the 256 positions. This yields a step size of 3V/256, or approximately 12 mV. It would be fairly easy to modify the program to allow you to program in dB or millivolts. By changing the ratio of the gain resistors, you can easily change the maximum peak-to-peak voltage. If you change the gain, you will need to change the value of at least one of the offset resistors. An option which may be of interest is to make the maximum output voltage 0 dBm. When driving 600 ohms, 0 dBm is 0.7746 volts. This is equivalent to about 2.19 volts peak-to-peak, which is very close to the output voltage of my circuit. By reducing the gain to a factor of 3.65 (instead of 5), you will get a maximum output voltage of 0 dBm. You might also then want to change the output resistor (R11) from 200 ohms to 600 ohms, so you have a 600 ohm output impedance.

The Program

I have been using SPI devices for many years, but still managed to overlook a very important "feature" of SPI. There are four modes of SPI which define various phase relationships between the clock and data signals. As it turns out, the DDS uses one mode and the attenuator IC uses a different mode. At first, I did not realize this — mainly due to not reading the specs carefully enough. The program now changes the SPI mode when entering the attenuator function and then restores it to the mode required by the DDS when exiting.

The main parts of program are contained in two files: AD9834.C and AD9834.LIB. I also use a serial library — FSER.LIB — for the SPI communications. You will find that the source files (available on the *Nuts & Volts* website at **www.nutsvolts.com**) are fairly well commented and should be relatively easy to follow. Most of the code is in C but there are some functions partially in Assembly language.

The Menu

When not changing a parameter via the menu, the display will show the current frequency and phase values, as well as indicating the current mode and waveform. There are not enough characters in the display I used to also display the amplitude.

The menu system uses a two line, 16 character-per-line LCD and two push button switches. I found this to be adequate for my needs and it does allow you to easily set up the unit. Following is a brief description of the menu and how to navigate it. The main menu consists of the following options:

1. F0 — Change Frequency 0
2. F1 — Change Frequency 1
3. P0 — Change Phase 0
4. P1 — Change Phase 1
5. Waveform
6. Mode of Operation
7. Amplitude

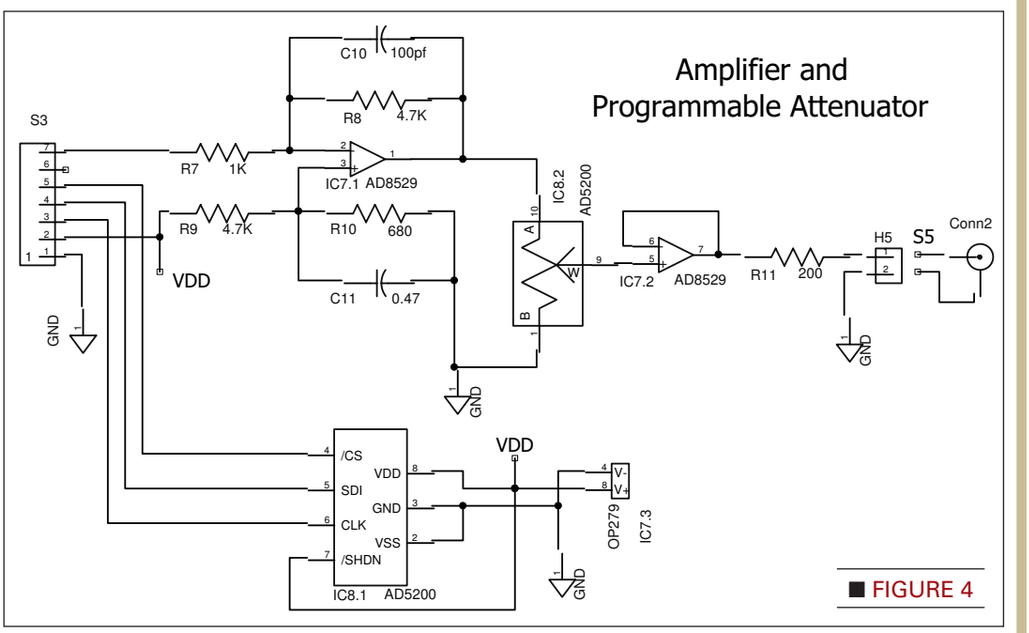


FIGURE 4

The Main menu is entered by pressing PB1. Generally, PB1 is used to step through the menu selections and PB2 is used to select the one you want. You can either hold PB1 or press and release it. Each press and release will advance to the next menu item. If you keep it pressed, the system will advance through the selections at one-second intervals. The menu will wrap around to the first item if you hold the switch past the last one. Pressing PB2 — after releasing PB1 — tells the system to select the currently displayed menu item.

For the Frequency, Phase, and Amplitude selections, the current value will be displayed and the cursor will be placed on the most significant digit. Press and release PB1 to advance the cursor to the next digit. Holding PB1 will cause the cursor to continue to advance through the digits with a one second delay between each until PB1 is released. Once the cursor is on the digit you want to change, just press and release — or press and hold — PB2 until the desired digit value is displayed. Using PB1 to advance past the last digit will cause the displayed value to be programmed into the system when PB1 is finally released.

The Waveform and Mode menu operations are similar to the Main menu operation. PB1 is used to step through the available options. PB2 is used to select the desired option.

The values in () are what shows in the status display.

The Waveform menu has the following options:

1. Reset (RE) — Turns the output off
2. Sine (SI) — Sine wave
3. Triangle (TR) — Triangle wave
4. Square (SQ) — Square wave the same as the selected frequency
5. Square/2 (S2) — Square wave generated by the sign bit of the D/A, effectively half of the frequency

The Mode menu has these options:

1. CW-F0 (CW) — Continuous wave of F0 and Phase 0
2. CW-F1 (CW) — Continuous wave of F1 and Phase 1
3. FSK/PSK (FS) — Enable FSK/PSK using SW2 or an external switch
4. Ramp 1 (R1) — Ramp from F0 up to F1, jump back to F0 and start over
5. Ramp 2 (R2) — Ramp from F0 up to F1, then ramp down to F0 and start over

For the Ramp modes, you are asked to enter both a step size and a delay value. The step size is in 1 Hz increments. The delay is in one millisecond increments up to 99 seconds. You can pause the Ramp at any time by pressing PB2. When paused, the LCD will show the current frequency of the ramp. Press PB2 again to proceed with the ramp. Pressing PB1 terminates the ramp process.

Construction Hints

The total cost of the parts is about $100. I can program the RCM3610 if you do not have the appropriate tools. All I ask is that you send me one along with return postage.

The two PCBs (printed circuit boards) can be purchased from FAR Circuits (**FARcircuits.net**) for $14 for the pair. They do not do plated-through holes, so you will have to insert some jumpers at the appropriate locations. You can find a complete set of PCB files (including the Gerber and drill files) on my website at **www.qsl.net/k3pto**.

There are no critical circuits in this design. I designed the board so that the output of the 25 MHz oscillator has a very short run to the DDS IC. In most cases, the resistor values are not critical. If you want a more exact output level, you may want to calculate better values for R4, R7, and R8 (see the text).

PARTS LIST

This indicates where I purchased the major parts. You can probably find equivalent parts from other vendors and possibly save some money in shipping costs.

LOC	PART NO.	QTY	VENDOR
❑ Circuit Boards		1	Far Circuits
❑ PB1,2,3	26622CK	3	Jameco
❑ Sw1,2	72160CK	2	Jameco
❑ RNET1, RNET2	24643CK	2	Jameco
❑ IC1	RCM3610	1	*Rabbit
❑ Socket for IC1	S7123-ND	1	Digi-Key
❑ IC2 (LCD)	67-1768-ND	1	Digi-Key
❑ IC3 (7805)	51262CK	1	Jameco
❑ IC4 (DDS)	AD9834BRU-ND	1	Digi-Key
❑ IC5 (Osc)	SE1714-ND	1	Digi-Key
❑ IC7 (Op-amp)	AD8529AR-ND	1	Digi-Key
❑ IC8 (Potentiometer)	AD5200BRM10-ND	1	Digi-Key
❑ H1,2,3,4,5	103341CK	2	Jameco
❑ D1	35975CK	1	Jameco
❑ LED1	333877CK	1	Jameco
❑ S1,2,4,5	103157CK	2	Jameco
❑ Pins for S1,2,4,5	100765CK	45	Jameco
❑ S3	S4107-ND	1	Digi-Key
❑ Conn1 (2.5 mm)	202358CK	1	Jameco
❑ Mate for Conn1	190537CK	1	Jameco
❑ Conn2 (RCA)	159484CK	1	Jameco
❑ Mate for Conn2	229921CK	1	Jameco
❑ Box (6x4x2)	270-1806	1	RadioShack

*Rabbit Semiconductor — **www.rabbit.com**

All resistors are 1/4 watt, 5%

❑ R1	100Ω
❑ R2	560Ω
❑ R3	6.8K
❑ R4,R5,R11	200Ω
❑ R7	1K
❑ R8,R9	4.7K
❑ R10	680Ω

All capacitors are monolithic ceramic

❑ C1,C3,C11	0.47 µF
❑ C2,C4,C5,C9	1,000 pf
❑ C6	0.1 µF
❑ C7,C8	0.01 µF
❑ C10	100 pf

Working with SMD ICs can be a challenge, especially the attenuator IC. I strongly recommend that you have a soldering iron with a very small tip and some very thin solder. You should also work using a lamp with a magnifying lens. I have a lamp with a 3X magnifying lens, but sometimes wish it were 5X. When soldering the SMD ICs, the method I use is:

• Tin the pads on the PCB — make sure there are no solder bridges!

• Align the IC on the pads making sure ALL the leads line up.

• Use a strip of plastic electrical tape, cut thin, to fasten it in place.

• Use very thin solder and an iron with a very thin tip to solder it.

• Use solder wick to remove any solder bridges.

You do not have to use the same method of construction as I did with the headers and sockets. I prefer them so that I can easily take apart the system pieces. You can solder wires directly to the PCB instead of using any of the header/socket pairs. If you do use the connectors, you will need to have either a crimp tool for the pins or a pair of small needle-nose pliers to use. I used pliers and did not have any trouble. Both the headers and sockets need to be cut to size. This can be done easily with a utility knife.

I cut the opening for the LCD using a Dremel tool. As it turns out, I cut my opening larger than required because I wanted the LCD to be close to the top surface of the box.

You may need to drill a new mounting hole for the voltage regulator. Some 7805s are different sizes with the hole further from the pins. I did not use a heatsink, but you can if you wish — there is room for a small one.

The crystal oscillator has pin 1 closest to the edge of the board.

Make sure you solder both sides of the components which have PC traces on the top. It helps if you do not mount the leaded components right up against the PCB.

When mounting the sockets, make sure you do NOT mount them right up against the PCB so that you can solder the pins which have connections on the top side.

I soldered 6″ wires onto the LED and then soldered the wires to the PCB. This allows you to mount the LED just about anywhere on the cover. Try to drill the mounting hole just large enough for the body of the LED and not so large that it allows the bottom lip to pass through. I usually use Goop to glue LEDs to a plastic lid.

LCD: Note that the schematic shows an extra pin at location 2. The correct wiring for the LCD to the connector is shown in Table 1.

LCD	Connector
1, 5, 7, 8, 9, 10	1 (gnd)
2	3 (+V)
3	No connection (VEE) this can be used for contrast control if desired
4	5 (RS)
6	7 (E)
11	12 (DB4)
12	13 (DB5)
13	14 (DB6)
14	15 (DB7)

■ TABLE 1

I "daisy-chained" the ground connection at the LCD using a long piece of bare wire with sleeving pushed onto it between pins 1 and 5, and 5 and 7.

You may need to drill mounting holes in the PCB, depending on how you choose to mount it in your enclosure. **NV**