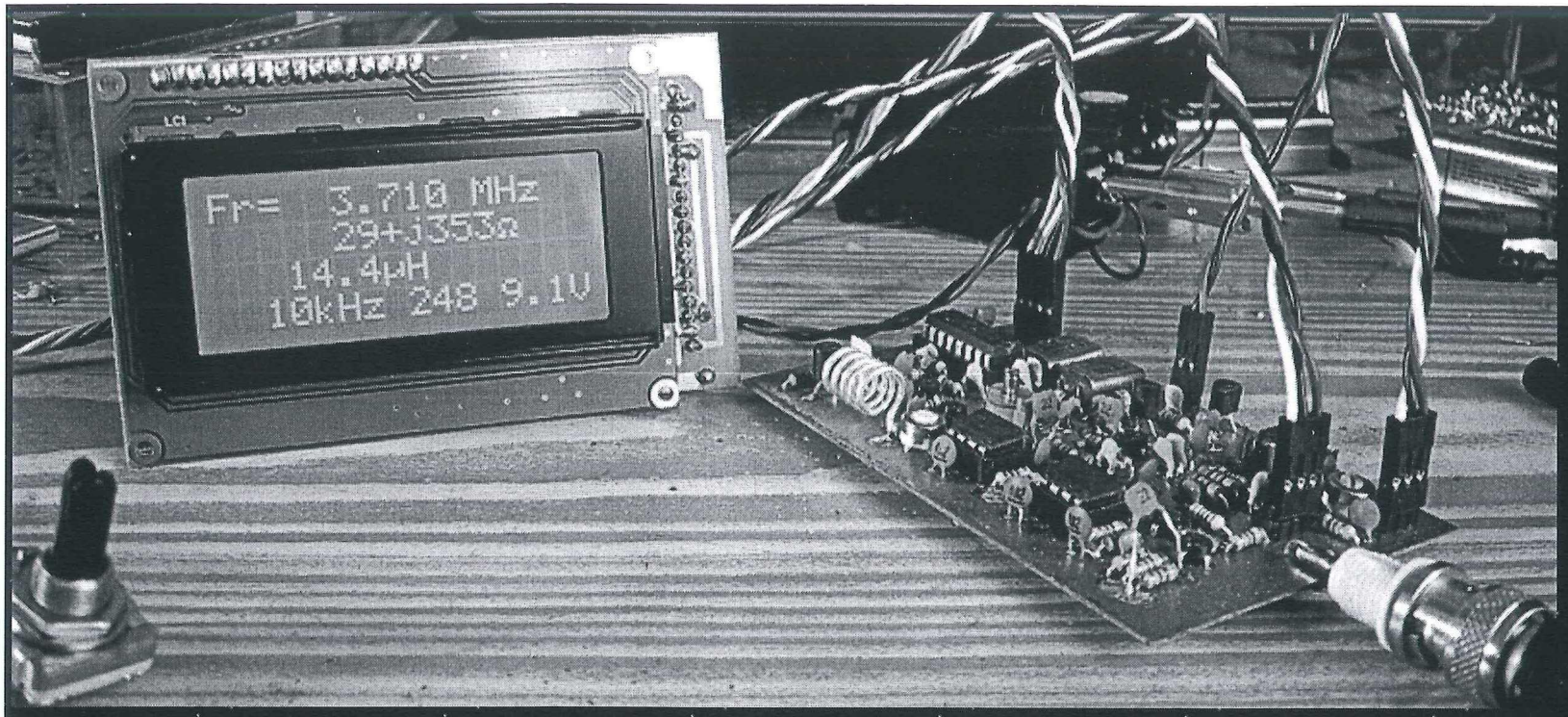
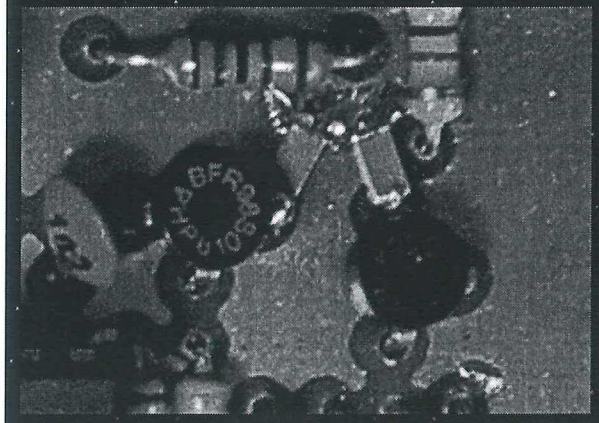
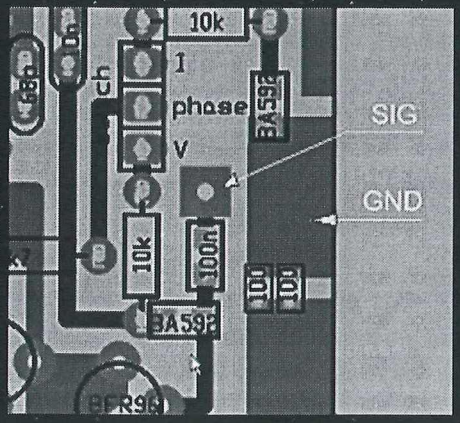
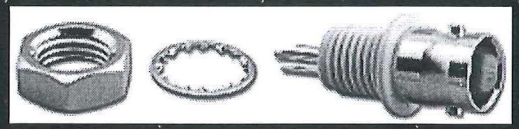






I used an isolated panel mount BNC connector. The picture on the left shows how to connect it to the PCB.



Basically the collectors of all transistors need to be decoupled directly to the ground plane using 1n or 10n chip capacitors. (far left) These are not shown on the PCB layout.

Modification

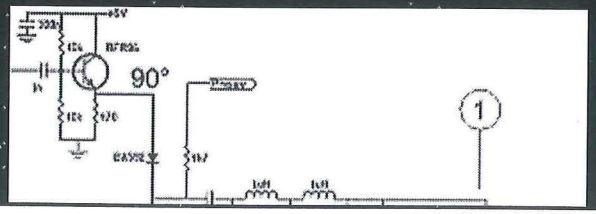
A slight modification on the controller board is needed to run version 2. This has to do with the way some of the settings are stored in EEPROM. In V1 frequency and back light level were written to EEPROM every time these settings changed. This is not very elegant since the number of times you can write to EEPROM is limited (it will ware out). With the introduction of the Mega168 it was possible to generate an interrupt on any of the I/O pins. One of the pins (PC0/ADC0) was already used to monitor the battery voltage through a 2*10k voltage divider. Although this pin is setup as an ADC input it is still possible to read its logical level and generate an interrupt when the logical state changes. So the settings are now stored in an interrupt routine just after the unit is switched off. Therefor we need to keep the controller alive long enough to write to EEPROM. This is done with a 100µF storage capacitor at the output of the 5V regulator. To prevent reverse current flowing through the regulator a diode needs to be inserted in series with the regulators input.

Testing and alignment

The controller PCB can be tested separately and should be tested first. The first time the controller is powered up (well it has to be programmed first of course) the display will show a strange frequency and step size (000 kHz) and the back light will be dimmed. First set the step size, change the frequency and set the back light level. Then switch off and on again to see if these values were retained in EEPROM. If not, increase the value of the storage capacitor. All functions should work as will be evident on the display.

Now connect the HF PCB (please do use connectors) and test the Synthesizer first. If the MC145170 is initiated properly by controller you should be able to measure a 10kHz pulse train on pin 9 (Fr). Then check the VCO and make sure it covers the whole frequency range from 49-78MHz. If not compress or pull the coil windings. The coil is made from 1mm silver coated wire, 5 windings on 6mm ID 10mm long and centre tapped.

Subsequently test the 48MHz oscillator and mixers. The frequency at the mixer outputs should be equal to the displayed frequency. Adjust the VCO output level to get 300mV pp at the output of the Low Pass Filter. (indicated by 1)



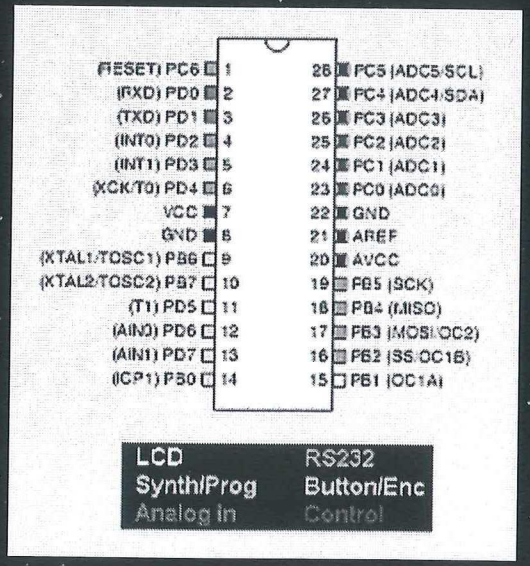
The best way to check the 90° phase shift is to trigger a scope to the output of the analyser and measure the output of the LPF while you switch the <PHASE> control signal between 0V and 5V. The LPF output should shift 90°. You can use the trimmer capacitor to set it to 90°. It is not very critical. If you hard wired pin 27 of the controller to the 4.7k resistor disconnect it at the controller side and apply 0V and 5V to it.

To set the potmeter at the input of the detector: observe the signal at the output of the first amplifier (connected to pin 24 of the controller) With a scope set on 2V/div and DC coupling and no load connected you should see a baseline of around 2.5V (half supply voltage) with excursions (positive or negative depending on the frequency) about every 100ms. These excursions should be within the range of 1V to 4V (absolute DC voltage)
If not reduce the input voltage of the detector. (test it for the entire frequency range)

Finally let the analyzer calibrate itself by switching it on while holding the function key. Do not release the function key until the calibration message disappears. Calibration should be done with no load connected to the analyzer. After calibration the measured values of R and X should be well over 10k. Now try some known impedances.

Firmware

The Firmware is written in BASIC using the Bascom AVR compiler. The controller is programmed and may be reprogrammed through the in-circuit programming connector on the controller board. A simple connection to the LPT port of a PC is all that is needed to program the controller. Please refer to the Sample Electronics cable programmer for more information. Note that the resistors are already incorporated on the controller board. The freeware version of Bascom AVR has a 4k memory limit. But I am not sure if this prevents you to use the programmer part to load the binary image into the controller. An alternative may be PonyProg and set it to "LPT1 using Avr ISP I/O" The controller should be programmed with the default fuse settings. (as it is shipped)



Open line measurements