Phase-locked VFO for VHF

Tim Forrester describes an advanced oscillator which forms part of a dual-band, multi-mode VHF transceiver.

The initial requirement was to produce a transceiver which could receive anywhere between 50 and 70.5MHz, and transmit in the bands 50 – 52MHz and 70 – 70.5MHz.

Inevitably, a microprocessor was included to do all the house-keeping work such as scanning the controls, driving the synthesizer, frequency display, and band switching etc. From the outset the design was about producing a transceiver with excellent radio performance, and not about a radio with average performance but with an all-singing, dancing microprocessor system (as is often the case).

First of all, I decided that the tuning must have the "feel" of a VFO, i.e. the minimum tuning step size from the synthesizer must not be greater than 20Hz. Any step size greater than this is too easily detected by the ear. To be

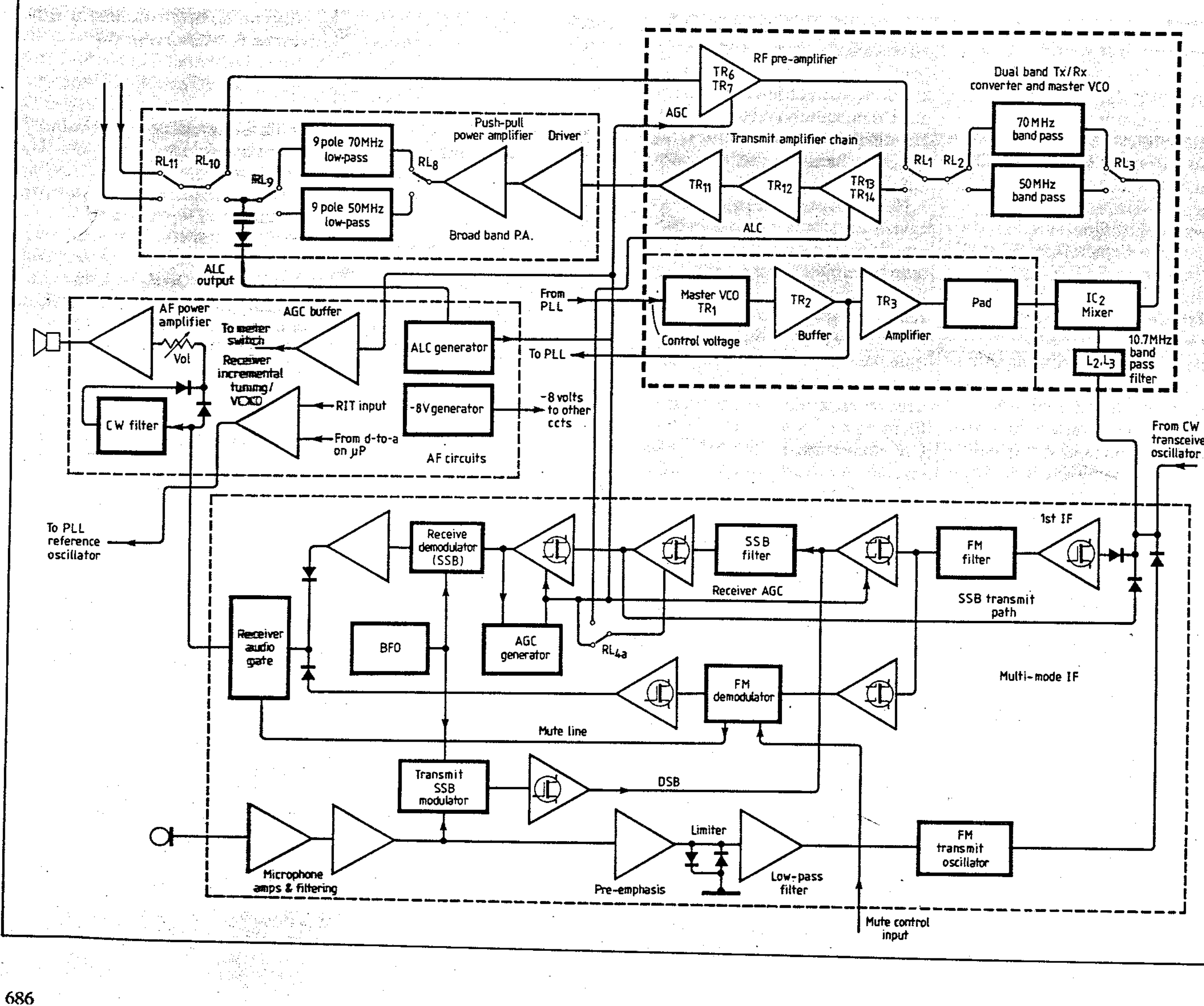


Fig. 1. Dual-band transceiver for 50MHz and 70MHz. The phase-locked oscillator and microprocessor control stages are outlined in Fig. 2. Aim of the design was to produce a transceiver with excellent radio performance.

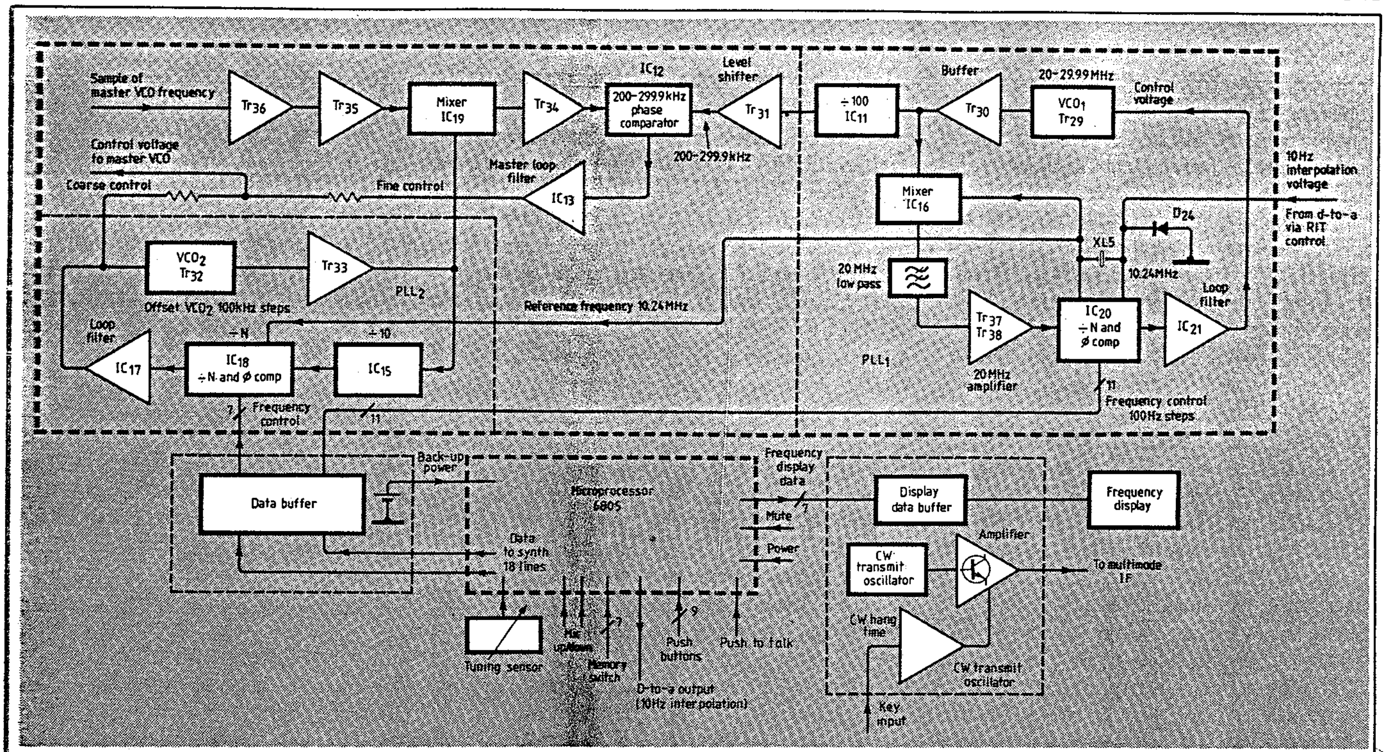
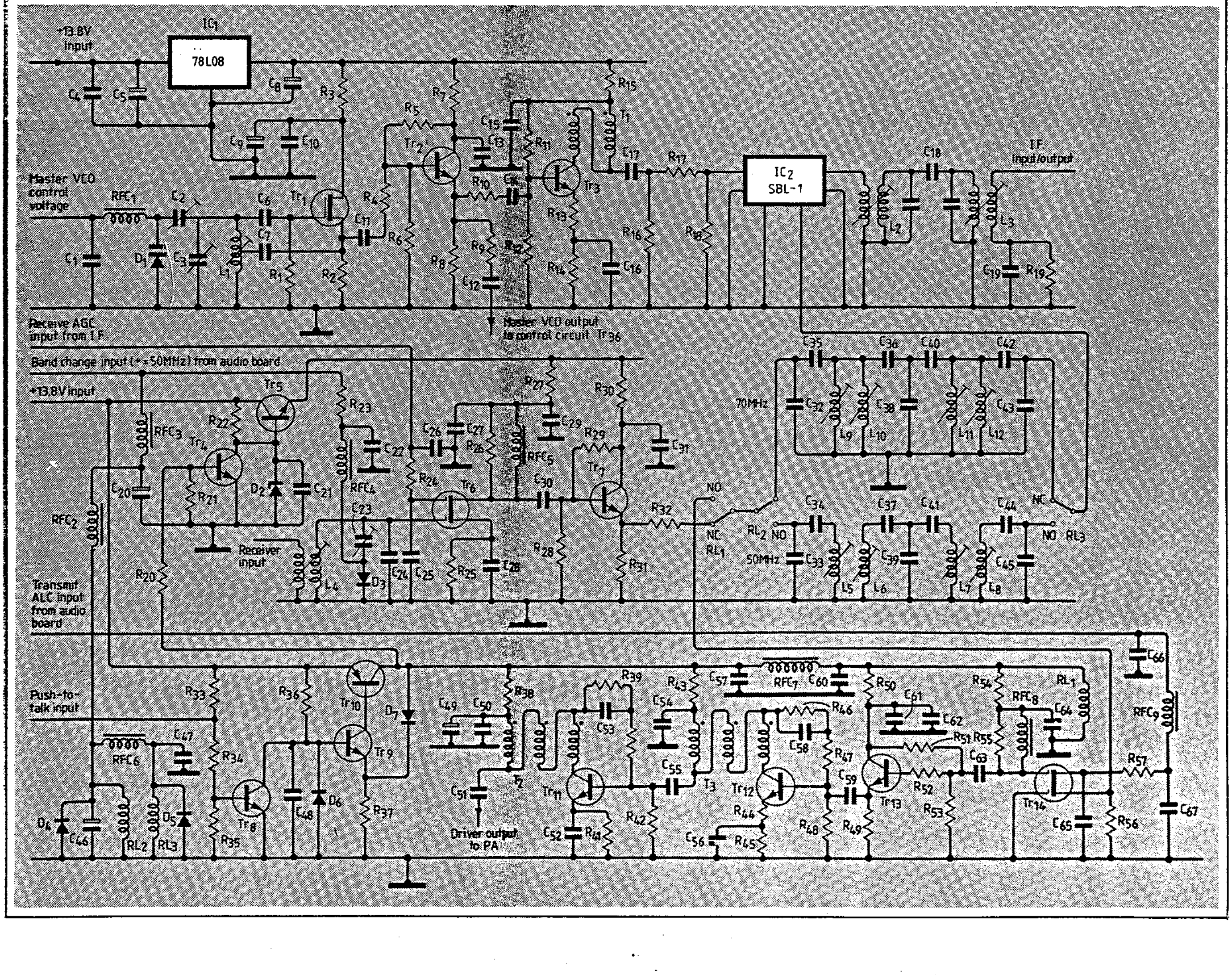


Fig.2. PLL, microprocessor and CW transmit oscillator for the dual-band transceiver. The processor is a 6805.

Fig.3. Dual-band transmit-receive converter and master VCO.



sure of a smooth VFO-like tuning response, I eventually decided upon a resolution of 10Hz as being a good compromise between complexity and resolution.

In the early stages of the design of the PLL, I considered a direct digital synthesis (DDS) approach. This would have had the benefits of very low phase noise and a very good tuning resolution, perhaps down to 1Hz step size. Unfortunately, though, at the time of starting the design the cost of fast digital adders and the other associated digital circuits made the use of a DDS oscillator prohibitively expensive. However, the cost of chips for DDS is falling rapidly. Plessey Semiconductors has announced an integrated circuit DDS device, which is capable of operating at up to 500MHz with a switching time of something around 10ns. No doubt devices such as this will eventually replace most conventional PLLs; but at present this particular device costs about £600 and is not yet freely available as a production item.

In this design the PLL would, ideally, enable the radio to operate over the range of 50MHz to 70.5MHz, with no gaps in the coverage. However, to keep the design simple to align, and to avoid the use of tracking filters and other complications in the actual RF signal path, I decided to restrict the coverage to just the amateur bands. Thus the PLL could operate at around 60MHz and use low-side injection for 70MHz operation, and high-side injection for 50MHz operation.

An additional benefit of restricting the tuning range was that the PLL had only to cover 3.4MHz in total to tune both the 50 and 70MHz bands. This enabled the PLL's performance to be optimized over a narrower bandwidth, thereby making its overall design easier. If the PLL had been required to work over the entire tuning range of 50 to 70.5MHz, inevitably some circuit parameters (such as VCO sensitivity) would have varied, causing the phase noise and/or the lock-in time to degrade.

For these reasons, combined with the need for tracking filters (to remove the unwanted in-band image response caused by the 10.7MHz first IF with its continuous coverage from 50 to 70MHz), I have restricted the tuning range so as to be able to use easily-adjustable bandpass filters to select the desired product from the mixer.

Trade-offs

Designing a PLL with a resolution of 10Hz and good phase noise performance is not too difficult if cost and complexity are not limiting factors. However in the present case certain compromises had to be made. The first was in the method of obtaining the 10Hz resolution.

It is fairly easy to design a synthesizer with a step size of say 10kHz, with reasonable performance, without resorting to complex multiple loops. This design, however, needed a resolution of 10Hz, which could not be achieved by a simple single-loop design.

After looking at several different schemes I decided on a basic digital PLL resolution of 100Hz, and to achieve 10Hz resolution by interpolation. This interpolation is achieved by slightly shifting the PLL's reference crystal. To understand how the resolution of 10Hz is obtained, it is best to break the operation of the PLL into sections **(Fig.2)**.

PLL1 is a conventional PLL operating between 20MHz and 29.99MHz in 10kHz steps. The only oddity in the design is the mixing down of the VCO signal from Tr$_{29}$ with the 10.24MHz reference signal. The purpose of this mixing process is to enable IC$_{20}$ to operate on the signal directly without the need for a prescaler. The output of this PLL is divided down by 100 in IC$_{11}$ to produce a signal of between 200 and 299.9kHz in 100Hz steps. This signal is used as the basic 100Hz digital increment in the PLL and is fed to IC$_{12}$, a

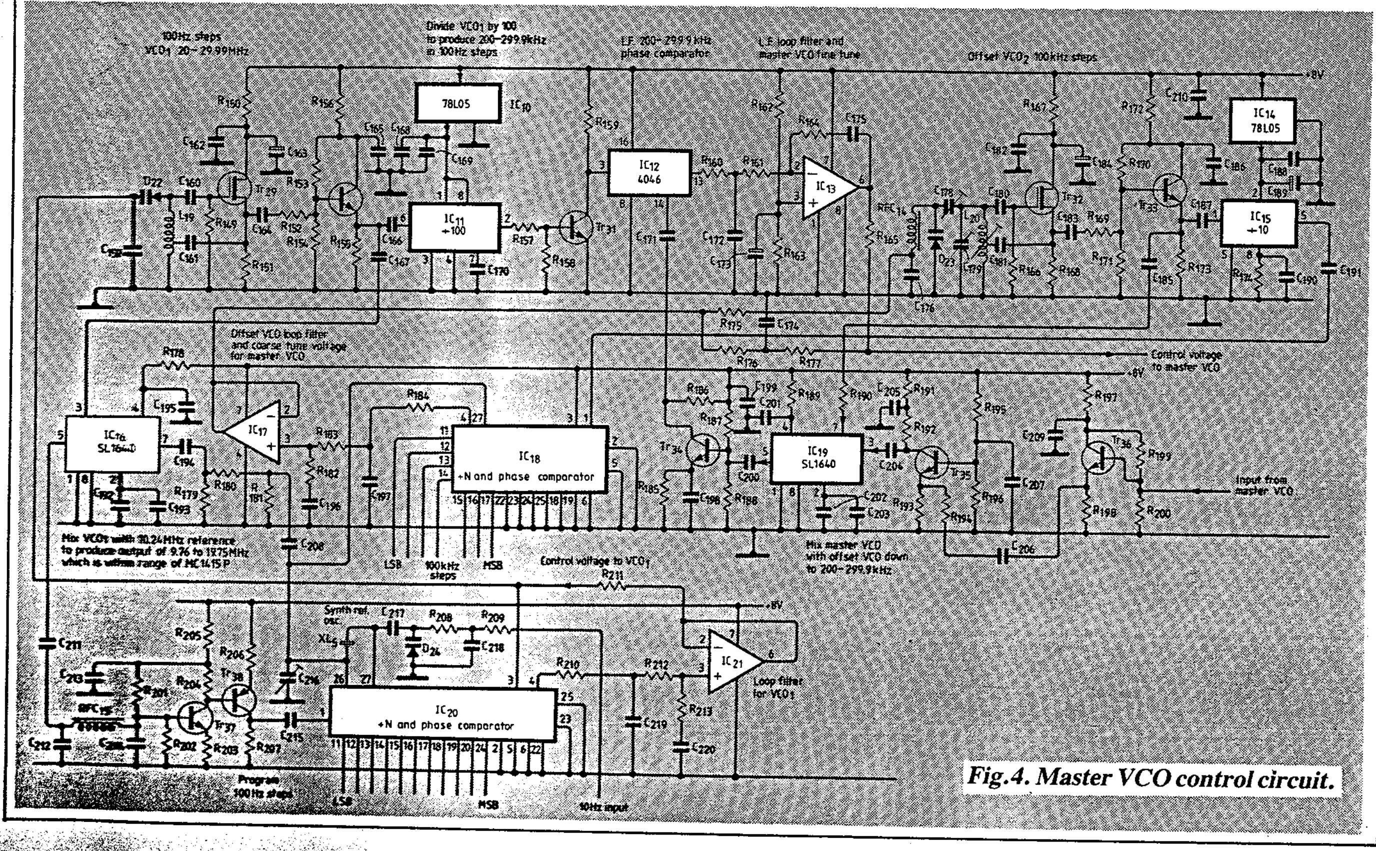


Fig.4. Master VCO control circuit.

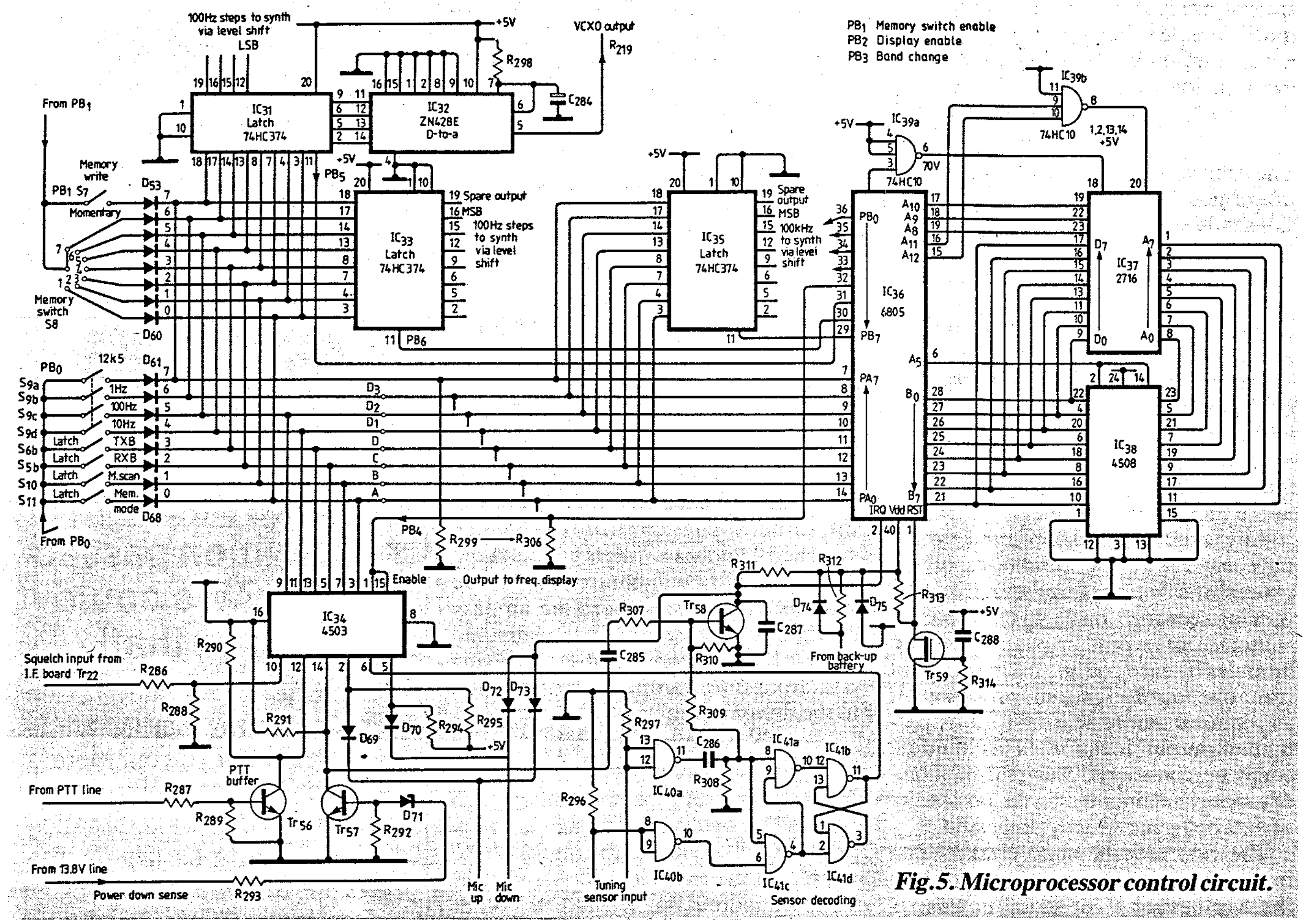


Fig. 5. Microprocessor control circuit.

low-frequency phase comparator operating in the range 200 to 300kHz.

PLL2 is another conventional PLL, but this time operating in 100kHz steps; it uses the same 10.24MHz reference as PLL1. Transistor Tr_{32} is a VCO operating in the 60MHz region, whose output is split between IC_{15} and IC_{19}. The output to IC_{15} is divided by 10 to be within range of IC_{18}, which is the divide-by-N and phase comparator device. PLL2's frequency is chosen such that it heterodynes with the master VCO to produce a signal in the range of 200 to 299.9kHz. For instance if the offset PLL is set to 61.5MHz, and reference frequency set by PLL1 is 200kHz, then the master VCO has to be on 61.7MHz to be in phase lock. If the frequency of the 200kHz reference (generated by PLL1) were to change by, say, 100Hz, then the master VCO would have to change by 100Hz to track it.

To ensure that the master PLL locks up quickly, a steering voltage from the VCO in PLL2 is applied to the master VCO such that the 200kHz phase comaprator has only to fine-tune the frequency. This steering voltage from the offset VCO also ensures that the master VCO is within the capture range of the 200 – 299.9kHz phase comparator.

The final 10Hz resolution is obtained by slightly varying the crystal reference frequency of 10.24MHz. An analogue control voltage is used to change slightly the bias voltage on D_{24}, which in turn shifts the reference frequency; this voltage is generated by an eight-bit digital-to-analogue converter on the microprocessor circuit board. As only a total of only 10 voltages are required (0Hz to 90Hz shift), only the four most significant bits of the D-to-A are used.

Control of the PLL

This method of heterodyning the master VCO with another PLL to produce a signal in the region of 200kHz for phase locking has been used for several years. But with the advent of single-chip PLL devices such as the Motorola MC145150 series, it has become increasingly easy to implement, thereby avoiding the masses of discrete logic which would have been previously needed.

Programming the PLL oscillator to the desired frequency is achieved by 18 parallel lines from the microprocessor via level shifters to IC_{18} and IC_{20}. Parallel programming is adopted in preference to the more usual serial method to make initial testing of the PLL easy without the need for a special serial interface. If serial programming is preferred, to lessen interconnections and improve overall reliability, then IC_{18} and IC_{20} could be replaced with IC type MC145155. A serial driver routine would then have to be added to the microprocessor program, because the number crunching in the processor is all parallel arithmetic.

With a design such as this, combined with an IF offset of 10.7MHz which can be on either side of the local oscillator, there is no easy or direct relationship between the eventual operating frequency of the radio and the data required to program the synthesizer. It would be possible to design some form of discrete logic circuitry to drive both the PLL oscillator and the frequency read-out, but this would be rather complex and inflexible. A much better solution would be to use a microprocessor.

Indeed, using a microprocessor allows a much greater flexibility in both the design and implementation of the control functions.

Software

The processor is a Motorola 6805, which offers good facilities for control functions, while at the same time being easy to program. In addition, since the processor can be single-stepped through its code, it is easy to debug the code by simply monitoring the state of the address and data lines.

Processor control was initially broken down into a number of basic modules which would form the basis for driving the synthesizer and frequency read-out. Subsequent subroutines would make use of these driver routines.

Any module or subroutine would have to restore the conditions of the processor's internal registers, before passing control back to the parent routine. Data would be passed between modules by each routine taking an input from one location in ram, processing it as required before writing it to its particular output location. This method could be considered wasteful of memory space, but does lessen the possibility of data being incorrectly processed.

The first module simply takes data from a location in ram and outputs it to the synthesizer. The data in ram is initially written to a particular location by the program itself, and contains the digits required to program the synthesizer to a particular frequency. Another module operates on the same source data as the synthesizer driver module. This module was designed to drive the frequency read-out, taking into account the IF offset and frequency band in use.

The frequency read-out driver module has a fair amount of number crunching to perform, and is therefore broken down into a number of subroutines.

For the tuning control I selected a cheap and readily available rotary encoder, whose outputs are two square waves in anti-phase. This encoder needs only a very simple logic circuit to detect the direction in which the tuning knob is being rotated and at the same time generate an interrupt to the processor. This enables the interrupt routine to update the frequency data and call up the driver programs previously described.

The interrupt routine of the tuning control also scans the front panel controls to determine in what step size the frequency is to be altered. The interrupt routine is in turn broken down into sub-routines. This was necessary as the tuning rate could be 10, 100, 1k, 10k or 12.5kHz per step. Routines are therefore needed to add or subtract these amounts to the data operated upon by the driver routines.

Included in the interrupt routines are limits on the frequency data, to ensure that the radio is not tuned out of band, and that the tuning wraps around at the band edges.

Also included are memory and scanning routines which enable the radio to scan spot frequencies on either band, automatically switching from band to band as required.

A further feature of the software is the ability to operate cross-band; that is to transmit on one band, then receive on the other. The control line which switched band pass filters in the radio was already being controlled by the program; and it was therefore an easy task to test the state of the transmit and receive band buttons, before outputting data from the appropriate ram location to the driver routines.

Microprocessor hardware

The 6805 microprocessor contains two PIAs, 112 bytes of ram and a clock generator. To interface the processor to the rest of the radio, one PIA is used as an input output bus, while the other PIA is used to enable various signals on to the PIA bus. Extra circuitry is included to de-bounce the rotary encoder and to generate interrupts when either the tuning knob is operated or the power supply falls below about 10 volts. If an interrupt is generated by low power supply volts, the processor is shut down and all present settings saved in the processor's internal ram.

When the processor is shut down it draws only a fraction of a milliampere of supply current, which is provided by a back-up battery. ■

Fuller details of this design will appear later in publications of the RSGB.