

running in the 3.5 to 7MHz region.

"What followed was an enjoyable four weeks of design, correspondence with G7IXH and G4GXO, and experiment – Ham Radio at its best! I find the combination of digital and analogue techniques most interesting and resulted in the fast-stabilising VFO of **Figure 4**.

"Software-wise, the VFO uses the PIC differently from the G4GXO method. The sample rate is determined by dividing the PIC crystal clock using an onboard counter. The VFO input is sampled directly on a clock edge rather than feeding the incoming VFO to an outboard counter and using a counter bit as a sample. The shift register and control XOR function is provided by an interrupt driven routine associated with the sampling operation. Provided that the code for the interrupt service routine (ISR) will run within the sampling period, its exact duration is of no consequence. Tests have shown that sampling rates around 1 to 2 kHz work very well which leaves 0.5-1.0 ms for processing; in computer terms that is quite a long time. Further, this obviates the need for high PIC clock speeds.

"A good VFO needs to be smoothly tuneable. The power of Peter Lawton's design results in the stabiliser having a tenacious hold, especially if the step size between lockup points exceeds 10 Hz. Even at 10 Hz, the pull of the stabiliser is just felt when small changes in frequency are intended. Going below 10 Hz, although easing the pulling problem, puts more demands on intrinsic VFO stability. Faced with these conflicting demands, I reviewed some of my previous ideas of using CMOS gates as a VFO. In the past such VFOs have proved to drift steadily, but I found this problem can be easily remedied by restricting the amplitude of the oscillator gate's output swing to less than saturation (ie rail to rail). This is done simply with a pre-set resistor in the gates feedback circuit; this can be adjusted to reduce the oscillatory swing to 4V or less with a 5V supply. The effect is magical.

"One other facet of PIC control annoyed me. Whereas it is easy to produce a lock-up step of around 5Hz using cheap mass-produced crystals, I wanted an exact 5Hz step. The ability to use lower-frequency crystals brought the solution, since a crystal intended for controlling serial asynchronous communications (Baud rate) can be divided to exactly 5Hz. Problem

solved! Interestingly, the step, once established, is independent of the actual VFO frequency – unlike my previous one where there has been some inter-relationship. Changing from 3.5 to 7MHz needs no other changes than tuning the VFO itself. Shifting frequency beyond a 2:1 ratio can, however, be troublesome, due to the need to control the varicap sensitivity.

"The 16F84 PIC can only support a 256-bit shift register due to its limited RAM. When my samples of 16F627/8 arrived, I adapted the code and expanded the shift register to 512 bits. This made a marginal improvement and is the preferred option.

"The end result is shown in Figure 4. Using a 2.4576MHz crystal as the clock, a counter dividing by 1024 and a 480-bit shift register results before in a 2400Hz sample rate and an exact 5Hz step. Lock-up is virtually from switch-on.

"One other minor annoyance appeared in the form of an occasional initial control voltage of other than half Vdd. The good idea in ('TT' many years ago) of using a pair of capacitors in series across the supply rails would normally be expected to divide the voltage equally at power up, but sometimes they did not. However, as most of the PIC's I/O capacity is unused, the software runs a routine at start-up to force the capacitor voltage to the correct value. It would be nice to be able to detect fast tuning, ie when the PIC is not controlling the frequency, and run this routine during normal operation, but I confess that so far this subtlety has evaded me.

A test point (TP) is provided to monitor the software operation and outputs a pulse of duration equal to the ISR and occurs at the sample rate. Construction is easy, but

AN ALTERNATIVE PIC STABILISER. Chas Fletcher, G3DXZ, writes: "TT has again spurred me into action. The February review of G4GXO's work in implementing G7DXF's idea of building a 'Fast Huff & Puff' frequency stabiliser using software led me to think it was time to review the stabilisers in my home-brew station. I took advantage of Ron Webb's web site (given in the February 'TT') to look at his software methods, but soon realised that the only PIC I had on hand, an old 4MHz 16F84, was not up to running his software routines. Not to be beaten by a minor problem and itching to get started, I decided to re-write the software and see if the F84 would work. My target was a VFO

