

VHF PLL Frequency Synthesiser

January / February 2010

JUMP TO THE LATEST CHANGES ON THIS WEB PAGE...

If you have visited my GPSDO web page, you will have seen references to my need to develop a "frequency synthesiser" (yes, Australian / English spelling in lieu of the American 'synthesizer') to be able to lock my radio transceivers and transverters to the 10.00000 MHz GPS-derived frequency reference and hence eliminate both frequency error and frequency drift with temperature. It turns out that I will need a multitude of different frequencies to be generated to achieve the desired outcome. For instance, both the Icom IC-7400 and the IC-718 I use will require 32.0000 MHz, my 70cm transverter will require either 69.66666 MHz (14.0000 MHz I.F. = 432.0000 MHz) or 67.33333 MHz (28.0000 MHz I.F. = 432.0000 MHz), and finally for now, the 23cm transverter will require either 96.0000 MHz or 95.916666 MHz for I.F values of either 144.000 or 145.000 MHz respectively. In practice it looks like I will require at least 3 discrete synthesisers to generate the desired range of frequencies. The question is really whether I can make the same basic design work at around 32 and 68 and 96 MHz, with little in the way of circuit changes.

If you are Google searching for a frequency synthesiser / synthesizer circuit / schematic for 69.666 MHz, 67.333 MHz, 96.000 MHz, 95.916 MHz, 32.000 MHz or even 30.000 MHz then you are probably at something approaching the right web page. While this page is purposefully blog-ish, it goes to demonstrate that building up synthesisers from scratch should not be taken for granted - as reading the text will show - and that there are areas which you must pay attention to.

Please note that I am neither a hardware nor software engineer so I don't have all the answers. I am a keen amateur / experimenter who is now travelling this route to do two things : (1) replace drifting crystal oscillators in other equipment ; and (2) learn more about PLL techniques.

I chanced across a sell-off of some Siemens TBB206G SMT PLL chips by one of our Australian electronics parts suppliers (Rockby (<http://www.rockby.com.au/>)) last year (stock code 26670 (<http://www.rockby.com.au/searchres.cfm?searchkey=tbb206g>)) , and they were relatively 'cheap'. They also had some TBB202 prescaler chips (stock code 37780 (<http://www.rockby.com.au/searchres.cfm?searchkey=tbb202>)) at normal price so I bought just a few of those. These parts all went into my " future plans " box and awaited both the time, availability and enthusiasm to develop the PLL project. Of course I was spurred on by the fact that I had the GPSDO running so had a reliable frequency reference to lock to PLUS I had already noted considerable frequency drift in the L.O. (local oscillator) in my 23cm transverter....

For those who are interested : TBB206 data sheet PDF (</~vk4adc/web/images/UserFiles/File/tvtr-synth/TBB206.pdf>) (1.3MB) & TBB202 data sheet PDF (</~vk4adc/web/images/UserFiles/File/tvtr-synth/TBB202.pdf>) (90KB)

I spent quite a while on the web trying to find out more on these Siemens chips and their use but information is fairly scarce because the chips are somewhat dated. I only found 2 or 3 articles that were of any help and what I developed was the result of interpolation of what was revealed there plus some data sheets for similar styles of devices from other manufacturers. It became obvious that there were any number of PLL-style chips around on the market in the past but current availability - and cost - was more-or-less an unknown. At the time of initially creating this page (Jan 2010) , the supplier (Rockby) had 1750 of these TBB206G chips available at \$4.50 each (in 1 off quantities, 10-up at \$3.60 each). { TBB202G - 136 available at \$4.50 each in 1-offs. } [Even at the listed prices, the PLL synth project is ultra-cheap to build.]

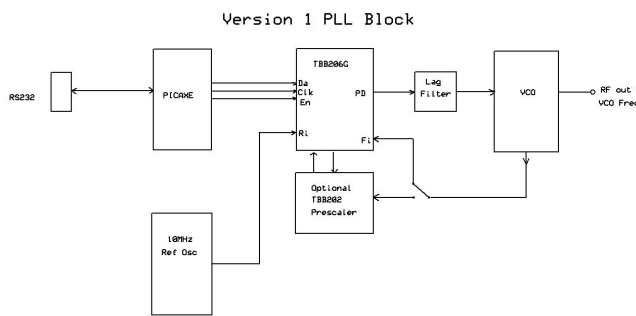
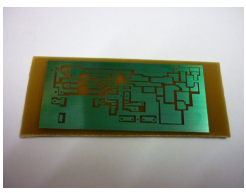
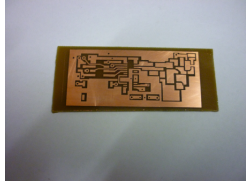
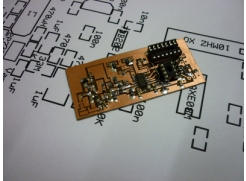
One of the things I noted in the web pages viewed was that 'everyone' seemed to be using a standard PIC controller but to me, that was a problem as I don't have a PIC-programmer and don't want to buy or build one. In general, you also generally need to be able to program in the specific PIC chip's assembly/machine language - too hard for someone my age and enough moral dilemmas anyway ! I had already developed some expertise using the PICAXE devices (see my transceiver sequencer project page) - these being a PIC with a pre-loaded version of BASIC within - so I decided that I would go with that method/device type. The TBB206G is a "3-wire" bus device so requires 3 control signals for operation : a Data line, a Clock line and a Enable line. I figured that I could probably get away with the baby of the PICAXE family - the 08M - which would give me one spare input pin on the chip although bigger capacity and/or higher I/O count PICAXE chips (eg the 14M) could be used in lieu. Just be aware that the available programming space in the 08M/14M/20M is virtually already used up by the current version of the software (these are all about 80 lines of code & limited to 16 GOSUB's and varying I/O pin counts) so an 18X (600 code lines, 255 GOSUB's, 5 inputs, 9 outputs) is probably a better option as a bigger/more flexible replacement. PICAXEs can be bought from MicroZed (<http://www.microzed.com.au/>) here in Australia.

I also investigated the available VCO "chips" and decided that with a simple L-C oscillator using a high frequency transistor, it was going to be easier to change the frequency range, be it HF, VHF or even UHF. It also solved the issues that I might not be working within the published specification range of these commercial devices. A conventional L-C oscillator was the firmly decided outcome.

That left the PLL's reference frequency source to be resolved in the hardware creation. My concern was that there might be times when the GPSDO was not going to be connected and I didn't want the whole synthesiser to fail because of that. I decided that I would add a 10.000MHz TTL-style crystal oscillator (14 pin DIL) to the PCB layout. In practice I found that I had a 10MHz TCXO which was pin-compatible so used that for all initial testing. Of course I could have just added a discrete 10MHz crystal oscillator but why build one when you can buy one prebuilt for under \$2.00. I have plans to buy a batch of 10 of the 10MHz TTL oscillators and do a track of frequency stability at some future time but for now, the drift at 10 MHz will hopefully be low. Of course I could revamp the PCB so that I could use a standard 10.000 MHz crystal - and it may come to that if these crystal oscillator units are not close enough to 10.000000 MHz or drift unreasonably!

The block of the prototype PLL was developed, then fleshed-out to a schematic and then to a PCB layout. The block diagram is fairly standard fare, the schematic just a little more complicated, the PCB layout relatively simple when using surface mount components.

Roll your mouse over the images for a larger view.

 <p style="text-align: center;">Version 1 PLL Block</p> <p style="text-align: center;">Version 1 / prototype block diagram</p>	 <p style="text-align: center;">V1 PCB created by using positive resist PCB, after exposure, developing & etching. Positive resist coating still on the PCB - the green colour</p>
 <p>The V1 PCB after the resist was removed. Note that even though the PCB material is double-sided, only one side is processed / etched with the 2nd side remaining as a full copper plane.</p>	 <p>This is the mostly-built V1 PCB, optional TBB202 prescaler not fitted. You can tell by the size of the 14pin DIL IC socket that the PCB isn't overly large...</p>

The remainder of the parts were soldered to the PCB, namely the 3-terminal 5V regulator and the VCO coil and the PICAXE had to be plugged into the 8 pin DIL socket, the crystal oscillator into the 14 pin DIL socket, the 5V points jumpered etc... In short, the physical part was completed. I even powered up the PCB and found that the VCO was free-running at about 76 MHz.

One thing that hasn't been mentioned before now is that the PICAXE, like any other PIC, has to be programmed to send the correct commands to the PLL chip. When I say the correct "commands", that means that the PLL's Data, Clock and Enable pins have to be activated in the correct order, for the correct time and with specific data formats to even hope to get the PLL to accept any data at all. I had the data sheet for the PLL chip and I found a web article on using the PICAXE-08M to send data to a TV tuner using the same 3 data lines : Da, Cl & En. In short I really had little else except a will to achieve a working PLL PCB.

It took several days but I finally had written some code using the PICAXE simulator that I thought would work. I programmed it into the PICAXE - no go. The problem is that you can put data into a PICAXE but you don't necessarily get useful data returned. This is even more true with the PLL - if you haven't the correct timing and data then the PLL just sits there as a 'blob' on a PCB and does nothing. I took a while with me re-writing code, sending it to the PICAXE before I had one version that started to see the Lock LED on the PLL blinking... I then knew I was getting close because the PLL had actually accepted some data. Within an hour I had conquered the timing and the data structure errors and actually had the VCO locking at the target 69.3333 MHz. Sure I had to hand-craft the binary data for the divisors but the thing was obviously doing close to what it should. I quickly saved a copy of the working PICAXE code and was seriously elated.

Of course, getting the code to work in this form was only step one. I still had to write code revisions to let me just enter a set of divisors into the thing so I could just send a new version to the PICAXE and I could get it to shift to a new frequency. The next night was the real breakthrough. I had the code working in a suitable manner, could get the frequency to change by adding or removing the optional jumper header pin on the spare PICAXE input. Not only that but by changing the coil and tuning capacitances to achieve suitable VCO tuning ranges, I had the PLL workable on frequencies from below 30 MHz right up to 101 MHz. This thing WAS going to work and by making multiple PCBs for the different projects, I would be able to use the same basic circuit concepts right through to cover my 32, 67-69 and 95-96 MHz requirements.

I set about putting the RF output under close scrutiny. The frequency stability at 69.333333 with the counter using the GPSDO as the reference gate timing showed the last digit (Hz) would occasionally shift 1 digit higher or lower (normal counter resolution) after the PLL had been powered on for a while. Even from dead cold, it shifted only some 8 Hz at 69 MHz - and that was all due to the drift of the on-board TCXO at 10 MHz !!!! The overall spectrum purity was looking good on my cheap-ex analyser too, even though the second harmonic was only about 20dB down. Please note that I do not have the test equipment necessary to make absolute qualitative measurements of close-in sideband or phase noise etc... I am using the Icom R7100 receiver to tune closely down the skirts of the main carrier and to evaluate the generated levels at the PD frequency either side (eg 96.000 and +/- 83.333KHz). I can then interpolate actual levels using my sig gen as a source into the receiver.

There was a problem when I listened to the signal on my R7100 receiver though - a sort of a buzz effect - realistically like a low level frequency or amplitude modulation - was clearly audible. I did a little experimentation and discovered that the lead/lag filter between the PLL chip and the VCO was certainly not using the correct values for a pure "carrier" output. I proved that because I could improve or worsen the effect by adding a parallel capacitance or varying the load resistance across the PD output pin. The input control line to the VCO's varicap showed up on the CRO - a low level periodic pulse was present & superimposed on the DC bias voltage.

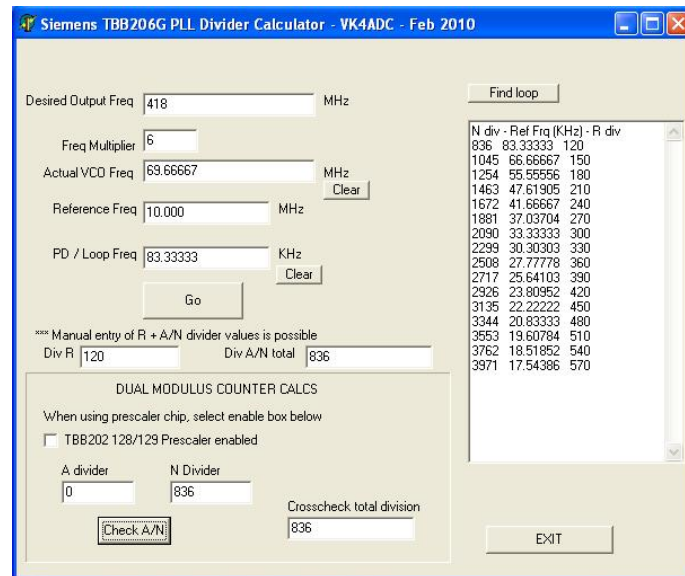
The VCO would lock to frequency and stay there but I was going to have to solve this one before it was actually useful enough to include in any practical application. A bit of research on the WWW helped me find a piece of software sourced from National Semiconductor which helps people design the simple lag filters for PLL applications (preferably using their brand of IC's). Details like the reference frequency, R or N divisor, VCO frequency, and like details are required for inputting into the National software to help develop the "lag filter" element values. { Available here as a ZIP file ([/~vk4adc/web/images/UserFiles/File/tvtr-synth/NS_PLL_Lag_Filter.ZIP](#)) or as a Self-Extracting ZIP File ([/~vk4adc/web/images/UserFiles/File/NS_PLL_Lag_Filter.exe](#)) , just run Setup.exe from the extracted archive }

Maybe it should be noted at this time that most of the synthesised frequencies I want to generate can be created with a PLL reference frequency of 83.33333 KHz, obtainable by dividing the 10MHz reference input by 120 (thus setting the value required for the PLL's R divider). The maximum divisor for the A & N divider (for me) is 1212 and the VCO frequency is just about within the specification for maximum input frequency

to Pin 8, the Fi input. This means that I only have to set the value into the N divider, the A divider is preset to 0 - thus making it a single modulus divider process. I have to admit that I wrote a "helper" program that I called TBB206.EXE { in Delphi for Windows } that allowed me to display suitable divider values for the R, A & N registers for any input reference frequency (though normally 10.0000 MHz) and any desired VCO frequency, whether single or dual modulus.

This is a screen shot of the TBB206.EXE application I wrote in Delphi to help me work with different reference and VCO frequencies.

Scroll over the image for a larger view.



I can also manually put in values of A & N divisors for any reference input frequency (10.000 MHz in the example above), different final frequencies (418.000 above) and frequency multiplication values (6 above) etc.

The RHS section below the Find Loop button allows me to find any PD loop frequencies that give integer divisors for any desired VCO and reference input frequencies.

The Dual Modulus section also takes into account when using the TBB202 prescaler and gives revised A & N divisors.

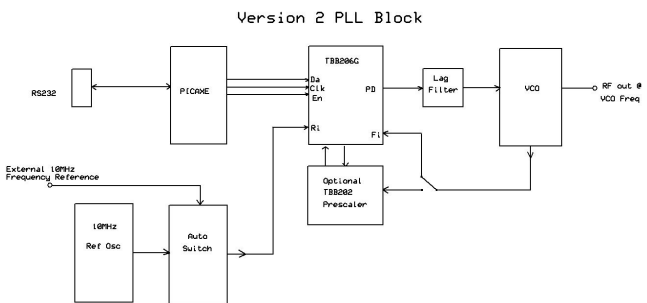
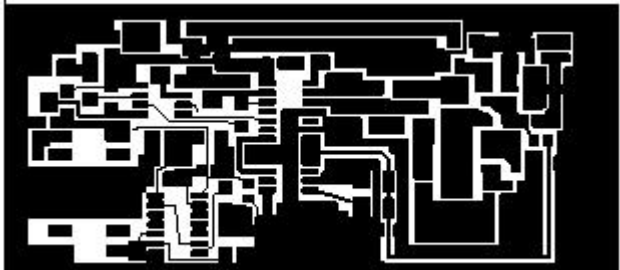
This software application is NOT going to be released into the public domain but if you want some help establishing such divider values for your PLL project, drop me an email and I MAY be able to help.

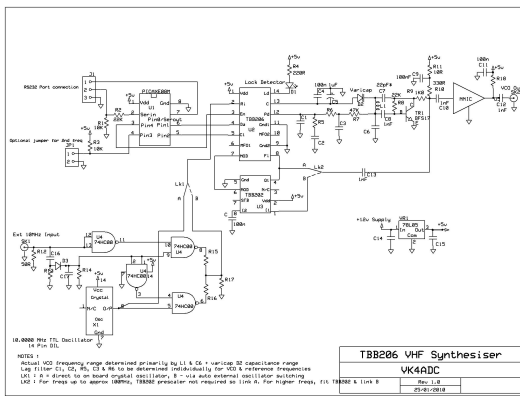
This prototype wasn't without another problem - microphonics. The R7100 receiver tuned to the VCO frequency would produce tapping, bumping, touching sounds when I either moved the PCB on the workbench or even tapped the workbench itself. I tried to discover where on the PCB the microphonic component(/s) were and after replacing most of the SMD capacitors with ceramic discs, I had eliminated most component possibilities. I changed the VCO inductor and still had the same problem. I started to wonder if one of the SMD resistors was microphonic but by putting firm pressure on each component in turn with an insulated alignment tool, and lightly tapping the free end, I found myself back at the PCB pads for the inductor. I replaced the 200nH inductor again with a coil wound from thicker enamelled wire, no slug tuning, and found that the physical microphonic sensitivity had reduced. I then liberally coated the coil with fingernail polish and let it dry. The microphonics then almost completely disappeared. Subsequently I coated the previous slug-tuned coil (the 2nd one used) with nail polish and re-installed it onto the PCB pads - again the result was basically no microphonics. The strange thing about this second coil was that it was already lacquered to the former, but maybe not quite well enough somewhere along the winding length.

A final note about the VCO tuning control voltage. I found that the PLL was best "behaved" when the "locked" control voltage was between about +0.3 to +1.0V. Note that the entire control voltage lock range is actually around -2.6V to +4.9V. By adjusting the slug in the VCO coil so that I had this +0.3 to +1.0V voltage at the output of the lead/lag filter (as measured with a digital multimeter), the PLL would always lock on power-on and had minimal PLL-style noises generated.

If you had no need to lock to an external GPSDO 10 MHz reference, there is no real reason why you couldn't use another TTL oscillator on a different frequency. I experimented with on-hand TTL oscillators at 10.000, 14.318, 16.257, 24.000, 25.175, 27.000, 28.322, 30.000, 32.000 and 33.333 MHz and by changing the R & N divider values in the PLL chip, I was able to get the synthesiser to work on or close to my desired frequency with each one. Just be aware that there is no frequency trimming with TTL oscillators and they do drift a bit from power-on. Some went up, some went down in frequency - none were spot-on and none stayed in one spot. If your desire is to just run a stand-alone synthesiser (no GPS locking) then the highest possible frequency within the allowable range for the TBB206 should be used (eg 30.000 MHz) keeping in mind that the harmonics of the reference oscillator may end up falling inside the desired receiver coverage. { Ditto the comments for the discrete crystal oscillator version }

While I was working on this first version, I realised that when built "for real", there were a few changes that I would want/need to make in version 2. The first point was that I would really like to maintain the onboard 10MHz oscillator feature but to automatically switch to an external reference if one was plugged in - no manual switching to be involved. The second was that I seemed to have a lot of "jumpers" supplying +5V to the different sections of the PCB and that needed to be tidied up. The third was that the output signal from the VCO back to the PLL divider, whether direct to the TBB206G or via the TBB202 prescaler could be tidied up. Finally there was the re-arrangement around the VCO coil connections so that a multitude of different coil formers could be used, with or without shielding cans. Then I came upon the fact that these PICAXE chips were also available in an SOIC 8 pin surface mount package. By shuffling parts on the PCB, I should be able to fit all of the components into the same area PCB. My thoughts also moved to doing an alternative PCB with the 14 pin DIL TTL oscillator removed and a standard quartz-crystal oscillator circuit substituted in it's place on the PCB.

<p style="text-align: center;">Version 2 PLL Block</p>  <p>V2 block diagram - not much has really changed at this level here except for the addition of the auto-switching of the reference oscillator signal.</p>	 <p style="text-align: center;">TBB206 / TBB202 / PICAXE 28M synthesiser</p> <p>V2 PCB layout / TTL oscillator Top layer only is printed Bottom layer is plain copper.</p> <p style="text-align: center;">Artwork was done in ExpressPCB (http://www.expresspcb.com/)</p>
--	---



(/~vk4adc/web/..t)

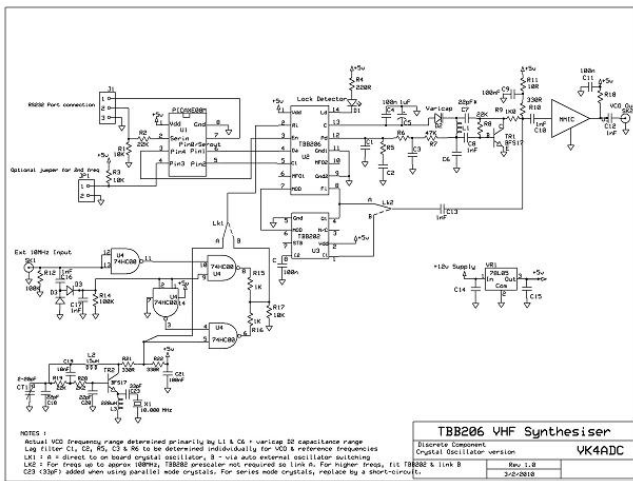
(/~vk4adc/web/..t)

V2 Schematic / TTL oscillator version.

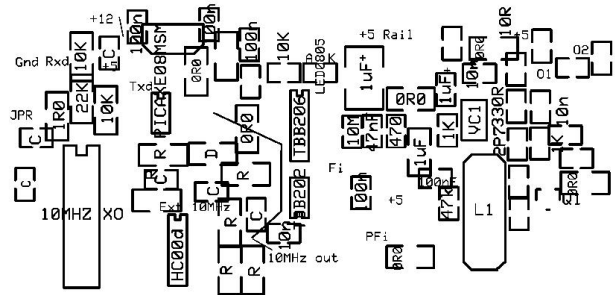
Note that several components do not have final values shown. These are mainly around the lag filter, particularly C1, C2, C3 and R5 & R6, which are determined by frequencies involved, division ratios etc and will vary somewhat from one band segment to another (eg 32 to 68 to 100 MHz).

Final values will be listed only at the completion of the entire project.

Schematic drawn with ExpressSCH
(<http://www.expresspcb.com/>)



V2 Schematic, version with discrete component crystal oscillator at bottom LHS.

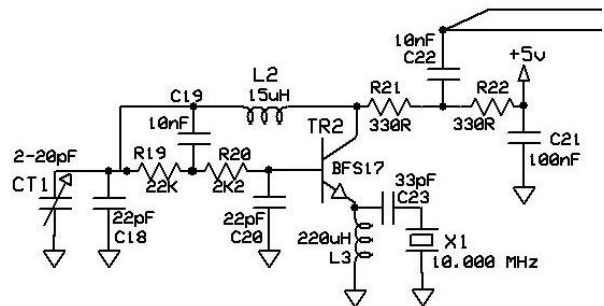


TBB206 / TBB202 / PICAXE 08M synthesiser
Jan 2010

This is the PCB overlay to suit the V2 PCB/schematic - TTL oscillator version
There are values on the lag filter components that had not been updated at the time of creating the graphic.

The only wire jumper is in the selection of the reference oscillator signal (ie LK1) - either directly from the TTL Oscillator output - or - from the auto switch circuitry.

Link LK2 is achieved using an optional 0R0 surface mount resistor and cutting a PCB track so is not visible as an actual wire link.



This is an extract of the schematic at left, showing just the crystal oscillator segment - a series-mode configuration.

C18 - a 22pF - was removed during experimentation but may be required for some crystals

C23 - a 33pF - was added to shift the frequency up for parallel mode crystals

The ratio of R21 to R22 (1:1 above) may yet need to be varied to increase the voltage levels into the HC00 gate

Details as per notes below.

10Feb10 - This circuit has now been altered a little.
New arrangement coming...

3 Feb 2010 : I am about ready to create the V2 PC boards having now more-or-less solved all of the major version 1 issues. I have two PCB layouts done, one with the TTL oscillator and a second with a discrete component quartz crystal oscillator as an alternative.

3 Feb 2010 : The V2 PCB that I chose to create first was the one with the discrete transistor crystal oscillator on it rather than the TTL oscillator version, this time using (and for the first time) RS Components 397-0053 (<http://australia.rs-online.com/web/search/searchBrowseAction.html?method=searchProducts&searchTerm=397-0053>) double sided FR4 positive resist PCB material, 203mm x114mm - about \$18.50 + GST = \$AUD20.35. I actually printed both version artworks (TLL osc & discrete) on the one inkjet transparency and it was only at the last minute that I decided that I would create the discrete version so that I could test the oscillator circuit itself and the reference auto-switching circuit. BI**dy hell some of those PCB tracks are fine... 0.25mm !

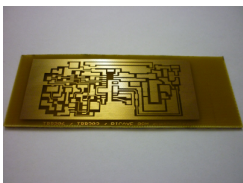
For details on how the PCB was actually produced, visit my page about *Producing PCBs from positive artwork and positive resist PCB material* ([/~vk4adc/web/index.php/general-projects/33-pcbs/76-producing-pcbs.html](http://vk4adc/web/index.php/general-projects/33-pcbs/76-producing-pcbs.html))

By the way, those fine (0.25mm) PCB tracks came out just great....

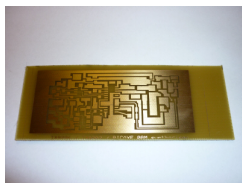
I built up the discrete crystal oscillator section of the PCB first and put the 5 volt regulator on the board, complete with bypass capacitors. I found 2 old 10.000 MHz HC-49 style crystals in my junkbox, presumably extracted from old computer boards in the distant past so gave them a try first. The oscillator worked ok, the frequency showing as 9.996xxx MHz so I tried the second crystal : 9.997xxx MHz. I quickly came to the conclusion that these were actually parallel mode crystals operating in series mode in a series mode oscillator circuit. I removed the 22pF SMD cap (C18) from across the frequency adjustment trimmer CT1 and the frequency rose a little - about 1.7KHz - but still low of 10.0000. Back in the "old days", we used to "pull" crystal frequencies by either series or parallel capacitance or inductance depending on whether it was a parallel or series mode oscillator and to shift up or down. I grabbed a 33pF ceramic disc cap and simply soldered it in series with the crystal and powered the board up again. The frequency was now 10.000016 and a quick adjustment of the trimmer brought it back to 10.000000 MHz. I watched the frequency while using the 10 second gate time on the counter (i.e. last digit displayed is .x of a Hz) and the frequency basically sat within 0.8Hz of the target. I subsequently replaced that 33pF disc with a 33pF NP0 SMD style, reset the trimmer cap to give exactly 10.000000 and watched the frequency for a while again - no real drift was noted, certainly it would be close enough for the "free-running" mode of the synthesiser. This series capacitor has been included on the relevant schematic above as C23 but can simply be replaced by a short-circuit when using actual series-mode crystals. C18 (22pF) may be required for some crystals to run on-frequency.

The next step will be to install the parts for the auto-switching of the external reference signal input and test that...

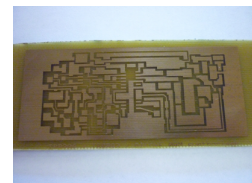
The next step will be to install the parts for the auto-switching of the external reference signal input and test that...



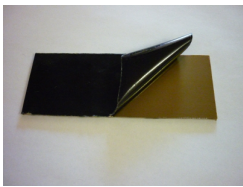
Production steps - After developing



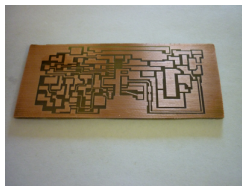
After etching



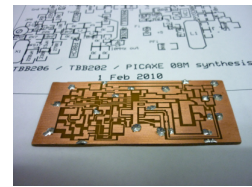
After cleaning off resist from main side



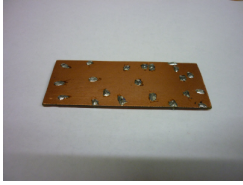
Removing protective cover from side 2



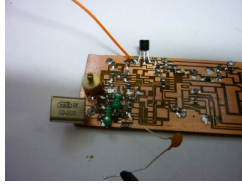
PCB after trimming & spraying with clear lacquer



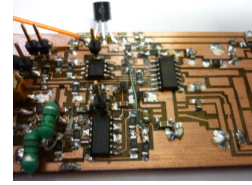
Solder-through "vias" to connect the earth lands on the top to the bottom ground plane



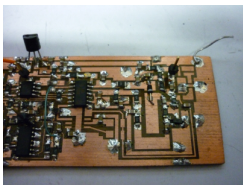
Bottom view with "vias" in place



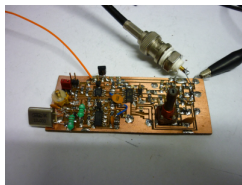
With components populating the crystal oscillator, regulator. Standard RFCs (green colour near crystal at bottom LHS) have been used until the SMD ones arrive. 78L05 regulator up near the +12V orange wire at top LHS.



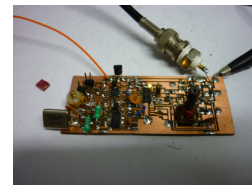
Some of the PLL components in place. The 8 pin SOIC chip at top LHS is the PICAXE, TBB206 PLL at centre top, 74HC00D 14 pin SOIC at bottom LHS.



Most of the VCO parts in place, tuning inductor excepted



V2 PLL board up and running at 96.000 MHz, PLL locked. The ceramic disc in the middle of the picture is the coupling from the HC00 mix output to the PLL reference input pin.



Yellow LED at centre top lights when the PLL is unlocked. The removed jumper header is visible near top LHS of PCB

4 Feb 2010 : The 74HC00D and parts were installed on the PCB and it *almost* worked properly first time around. The standard peak envelope detector's output voltage was marginal with the 10MHz external reference signal applied so it was quickly changed to a voltage-doubler style by replacing one resistor with another diode. I intentionally re-set the on-board trimmer so that the crystal oscillator was off-frequency by about 500Hz and that made the testing easier. Without external 10MHz, the frequency counter on the autoswitch output (across R17) displayed 10.0005xx and the counter changed to flick between 9.999999 and 10.000000 as soon as the external 10MHz source was applied. The final values for the parts around this part of the circuit have been updated on the schematic above.

5 Feb 2010 : The balance of the PICAXE, PLL & VCO parts were fitted to the PCB. Computer connected via RS232 port & PICAXE loaded with the PLL programming code. The only other "variable" is then the VCO frequency and whether the PLL can be made to lock. I fed the VCO's RF output into a BNC-T and then into both the frequency counter and my spectrum analyser and changed the turns on the coil until the VCO was running at about 90MHz. I then used the slug in the coil former to shift it up a bit - and when I got to about 92 MHz, the PLL jumped into lock at 96.000 MHz. The yellow 'lock' LED went out to indicate that as far as it was concerned, the PLL was locked. I used a DMM on the PD output pin and set the coil slug for about +1.2V then shifted the red jumper across the PICAXE input, the LED lit then the frequency shifted to 95.91666 MHz and the PLL lock LED again went out. I rechecked the PD output voltage to find it was about 1.5V. I experimented a little with slug position but had no problems getting a very positive PLL lock.

I re-programmed the PICAXE with divider values for the 69.6666 and 67.3333 MHz frequencies, added a few more turns to the coil and after adjusting the slug, the PLL locked positively at both frequencies. I was about to add more turns to the coil to try 32 MHz then had a minor inspiration - just add more tuning C. I tried a couple of different bridging values while watching the VCO frequency and ended up placing a 100pF directly across the coil to achieve a free-running VCO at about 30 MHz. I put new divisor values into the PICAXE again and it immediately locked at 32.0000 MHz.

With the feed from the GPSDO in place, the PLL lock was "strange" and after experimentation, found that the PLL reference input pin signal voltage was too high with this as the source. I ended up putting a small capacitor across the resistive-mix output of the HC00 and it was then operating correctly with either source. The on-board oscillator or the GPSDO both produced a counter reading exactly on the desired frequency and listening to the VCO output on USB on the R7100, swapping from one to the other caused an almost indeterminate change in beat note.

I also did a minor modification so that the bias was direct to the varicap rather than via the main coil and that change was achieved by merely unsoldering the 47K SMD resistor and rotating it 90 degrees then re-soldering. While playing with the varicaps & biasing, I tried another varicap diode in parallel and found that the VCO sensitivity changed from 2MHz/Volt to about 3MHz/volt at 96MHz. The varicap I used was a BB405B, a leaded style, which should give about 10pF variation in capacitance over 0V to +5V . One thing I determined absolutely today was that you cannot use the same PLL board at the 3 desired segments (32, 69 & 96 MHz) without changing at least one of the VCO components. Quite simply there isn't enough VCO range possible in the circuit used here because of the limitations in varicap capacitance values from minimum to maximum bias voltages.

The lead/lag filter between the PLL & the VCO still requires some exploration. Using values calculated by the National Semi's software, there was still some "buzz" apparent on the receiver. Lock tuning range became wider/better with an additional larger RC network lag filter across the PD line to ground, which was also accompanied by a reduction in the "buzz". I think since I am not "channel switching" (i.e it is basically always on one frequency) that a longer lock time is a better option (ie larger RC filter values) if the carrier noise levels are lower.

Finally for now, I measured the RF output level after the ERA3 MMIC amplifier (with a 47 ohm bias resistor to +5V) at 96.000 MHz : +6dBm (about 5dB below maximum rated output (i.e. +11dBm) so should not be being overdriven).

8 Feb 2010 : After a few days break away from the project, I have re-visited it to do a few measurements on the PLL board.

The following were done with divisors loaded for 100.0000 MHz operation :

TC1 adjusted for 100.000000 MHz displayed on the counter within a couple of minutes of power-up.

Output level at f_{VCO} = 100.000 MHz : confirmed at +6dBm (+/- 1dB)

Level at $2f_{VCO}$ (200MHz) : approx -20dBc

Level at $3f_{VCO}$ (300MHz) : approx -42dBc

Level at $4f_{VCO}$ (400MHz) : approx -65dBc

Level at $5f_{VCO}$ (500MHz) : approx -60dBc

Synthesiser products at f_{VCO} +/- PLL PD frequency (ie 100.0833333 & 99.9166666 MHz) : approx -58dBc (ie -52dBm)

Frequency drift/error over 1st 1/2 hour : -22Hz [+/-1 digit] @ f_{VCO} (100MHz) - (ie 99.999978 MHz) - however this was probably not a fair test as the room was being cooled down by an air conditioner from about 29-30C to 24-25C during the same period.

At 1 hour later, the frequency error was at -24Hz [±1 digit] (ie 99.999976 MHz)

Notes :

- The PLL was using it's on-board crystal oscillator & not the GPSDO as the synthesiser frequency reference.
- The crystal is a standard computer board crystal and not a high stability variety.
- Expected spec for typical crystals for this type of use is around 30PPM, radio communications spec is usually around 5PPM & sometimes even better.
- The PCB under test is not encased in any thermal insulation - it is simply sitting out in the open on the workshop bench.
- The spectrum analyser is not capable of being used for phase / sideband noise measurements.
- Harmonic & spurious levels were measured by using the calibrated output of a Marconi 2019 digital synthesised signal generator injected into the spectrum analyser input in place of the PLL signal to obtain reasonably close estimations of these signal levels.

There are a few component value changes on the schematics above that have yet to be altered to make them match up with the actual working circuit.

These details will be updated in the next week or so & this message will then be removed.

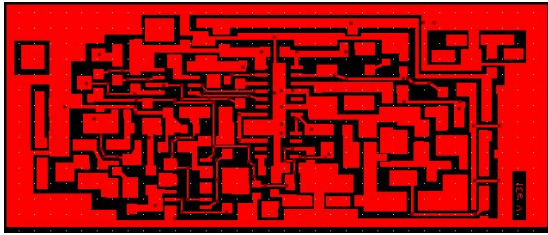
10 Feb 2010 : I started experimenting today with injecting the PLL V2 output into the LO chain of the 1296 transverter ([/~vk4adc/web/./23cm-EME72-tvtr.php](http://vk4adc/web/./23cm-EME72-tvtr.php)) in place of the 95.91666 MHz crystal. The PLL RF output was run via a short-ish length of RG174 coax (300mm) from the DC blocking capacitor at the ERA3 output to the emitter of the 2nd transistor on the LO PCB - ie the 2nd transistor in the Butler oscillator. The LO chain was producing around the same output level at 1151 MHz as when using the crystal but listening to the this LO signal on the R7100 on USB was like listening to a band of noise and no clear heterodyne was audible. Funnily, the direct PLL signal at 95.9166 sounded fine. It made me wonder about the stability of the rest of the multiplier chain on the LO PCB !

I dropped the PLL lead/lag filter back to the standard values and adjusted the PLL coil tuning slug (because the lock range reduced) and it started to sound better. It made me start to think about what other low noise VCO circuits could be used to see if it could be improved. I changed it to a more-or-less standard Vackar circuit by shifting around a few components and found that I now had to reduce the turns on the coil to get it to tune to 96 MHz and hence into VCO lock range. The output was definitely cleaner (in respect of noise) and I started to get a straight heterodyne while listening to the LO at 1151 MHz. At the same time, the signal from my RF sig gen at 1296.150 was now actually a heterodyne at 145.150 in the IF transceiver where it was just a band of noise before. I tried feeding the PLL RF into the base of this 2nd transistor too but it appeared to be overdriving it so I returned the feed to the emitter connection. It is still early days and there is still a lot of evaluation work to be done and minimal time, just at the moment, to try things. Revised VCO circuit to come to the web page after I have done more testing.

One thing that might have to happen is to replace the BFS17 VCO transistor with a BFR92A as these latter devices have a higher transition frequency (5GHz vs 1GHz) and lower noise figure (2dB vs 5dB), same pinout. This, theoretically, should result in even better noise performance. I haven't yet done that change because I want to finish my initial testing with the BFS17 first and get it as good as I can - then change the transistor and re-test to see how much difference it actually makes in practice.

13 Feb 2010 : Over the last few days I have been re-laying out the VCO section of the PC board in ExpressPCB, the new version now locally marked as a version 3 (V3). This new version does not have the long track on the board supplying the VCO frequency to the PLL N divider input (pin 'Fi') and is designed solely around the Vackar variable oscillator. The main advantage is that this board should be able to have the VCO run at something up to 1GHz and then the divide by 128/129 TBB202 prescaler will need to be fitted to provide the PLL with an input frequency within its operating limits. Of course the PLL can still be run at low frequencies like 30 to 100 MHz and just doesn't need the prescaler chip fitted.

The two previous boards (V1 & V2) are still complete and by making up the V3 board, I will now have yet another board to double-check the basic design on. The board was printed, developed, etched and partially built in the course of today and only a few components are still required to complete it. The overall board dimensions were altered only slightly (by about 6mm) to allow the fitting of a separate voltage regulator plus crystal heater assembly at that oscillator end of the board.. if required.

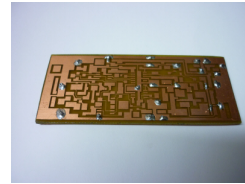


This is the V3 layout copper detail.

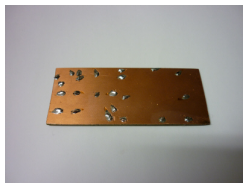
The main change is the VCO is now bottom RH corner across to middle RHS where in V2, it was at top RHS.

RF output is now dead-centre bottom instead of top RH corner.

The pads at the top RH corner are for an extra variable-use optional MMIC amplifier, requiring the fitting of the MMIC device plus 2 SMD 0805 caps & 1 SMD 1206 resistor. { The space was there, why not utilise it ??? }



V3 PCB top view after soldered wire-through "vias" in place

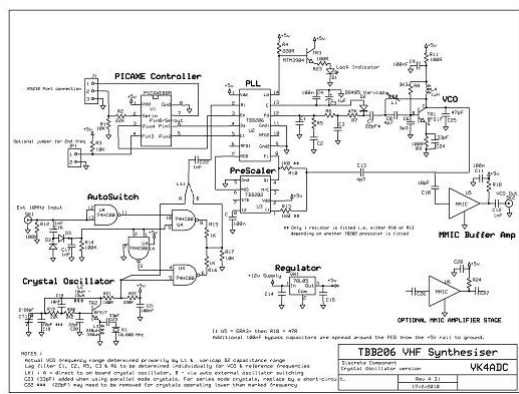


V3 PCB bottom view after soldered wire-through "vias" in place



Completed V3 PCB

The slug-tuned inductor protruding from near the bottom RH corner will eventually be replaced by a more appropriate style before being installed in any equipment.

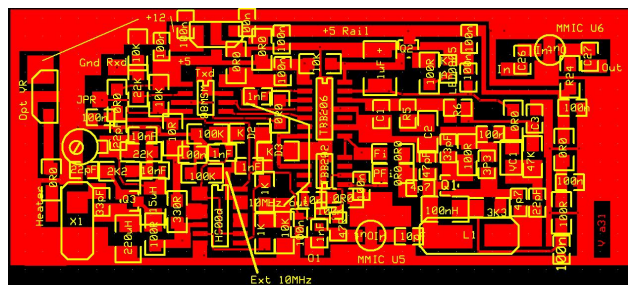


This is the latest update of the V3 schematic with the Vackar-style VCO (17/2/10)
 TR1 can be a BFR92A instead of BFS17.
 Do NOT use a low freq transistor for TR2.
 Increasing C7 (22pF *) value will give larger VCO tuning range
 For the 83.3 KHz PD freq, the lead/lag filter values are C1=330pF, R5=12K, C2=10nF, R6=150K, C3=33pF

L1 = 5T closewound on a 5mm OD F29 slug tuned former, 3mm long using enamelled wire 0.6mm diameter = 120nH with the slug out
 Using a BB405 varicap diode, the slug tuning VCO range is 75 to 125 MHz.

Click on these links for mid
 (/~vk4adc/web/images/UserFiles/Image/tvtr-synth/Freqsynth1%20a31a%20mid.JPG) and large
 (/~vk4adc/web/images/UserFiles/Image/tvtr-synth/Freqsynth1%20a31a%20large.JPG) images

This the V3 PCB component layout (17/2/10) showing the placement of the various components.



Note that there have been pads added at the LH end to allow the fitting of an extra separate voltage regulator for an optional crystal heater but these components are not shown on the schematic.
 C1, C2, C3 & R5 & R6 do not have permanent values marked as they will change for different PLL PD reference values.

Large image
 (/~vk4adc/web/images/UserFiles/Image/tvtr-synth/V%20a31%20layout.JPG)

16 Feb 2010 : The final few components were soldered into place on the V3 PCB this morning and power applied. The +5V was where it should have been on the PCB, the Lock LED was out (as expected), so I connected the frequency counter to the RF output - it displayed about 123MHz with the coil slug nearly fully out (minimum inductance) and that meant the VCO was running. Next I attached the flyleads to the serial port and loaded the PICAXE-08M with the synthesiser firmware. The Lock LED glowed dimly, again more or less as expected from the 2 previous versions. The divisors were set for 96.000 (no jumper) and 95.9166 MHz (with jumper) so I started to screw the coil slug in and the VCO frequency dropped and before long the Lock LED brightened and the frequency counter displayed 96.0xx MHz (I didn't note or write down the actual value). The error in the displayed frequency meant that the 10.000MHz crystal oscillator was off frequency by a 1KHz or so, so I adjusted the crystal oscillator trimmer capacitor until the counter read 96.000.

I measured the RF output level at +11dBm at 96 MHz for this version but there were a few final changes involved in the building of the V3 PCB that might account for the higher level (versus +6dBm for V2). The one change that was probably the main reason for the higher output level was the use of a BFR92A instead of the BFS17 as the VCO transistor and a smaller coupling capacitor value to the ERA3+ MMIC amp thus reducing the actual oscillator loading. I used a 10uH RFC in the crystal oscillator instead of the 15uH, and a 330uH instead of the 220uH but all other components were basically as per the standard V3 schematic.

For interest sake, I tried putting some other divider values into the PICAXE since I knew the VCO would run up to around 125MHz. At varying positions of the slug in the coil former, the VCO locked properly at test frequencies of 120.0, 108.0, 100.0, 96.0, 80.0 and down to 75.0 MHz but would not lock at 125 MHz. Obviously, with more turns on the inductor, it will lock at anything below 75 MHz. The TBB206G is rated up to 95MHz or so and it appears that this value may have up to a +20% tolerance available in a typical device. From my point of view, that would mean that using a TBB202 divide by 128/129 prescaler then I would certainly be able to reach above 1.2GHz (oscillator circuit permitting).

17 Feb 2010 : Just a quick check on the basic crystal oscillator drift : -26Hz at 95.916666 MHz over 1 hour - most of which occurred during the first 1-3 minutes, and about 8 Hz over the remainder of the hour period. During the second hour, the frequency dropped by another 6 Hz. Please note that this is free-running & not using the external 10MHz GPSDO source. This multiplies up to about 400Hz total at 1296 (x12 LO multiplication) but most is in the first few minutes - and that is a lot better than my current crystal oscillator stability.

Updated the V3 schematic and PCB layout and coil details (etc..) for inclusion on this web page. See the table above for photos & layouts.

22 Feb 2010 : The project got diverted for a few days while I built up a PICAXE-based audio decoder/frequency logger ([/~vk4adc/web/index.php/picaxe-projects/38-audio-detection/90-picaxe-based-audio-frequency-decoder.html](http://~vk4adc/web/index.php/picaxe-projects/38-audio-detection/90-picaxe-based-audio-frequency-decoder.html)) interface & PC software to help me track the actual drift in the LO's built up using the synthesiser on this page plus the inherent drift in the radios I use. I am not completely finished with that project simply because I want to verify the results on a random time basis against the likes of the frequency displayed on a "Spectran display". Moving on, for now....

I received my batch of 10 x 10.000MHz crystals last week so I set about checking them by unsoldering the original crystal on the V3a1 PCB after setting the PLL output to 96.000 MHz and just tacking the new crystals in place one-by-one. The basic crystal frequencies were random (best descriptive word I can come up with for now) with 4 coming up high (eg around 96.003), 5 low (down below 95.990 with the worst one), and only 1 crystal nearly the same frequency as the original. I didn't tabulate the actual values simply because I was somewhat astonished at the frequency variation of low to high. The moral of the story is that all of these crystals are not equal : some will be on/near frequency but most won't. I must also point out that these are simple HC-49 computer-use styles that are cheap (and therefore they must think 'near enough is good enough') but all were made by the same manufacturer and are probably even the same batch code. By altering the capacitances in the oscillator, the 'high' ones will pull down to frequency but the 'low' ones are basically throw-aways for this type of application. High-quality close-tolerance crystals are the best option.

Ok, I now know the crystal frequencies are random. I will test them for actual frequency stability & that WILL be interesting.....

Latest: (scroll up for all previous info..)

14 Mar 2010 : It's been a while since I made any progress on the PLL synth simply because most of my time has been directed towards the creation and testing of a new trapped inverted-V for 10 to 80 metres ([/~vk4adc/web/index.php/hf-projects/45-hf-antennas/104-trap-inv-v-for-hf.html](http://~vk4adc/web/index.php/hf-projects/45-hf-antennas/104-trap-inv-v-for-hf.html)) for the upcoming John Moyle Field Day. Today I managed to get back to the workbench for more experimentation & testing - only this time it was with the TBB202G prescaler soldered onto the PCB and the output from the VCO linked across to it instead of the TBB206G. The coil in the VCO circuit was reduced from a multi-turn inductance to a small hairpin loop and the VCO frequency jumped to around 400 MHz. My aim was to try to get the VCO to run at 418.000 to allow its use in the 70cm transverter ([/~vk4adc/web/index.php/vhfuhf-projects/24-v-u-](http://~vk4adc/web/index.php/vhfuhf-projects/24-v-u-)

transverters/56-70cm-tvtr-project.html) in lieu of the current crystal PCB and then make it run at 575/576 MHz for the 23cm transverter (/~vk4adc/web/index.php/vhfuhf-projects/24-v-u-transverters/55-23cm-tvtr.html). I succeeded in making it lock at 418.000 but have yet to make it move up far enough to do the 575 MHz. As it is the VCO's hairpin inductor is only about 15mm of tinned wire (a cut-off component pigtail actually) and I have doubts as to how much shorter I can make it and for the oscillator to still work. I may yet have to change some of the fixed capacitors to make it go high enough in frequency.

For those who haven't been involved with synthesisers too much, the dual modulus divider arrangement may be hard to figure out initially. In essence the PLL divides by a preset count (the A value) then changes the state of the MOD pin and counts for the balance of the count (the N value). The difference is that the N value is via the prescaler divider, effectively enabled by that MOD pin level change, so that the count is actually $N * P$ (where P is the prescaler factor eg 128).

The easiest way to understand it is to actually put some numbers in and see how the divider values work out....

The TBB202G prescaler divides by either 128 or 129, under the control of the MOD output pin of the TBB206G. The datasheet description is :

Modulus Control Output for External Dual-Modulus Prescaler. The modulus output is low at the beginning of the cycle. When the A divider has reached its set value, MOD goes high. When the N divider has reached its set value, MOD goes low again and the cycle starts again. If the prescaler has the divider factors P or P + 1 (P for MOD = high, P + 1 for MOD = low), the total division factor is $N \times P + A$. The value of the A divider must be smaller than that of the N divider.

In single-modulus operation and standby in dual-modulus operation the output is high-impedance for open-drain and tristate for push-pull.

What this means is that you have to connect the MOD output pin (pin 7) to the MOD input pin (pin 6) for the prescaler to allow dual modulus operation - i.e. for the A divider to actually do anything.

Without this connection in place, the only division that occurs is the N divider value, and it doesn't matter if $A = 0$ or $A = 127$ or any value in between - the VCO frequency won't change..

I can vouch for that effect because initially I omitted to solder pin 6 on the TBB202G and I couldn't figure out why the VCO frequency was close - but wrong ! Changing the N value by 1 moved the frequency but varying the A value did nothing. It was a while before I realised my error but that pin is now duly soldered and the thing now locks to the correct frequency.

The process to work out the divider values when using a prescaler is along the lines of this :

Desired LO frequency = 418.000MHz

Reference frequency = 83.3333 KHz = 'R' divisor value of 120 with a 10.000 MHz reference input.

Total N divider value = $418000 / 83.33333$ (both in KHz) = 5016

We know that the TBB202 prescaler primarily divides at 128 for the 'N' count - we'll call that 'P' and = 128

The actual 'N' = $5016 / 'P'$ = 39.1875 but since 'N' has to be an integer we take only the whole number part = 39

Next we calculate the 'A' value = $5016 - ('N' * 'P')$ = $5016 - (39 * 128)$ = $5016 - 4992$ = 24.

That gives us values to put into the PICAXE program for loading into the TBB206G : 'R' = 120, Ntot=5016, 'N' = 39 and 'A' = 24.

And guess what - it works !

Just one point in the above extract of the MOD function that cannot be ignored : "The value of the A divider must be smaller than the N divider". In the case above it is correct (24 versus 39), but other cases may not be - and the easiest way to fix that is to change the reference frequency lower so the total N division value is higher.

In the above example, let's try to use it at 404.000MHz instead

Ref Freq = 83.3333 so R = 120

Ntot = 404000 / 83.33333 = 4848

P = 128

N = 4848 / P = 4848 / 128 = 37.875 = 37

A = 4848 - (N * P) = 4848 - (37 * 128) = 4848 - 4736 = 112

So R = 120, Ntot = 4848, N = 37, A = 112 Oops - A is greater than N !!!

Will it work ?????

Tomorrow !!

15 Mar 2010 : The question is still - will it work ????? The answer - NO. I put the R, N & A divider values into the PICAXE and watched the frequency counter - it immediately jumped to 397.750 instead of 404.000. I thought it would be worthwhile just stepping the values around the N value of 37. A=38 (high) - F= 397.750 , A=37 (equal) - F = 397.750 , A=36 (low) - F = 397.6666 , A=35 (lower) - F= 397.5832 - and the data sheet is correct. The A value must be less than the N value.

The frequency 404.000 is a nice even number so I thought that first I would try a 100KHz reference frequency : R = 100, Ntot = 4040, N = 31, A = 72 - so still no good.

Let's check with a 50KHz reference : R = 200, Ntot = 8080, N = 63, A =16 - that seems ok.

I put those R, N & A divider values into the PICAXE and watched the frequency counter - it immediately jumped from 397.750 to 404.000.

Of course the moral of this story is that you must read the data sheet - if it says that A must be less than N - then it has to be. If it didn't matter at all then the statement would not be included there. Just because the numbers calculate to give the correct Ntot, that does not mean it will work.

I have updated my software "Siemens TBB206G PLL Divider Calculator - VK4ADC - Mar 2010" this morning to make sure that the 'A is less than N' test is performed - and that makes it very easy to find and test alternate reference frequencies and obtain the divider values. Hey, we have computers to do our number crunching so why use a calculator for tedious repetitive stuff ??

I tried a slightly shorter VCO hairpin loop but without changing any other components could not get the VCO up above 500MHz. That rules out 575.5 and 576 operation at this stage. Ok, what about tripling rather than doubling the LO frequency for that one. A quick calculation gives 384.000 MHz for 1152 LO injection, 383.6666 for 1151 MHz. Crunching the numbers in my software gave me the correct divisors and A < N ratio provided I reduced the reference frequency to 22.22222 KHz (ie R = 450). In the numbers went into the PICAXE - immediate locking at 384.000 MHz. Altered the A & N values to give the alternate 1151 frequency - up came 383.666 MHz on the counter.

16 Mar 2010 : One of the questions that often strikes constructors of "kits" is how many of these have been built before - and do they work ? While what I'm working on could not be described as a kit, duplication is sometimes a hit & miss process. Will this one work as well as the last ??

I did a few minor PCB layout changes to the PLLsynth board in ExpressPCB, mainly the incorporation of a coupling capacitor between the TBB202G prescaler and the TBB206G synth chip. The data sheet didn't indicate that one was necessary but my testing of the board a couple of days ago showed that the divider sensitivity was far better with AC coupling. I also added a couple of pads to allow the fitting of a 2 pin header pin/cap assembly in the varicap voltage line. There is a fine track bridging the pads and it can be left intact if the header is not required. By cutting the track, it allows the VCO frequency range to be checked by simply adding a potentiometer between the "C" output pin (-ve voltage) and the +5V supply rail and taking the moving arm to the varicap side of the two header pins. A little 2 pin header cap can be popped on when done.

That done, in a few hours I produced a new PCB, developed, built it - with just a couple of VCO component value changes - and powered it up. I programmed the PICAXE and up it came on 384.000 MHz (after the 10MHz crystal frequency was trimmed - initially it was about 80 KHz low - but that was just the reference oscillator had not been 'netted'). The VCO component changes were the 47pF collector cap (C25) down to 33pF and the 22pF (C7) down to 10pF - mainly to reduce circuit capacitances and allow the VCO to go higher in frequency. In real terms this VCO needed more inductance to tune so it looks like it might go up high enough - fingers crossed.

The "duplication" process seems ok - and the circuit certainly works very much the same as the previous one.

17 Mar 2010 : Last night I left the new synth PCB running overnight. When I last looked at the counter then it was displaying 384.0000. This morning it was still displaying 384.0000 so while not displaying down to Hz, it was still in the ball-park. The frequency stability is a factor decided only by the 10MHz reference oscillator stability anyway and since I used an old crystal, rather than a new one, it will already have partially or fully aged and be stable. The PICAXE was re-programmed for 418.000 MHz with an alternate at 404.000 MHz.

The output level was measured : +11dBm @ 418 MHz

Frequency : 418.00006 MHz - not bad for an non-oven-controlled crystal oscillator

Feeding my external 10MHz in, frequency : 418.00000 MHz

It was time to put this latest one into practice. The first step was to use some tinfoil to provide a shield around it. For Australian constructors, I finally located a source of tinfoil to make up shields for these LO PCBs - after a lot of web searching. The sheets aren't very big but are certainly enough for a couple of small projects.

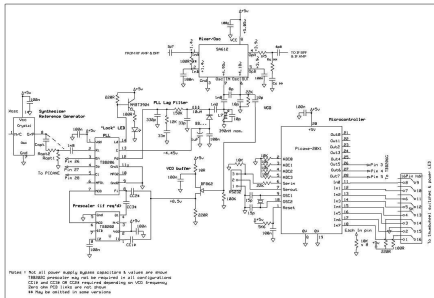
Tinfoil : K&S Engineering , stock #754, 0.008" x 4" x 10" - bought locally from a hobby shop under code " SHM-TIN-0.203x101x254 " - use the detail within the quotes to search in Google Australia, as these same people have a webshop and will ship.

The shields were cut 25mm high and as wide as each face of the PCB (eg about 102mm & 35mm, 2 each) and soldered along the edge of the copper and then each corner was soldered up to the top, sealing the lower part of the box.

The original crystal-based LO PCB was unscrewed and unsoldered from the innards of the 70cm transverter, mounting holes drilled through the new PCB to match the existing holes and the new board screwed down. The +12V supply lead was soldered on, as was the RG174 coax feed to the LO port on the transverter PCB. Power on, LEDs light - the power on the front & the PLL lock - the LO output checked with the counter - yes, 418.000 MHz.

The Icom IC-718 I.F. transceiver was connected - tuned to 14.150 USB, sig gen into transverter common antenna input, the transverter/receiver was easily hearing -127dBm at 432.150. The transmit side was not so straightforward. There was enough 'loose' RF in the diecast box to cause instability and distortion of the transmitted SSB. Additional tinfoil shields and straps were soldered from the LO PCB to the main transverter PCB and a couple of tinfoil straps in various places around the transverter board before it was stabilised. I want to point out that it was not the PLLsynth that was the cause. The transverter was probably prone to the instability before it was changed over from the crystal-LO and I hadn't really noticed. As far as the synthesised LO process was concerned, it was a winner.

August 2010 :



My need for yet-another synthesiser in a project resulted in the implementation as per the schematic above - only this time I wanted to be able to select a number of "channels". This requirement led to the use of a pair of BCD-coded thumbwheel switches to give values of 00-99 - and that led to my using a PICAXE-28X1 because of its extra input pins instead of the 08M used in all versions above. Only the relevant parts of the project are on the schematic.

This one was a little different in that I used a SA612AN mixer/oscillator device where the PLL's VCO was the oscillator section. It worked out to be a nice implementation and during the early testing phase of the SA612, I had the VCO free-running above 200 MHz. My final implementation did not require the TBB202G prescaler even though the PCB layout included pads for one.

The change in the PICAXE code to relate to the different device took just a few minutes due to the way I had originally written the code - with only the output pin definitions for the DA, CL and EN outputs needing to be changed. The value of the BCD thumbwheels was retrieved in a simple subroutine that totalled the "bits" and returned a value from 0 to 99. That value then altered the N-divider in a "case" statement that used a formula to set up the N value before sending it to the TBB206G. The frequency steps were based on a 25KHz raster so the highest available PD frequency that suited was... 25KHz ... so that high-ish frequency made the PLL lag filter pretty simple, noting that this version uses a 10uH RFC** to feed the lag filter output to the varicap (in lieu of a resistor). The BF862 FET follower between the VCO and the PLL chip is simply to provide buffering/isolation and keep the PLL input sampling from sending back "noises" into the VCO.

** The RF choke inductance should be around 25-50 times the inductance value of the VCO inductor eg 200nH VCO coil = 5 to 10uH RFC.

The project went well, the synthesiser very well and it was nice to see that it was so easy to vary the PICAXE type.

One thing that I must mention here - I am not publishing my PICAXE code or the actual PCB layout files on this web site.

If you are going to build a PLL project based on mine then you will need to contact me and ask for info directly.

I will then expect info as to the outcome of your project ... in the form of photos, reports, problems encountered... so that I can improve this project/web page..

{ "Tyre kickers" need not apply }

This page is meant to be a kick-start for others - to give them ideas and methods - rather than be a complete article on just building up a PLL step by step

eg. which says solder wire onto terminal A and connect to terminal B, install resistor 2 at position 32...

I have no intention of producing PCBs or kits for others so don't ask.

What I will do (eventually) is to supply genuinely interested & enthusiastic builders with the TBB206 Delphi application, the final PCB layout & schematic files and the PICAXE code file for their personal use.