# LMX2470 Fractional-N Synthesizer Development PCB

***Andy Talbot G4JNT February 2012***

## Overview

With a suitable separate Voltage Controlled Oscillator this PCB allows a complete Fractional-N Synthesizer operating within the frequency range 500 to 2600MHz to be constructed. The PCB has provision for an LMX2470 Synthesizer chip and allows components to be added for both passive and active loop filters; the latter making use of an LM6211 high speed rail-to-rail op amp. The board also includes a 2.5V regulator for the synthesizer chip and a 12F629 PIC for ASCII text based communications using a serial or RS232 interface for setting the chip's internal registers and saving in non-volatile memory. A small TCXO can be fitted to provide a reference frequency source for less-critical applications

Please note, this is not in any way a kit for a complete microwave synthesizer. It is just the PCB and, optionally, a PIC interface to allow you to build your own designs using this versatile chip. The VCO has intentionally not been placed on the board as the variability between different types would make it too restrictive if provision for one type only were to have been added. Any of the Mini-Circuits or Z-Comm range will work, as will the cavity oscillator from the old "Bricks" which typically run in the 1300 – 1800MHz region.

Before use, obtain a copy of the LMX2470 Data Sheet from the National Semiconductor website http://www.national.com/pf/LM/LMX2470.html#Overview where all supporting documentation and design software can be found. The Codeloader Software (see later), also to be found on the Nat Semi website, is an invaluable utility for calculating the register values to be set.

You will also need to design your loop filter. The Nat Semi. documentation has comprehensive instructions on how to do this. The spreadsheet included here, ***LMXFILTER.XLS***, implements the design equations given in several LMX series data sheets for 2nd and 3rd order filters and provides a starting point.

## Circuit Operation

The circuit diagram is shown in Figure 1. A higher resolution version of the circuit diagram is stored as ***lmx2470.gif.*** A separate VCO supplies a sample of RF at a level of between -15 to 0dBm to the chip's RF input pin. This is compared with a reference signal which is ideally supplied from an external master oscillator or low phase noise clock generator. Optionally, pads are provided on the board for a small TCXO to be installed. Reference input level is not critical, and the data sheets suggests -40dBm as a lower limit; the maximum should not exceed 2.5V peak. The 150Ω termination resistor, R6 has been provided as a compromise for long coax feeds and not loading TCXOs. Use any value you feel comfortable with, or leave it out altogether

The charge pump output is filtered by either a passive or active loop filter (components should be fitted for one or the other, not both sets) and the output, on the $V_{cont}$ pin, goes to the VCO tuning input. On board components allow for up to a third order filter. The final components for making up a fourth order filter can reside on the VCO

Note that although the LMX2470 does contain a second Integer-N IF synthesizer, no provision is offered for using this on the PCB.

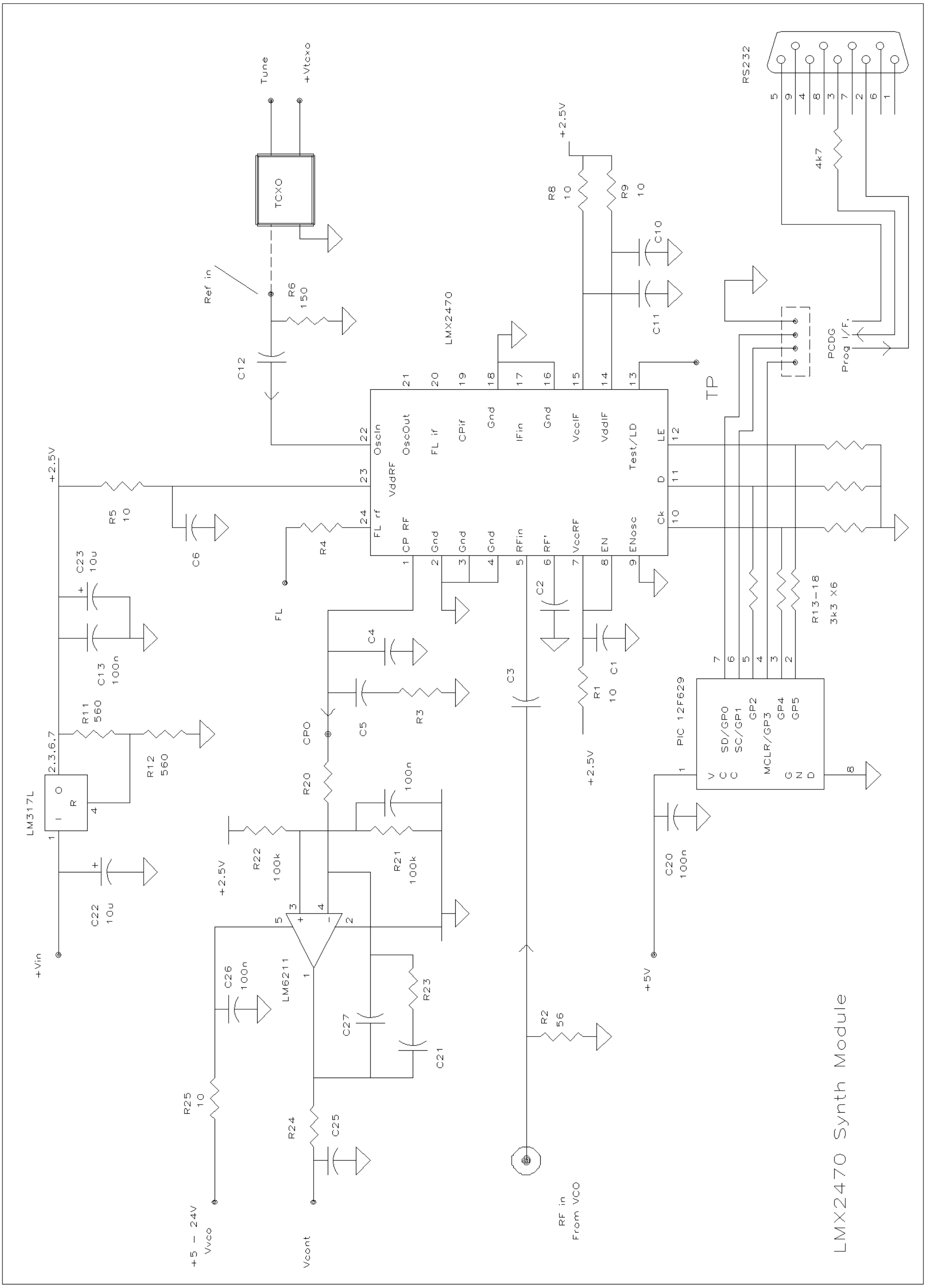


**Figure 1 Circuit Diagram File - *lmx2470.gif***

**Power Supplies**

The Synthesizer chip power supply needs 2.5 $V_{dd}$ supplied by the on-board LM317L regulator; current consumption is less than 10mA. Input voltage, Vin, to this can range from 4.5 to 20V.

The PIC controller will function with a power supply in the range 2.5 to 5V and in fact the control system installed on it would work if the PIC were powered from the same 2.5V rail as that used for the LMX2470. However, in-circuit programming of the PIC requires a 5V supply, and in addition the reliability of the pseudo-RS232 output is enhanced by running from a 5V rail. Vin to the LM317L regulator can be conveniently made equal to this 5V, which is then supplied as one from a second external regulator.

The difference in power supply voltage, and hence I/O levels between the PIC and synthesizer chip is compensated by the potential dividers on the three interface lines made up from resistors R13 to R18. Figure 1 shows values suitable for 5V $V_{dd}$ on the PIC, they must to be changed appropriately if any other supply voltage is used.

If the on-board TCXO is fitted, a separate 3.3V supply for this will also have to be provided.

**PCB construction**

The PCB layout is shown in Figure 2, component references should tie-in with those in Figure 1, but note that some references are missing *(don't ask !)* Component values not shown will depend on the design frequency and the PLL filter. Coupling and decoupling capacitors around the chip should be around 100pF for operation above 1.5GHz, increasing to around 220pF at the lower end of the range. Exact values are not critical.

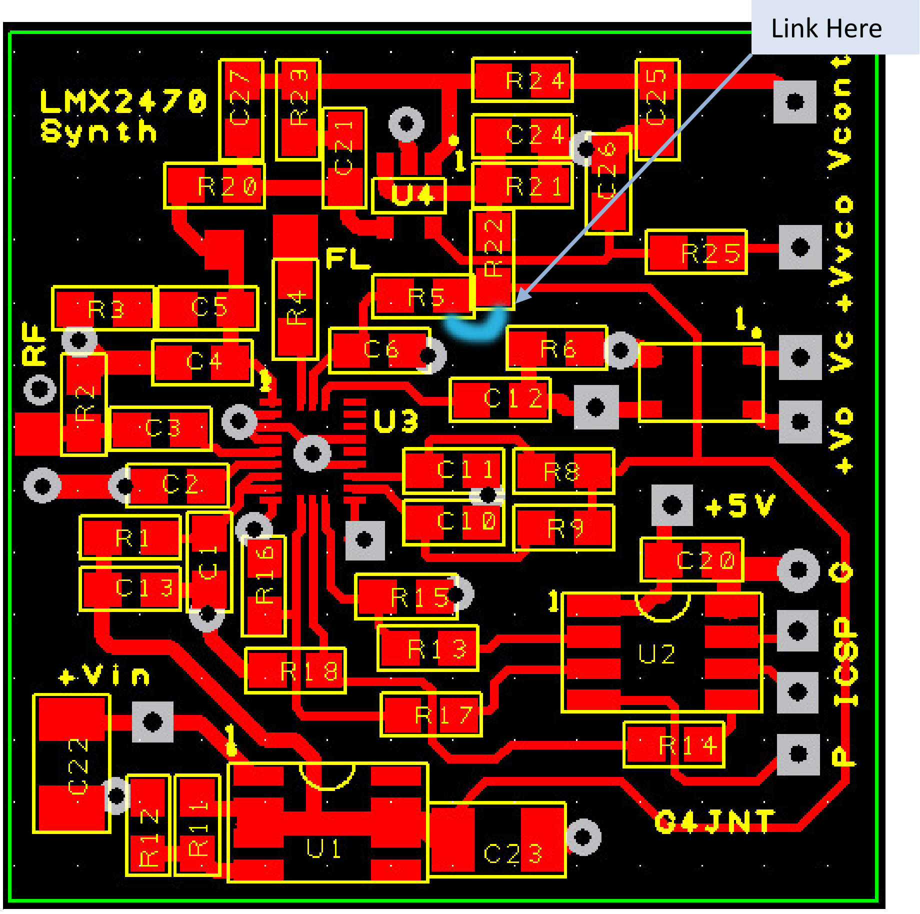


**Figure 2  PCB layout**

The only component that may give any difficulty in installation is the LM2470 itself. This should be fitted first, before any other components are added. The chip has 0.5mm pad spacing and is of the type where the connections are on the edge of the ceramic package, with no gull-wing tabs. The package is rectangular, with three pads

**Note : Version 1 of the PCB has a missing connection between the top end of R5 and the 2.5V rail at R22. A link must be made at the point shown labelled in blue in Figure 2**

each top and bottom, and nine on each side.  A marker shows Pin 1 at the top left.

This is how I solder the LMX2470 device.  Use a decent high magnification viewer; at least a headband type or for even greater magnification an old 50mm SLR camera lens is ideal.  The pads on the PCB for the chip should first be tinned, aiming to keep an equal thickness of solder on each pad as shown in Photo 1[1]. Then the pads should be liberally coated with flux. Double, triple and quadruple check k the IC alignment, place it central over the footprint so an equal amount of pad is showing on each side. Make sure there is flux still there on each pad. Press the IC down firmly, checking again for alignment. Peer though a magnifier at the IC pads sitting over the correct pads on the PCB while still pressing down on the IC.  Look away for a few seconds to relax your eyes, return and check alignment again.

Once absolutely certain each IC connection is above its pad and unmoved and still keeping it pressed down, using a medium-sized (typically 1mm) chisel-end soldering-iron bit place it in turn on each pad close to the IC, while watching the solder melt under the IC connections.  The flux will cause a sort-of wicking action, drawing molten solder up from the pre-tinning to fill the gap to fix at least some of IC connections.  Gently prod the chip to make sure it is now fixed by at least some pads and won't move of its own accord.  Now apply more flux to the IC / PCB and load the soldering iron tip with solder.  Wipe the iron against each side of the chip in turn, gently pushing solder into the gaps.  Providing there is enough flux present, it should nicely flow into the minute gaps between each IC connection and its PCB pad.  The flux – somehow, if there's enough of it – always seems to prevent unwanted bridges forming. These are more likely if there is not enough flux, or the solder on the bit has been allowed to get dirty and oxidised. Photo 2 shows a mounted chip. Observant readers will note the pad on pin 17 appears to be an un-pad. It is no more, it has gone to meet its maker. Guess who got a bit enthusiastic pushing solder into the joints.  Being otherwise unattached this pad when heated becomes rather delicate.  Too vigorous pressing with the soldering iron moved it, so to prevent shorts I removed the pad completely.  (Its only the unwanted IF input)

Once the chip appears to be fitted, breath a sigh of relief and get out the high gain magnifier. Look carefully at each joint making sure there is a continuous solder flow from PCB pad to IC connection. And also that there are no bridges.  Bridges can be removed using solder wick / braid, again making use of extra flux at all times to aid solder flow and cleaning.

If you have the right sort of a diode tester facility on a DVM, the connections to each pad of the IC can be verified.  The tester must have a voltage on the probes of **no more than 2 – 3V open circuit**, and no more than **a couple of 100µA test current.** If it meets these criteria connect the negative lead to ground and probe each IC connection in turn.  All will show a voltage of between 0.7 to 1.8V. Any that Indicate the full Open circuit voltage are almost certainly not connected.  I can't vouch this test will work for other chips or different manufacturer's products, but it does for this one.  The tester must be a diode / continuity test with a constant current / voltage reading.  Using the ohms range is not nearly so helpful.

After fitting the main chip, installing the rest of the components should be straightforward!

[1] Photographs appear at the end

| Quantity | Reference | Value |
|---|---|---|
| 1 | R (external on RS232) | 4k7 |
| 4 | C13,C20,C24, C26 | 100n |
| 6 | C1, C2, C3, C6, C10, C11 | 100 – 220p |
| 2 | C22,C23 | 10µ |
| 1 | C12 | 10n |
| 2 | R11,R12 | 560 |
| 5 | R25,R1,R5,R8,R9 | 10 |
| 2 | R21,R22 | 100k |
| 1 | R6 | 150 |
| 1 | R2 | 56 |
| 6 | R13,R14,R15,R16,R17,R18 | 3k3 |
| 2 | R4, R20 | 0 |
| 1 | U1 | LM317L |
| 1 | U2 | 12F629 |
| 1 | U3 | LMX2470 |
| 1 | U4 | LM6211 |
| C4, C5, R3 | Use with Passive Loop filter only (no LM6211) | |
| C21, C27, R23 | Use with Active Loop filter only, first two poles | |
| R24, C25 | Loop Filter Third Pole | |

*(nb. Several component references are missing )*

The PCB has been sized to fit inside a standard 37mm tinplate box, with the RF input provided by a socket soldered to the side in the conventional way for hard-cabled RF solutions. Alternatively, an uncommitted through-hole has been provided adjacent to the RF input tab so RF can be fed through the PCB from the groundplane side. This gives the opportunity to install a VCO assembly on the reverse of the main board for a compact single box assembly.

All power supply and signal pins are provided with through board connections, so these can be made on either side.

**Programming and Getting it Going**

The chip's internal registers have to be programmed to the correct values for your desired frequency and fractional grid, and the loop gain, and etc . The data sheet and an understanding of Fractional-N synthesizers is invaluable here, but the easiest way to start off is to use National's ***CodeLoader*** software to work out the values for you [2] ***CodeLoader*** shows a graphical representation of the RF PLL where you can interactively set frequencies and division ratios, as well as all the other PLL settings like gain, and phase detector polarity. What is does not show on its GUI (at least version 4.5 doesn't) is the X2 reference multiplier so if you choose to have this running, the clock frequency has to be entered into the GUI window already pre-multiplied. To ensure this is actually enabled on the chip, go into the Register / Bits screen and tick the X2 box. Also select whatever you want from the TP test point on pin 13. Setting this to deliver the divided down N or R dividers is instructive - it actually supplies a square wave at half the divided down or reference value and can make a quick confidence check of correct register programming before a complete closed loop system is established.

Look at the waveform from here on a scope and experiment with programming the R6 register to toggle between divided down R and N values.

Otherwise, the default registers programmed into the PIC will work as a starting point, but read the data sheet before modifying registers at will. For most operations, only R0, R1 and R7 need to be set to give a

[2] The ***CodeLoader*** software can be used to programme the synthesizer chip directly using three wires from a parallel printer or LPT port, bypassing the 12F629 PIC completely. This is adequate for purely experimental work, but there is clearly no opportunity for a finished boot-at-turn-on facility. Furthermore, great care has to be taken matching voltage levels. However, it does work and provides an elegant solution for some general experimentation with the chip. Full interface details are provided in the ***CodeLoader*** help section

particular frequency, grid and reference. The other registers can usually be left alone. But remember to set the phase detector polarity. Assuming the VCO tunes positive, for a passive filter the polarity bit should be set to +Ve; for an active filter the response is inverted and it should be set –Ve. The sign is shown on the ***CodeLoader*** GUI screen and is translated into the register set.

The ***LMX2470_Calc.exe*** programme include in the archive allows quick calculation of R0/1 and R7 for specified values of Fosc, Fout and frequency tuning grid.

## *LMX2470CTL* PIC Operating System

The 12F629 PIC containing the ***LMX240CTL*** code allows the registers in the synthesizer chip to be set directly to the wanted values using a serial RS232, or COM Port interface with simple ASCII text commands from any terminal emulator programme such as ***Hyperterm***. The register values can be programmed into non-volatile EEPROM memory in the PIC. At turn on, these stored values are recalled and sent to the synthesizer for immediate start up.

Connect the RS232 lead to the header pins as shown. The off-board 4.7k resistor serves to limit the current driven into the PIC input pin from the typical +/-9V of most RS232 interfaces. Set the RS232 settings to 9600 baud, 8 bits, 2 stop bits, no parity, 9600-N-8-2. Turn on or reset the LMX2470 module whereupon a display similar to that shown below should appear. This shows a summary of allowed commands followed by the stored register contents in both binary and hex that are immediately loaded into the chip

Registers are updated one at a time by typing R followed by the six digit hexadecimal representation of their new contents, terminated with a [cr].

Eg. *R0E4500[cr]* *Note that the 0x shown on the response must not be typed*.

Typical opening screen sent via the 9600 baud RS232 interface when the LMX2470CTL Operating system starts.

```
G4JNT
LMX2470 CTRL
Rxxxxxx [CR]
Nxxx [CR]
Qxxx [CR]
[W]rite reg EE
[I]nit from EE          (The word 'INIT' has to be entered in full)

0000 1110 1001 0000 1011 0000  0x0E90B0
0100 0001 1000 0011 1100 0001  0x4183C1
1100 0000 0000 0111 0100 0011  0xC00743
0111 1000 0000 0000 0001 0101  0x780015
1011 1100 0010 1011 1100 0111  0xBC2BC7
0001 0000 0001 1111 0100 1001  0x101F49
0000 0010 1100 0010 1111 1011  0x02C2FB
0111 0001 0100 0001 0100 1101  0x71414D
0000 1100 0000 1000 0110 1111  0x0C086F
```

The final LS digit of the six defines the register address – see the data sheet for full details of register addressing. The new value is immediately transmitted to the synthesizer chip, which should straightaway respond accordingly.

To save this new register value to non-volatile memory in the PIC, type *W[cr]* after the register contents have been accepted. A response will show that individual register has been written to EEPROM. To store multiple registers , enter these one at a time, issuing a W command for each one. This is not as tedious as it sounds as subsequent updates will quite likely only require that one or two registers be altered at a time.

If you get in a mess, type *INIT[cr]* which has the same effect as switching off and back on; reloading all the registers with the prestored values. When updating the EEPROM with several new values, it is worth issuing INIT commands periodically to check these have all been stored correctly.

Two additional commands have been included, mainly with a view to use with generating multi-frequency datamodes. These are *Qxxx[cr]* and *Nxxx[cr]* for a quick update of the lowest significant bits only of the Fractional F and Integer N registers stored in Register 0. These two commands, and *Qxxx* in particular, are intended for directly driving the synthesizer from a code generator for generating, for example JT4 or JT65 signalling waveforms [3]. They write the associated portion of the R0 register only, the 'other' part is recalled from EEPROM each time. So make sure this has been correctly set to start with. The Q and N commands generate a shorter acknowledgement, allowing the complete frequency change to be performed in around 15ms making the process jut about fast enough for generating RTTY - type baud rates. A constant processing delay in the PIC ensures no time skew between symbols.

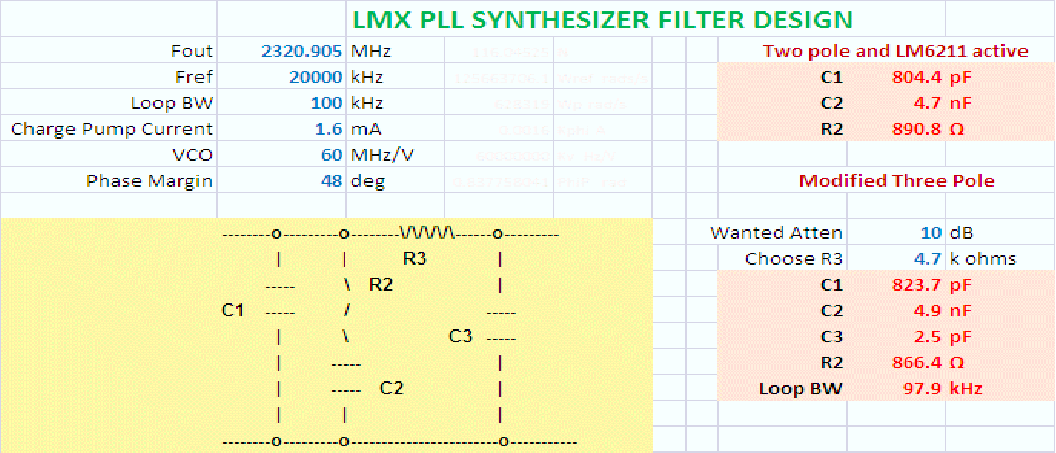
**PLL Filter Design**

The design of PLL filters for Fractional-N synthesizers is no different in concept from that for Integer-N designs, apart from the fact that the reference frequency is usually (a lot) higher with corresponding reduction in the integer part of N, and usually a higher loop bandwidth. The spreadsheet ***LMXFILTER.XLS*** (download from http://www.g4jnt.com/lmxfilter.xls ) as shown in Figure 3 lets the basic values for the three main components forming the first two poles and the optional RC pair for a third pole be calculated. Enter values the figures highlighted in blue.

R4 provides for Fastlock operation if desired. See the data sheet for how this is implemented and place a wire link to connect into the PLL filter at the right point if Fastlock is wanted. R20 should normally be made 0 ohms; it is included just as a convenient place to isolate a passive filter from otherwise unused tracks.

The Component values shown below for the PLL filter work for VCOs in the 1 – 2GHz region with a few tens of MHz /V tuning rate. Adjust values to optimise spurii / loop bandwidths. As an alternative starting point, with a MiniCircuits JTOS-3000W VCO having the rather high tuning sensitivity of 123MHz/V, using a loop bandwidth of 80kHz and charge pump current of 800uA values of C1 = 1nF, C2 = 6n8, R2 = 620, R3 = 470 and C3 = 100pF works (because that's what the test breadboard has). Play with the filter values, charge pump current until you get what you want.

Fractional-N synthesizers provide a lot of options for optimisation!



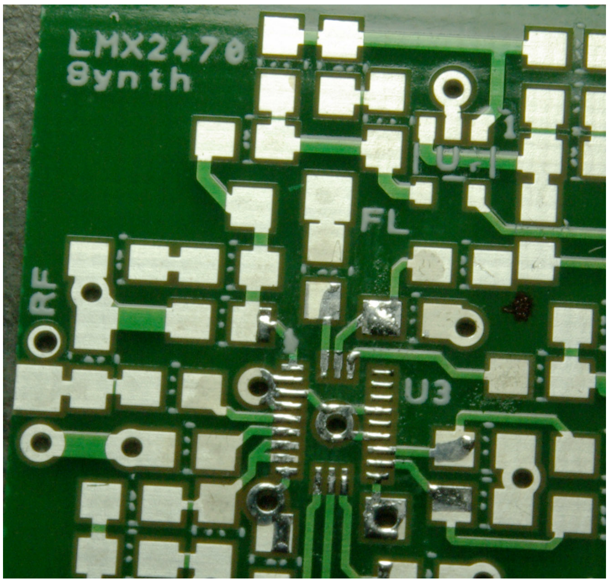
LMX PLL SYNTHESIZER FILTER DESIGN

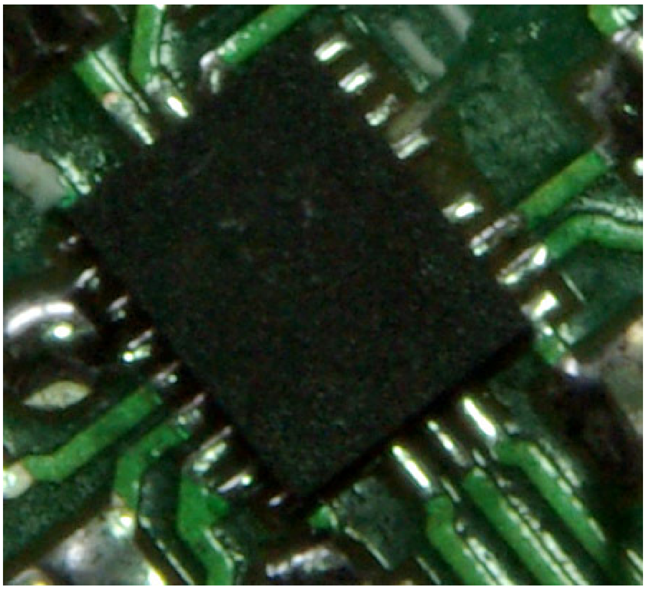
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Fout | 2320.905 | MHz | 116.04525 | N | Two pole and LM6211 active | | |
| Fref | 20000 | kHz | 125663706.1 | Wref rads/s | C1 | 804.4 | pF |
| Loop BW | 100 | kHz | 628319 | Wp rad/s | C2 | 4.7 | nF |
| Charge Pump Current | 1.6 | mA | 0.0016 | Kphi A | R2 | 890.8 | Ω |
| VCO | 60 | MHz/V | 60000000 | Kv Hz/V | | | |
| Phase Margin | 48 | deg | 0.837758041 | PhiP rad | Modified Three Pole | | |
| | | | | | | | |
| | | | | | Wanted Atten | 10 | dB |
| | | | | | Choose R3 | 4.7 | k ohms |
| | | | | | C1 | 823.7 | pF |
| | | | | | C2 | 4.9 | nF |
| | | | | | C3 | 2.5 | pF |
| | | | | | R2 | 866.4 | Ω |
| | | | | | Loop BW | 97.9 | kHz |

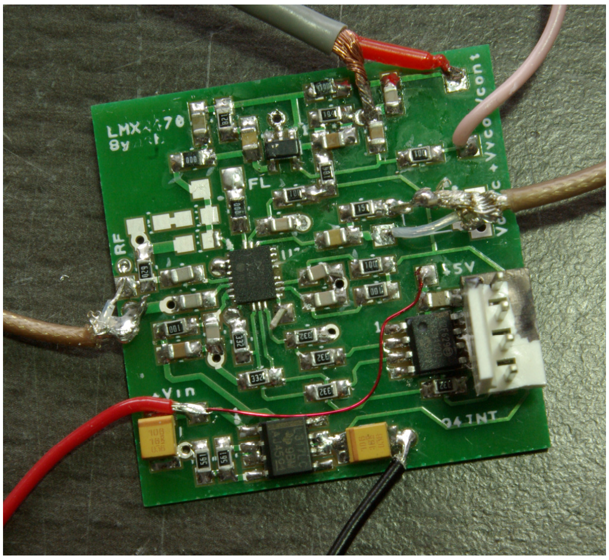
```
--------o---------o--------WWW------o---------
        |         |        R3       |
      -----       \  R2             |
  C1  -----       /               -----
        |         \            C3 -----
        |       -----               |
        |       -----  C2           |
        |         |                 |
--------o---------o-----------------------o-----------
```

**Figure 3 Screen capture from the PLL Filter design spreadsheet File - *LMXFILTER.XLS***

[3] PIC Generator code for generating WSJT modes on this synthesizer module by sending correctly timed Qxxx commands will be available at a future date.


LMX2470
Synth
RF
FL
U3




LMX2470
RF
FL
+Vin
G4JNT