A *Simple* GPS Stabilized 10 MHz Oscillator

Based on the original construction by James Miller, G3RUH[1]
Mike Seguin, N1JEZ
N1JEZ at verizon dot net

As N1JEZ/R, I operate a Rover station in the ARRL VHF/UHF/SHF contests. My vehicle, a Subaru Forrester, is equipped to operate on all bands from 6 Meters through 47 GHz. This equipment gets bounced all over as I travel from grid to grid. I wanted some way to check basic receiver performance and also frequency accuracy of my transverters when I arrived at a new location. There are many times I am not within range of any beacons, so some other method was needed. The 10 MHz reference oscillator described here along with a Qualcomm synthesizer board are used for an accurate marker system designed to fulfill this goal.

THE PLAN

- Use a Qualcomm 750 – 1000 MHz synthesizer for markers.
- Modify the synthesizer for 1152 MHz.
- Generate harmonics 2304 – 24192 MHz.
- The Qualcomm uses a 10 MHz reference.
- Use a GPS stabilized 10 MHz oscillator for this reference.

THE QUALCOMM BOARD

The Qualcomm synthesizers are easy to acquire from sources such as Chuck Houghton, WB6IGP. The modifications needed to run these at 1152 MHz are well documented at the SBMS website[2] so I won't go into them here.

THE JUPITER GPS ENGINE

The 10 MHz reference oscillator is based on the original design and construction done by James Miller, G3RUH[1]. It utilizes a GPS engine called a Jupiter. To quote James,

> *"A well kept secret of the Rockwell/Conexant/Navman Jupiter GPS engines is that they have a 10 kHz output, synchronised to GPS time. So, with a few additional components you can phase lock a 10 MHz VCXO to the 10 kHz, and you then have a simple lo-cost frequency standard with remarkable performance."*

The Jupiter GPS engine is a single board 12 parallel-channel receiver intended as a component for an OEM product. Some of the features of interest include:

- Small size – 2.8" x 1.8" x 0.44"
- 10 kHz output synchronized to GPS time.
- Support for NMEA-0183 data protocol.
- Compatible with either passive or active antenna.
- 1 PPS output.
- RTCM SC-104 data capability. (DGPS)

These engines have been showing up as surplus lately. There are 6 versions of the engine that I am aware of: Standard, Dead Reckoning (DR) and Timing (T). These 3 versions come with either 5 volt or 3.3 volt supply requirements. I have successfully used both the 5 volt and 3.3 volt

Standard engines. While I believe that all versions will work, intuitively I don't think there is any added advantage to using either the Timing or DR version in this application. However I can't support that with measurements.

Documentation for the engines is available on the web. One of the more comprehensive sites is: http://www.gpskit.nl/downloads-en.htm. While the model numbers may not be an exact match for a surplus engine you might obtain, the information at this site should still be applicable. The Jupiter series is still a current production item and can be seen at http://www.navman.com.

The Jupiter engine has a 2 x 10 pin field connector. The pinout is described in the documentation. Depending on the version of documentation you acquire, it may not describe a 3.3 volt engine. However, it's easy to identify one. The standard 5 volt engine has pin 2 on the connector. This is the normal 5 volt input. If that pin is missing, then the engine is a 3.3 volt board and power is supplied to pin 4. On both versions, an active antenna can be used by supplying power to pin 1. This pin can accept up to 12 volts at 100ma so it's possible to power an active antenna that might not be of the garden-variety 5 volt type. Be aware that there is no current limiting on the board. If you happen to attach a passive antenna that presents a DC short, the board can be damaged. To safely use a passive antenna, remove voltage from pin 1.

The default power up behavior of the serial port on the engine is defined using pin 7 and 8. For NMEA, pin 7 is always pulled low. Pin 8 is used to select default initialization with the choices being ROM (slower) or SRAM/EEPROM. The advantages of using the SRAM/EEPROM only apply if you install a backup battery or SuperCap. Using either of these backup solutions, you can maintain the Almanac, Ephemeris and RTC (real time clock) in memory during power down and the TTFF (time to first fix) is reduced significantly. All of these options are explained in the documentation.

One added bonus of this system is that the serial output data is NMEA-0183 compliant. Therefore it can be used with any software program that is able to utilize generic NMEA such as Delorme Streets or Topo[3]. Another application for the NMEA output that I've considered is time synchronization for my laptop in the field for use with digital modes that require accurate timing like JT65[4].

THE INTERFACE BOARD

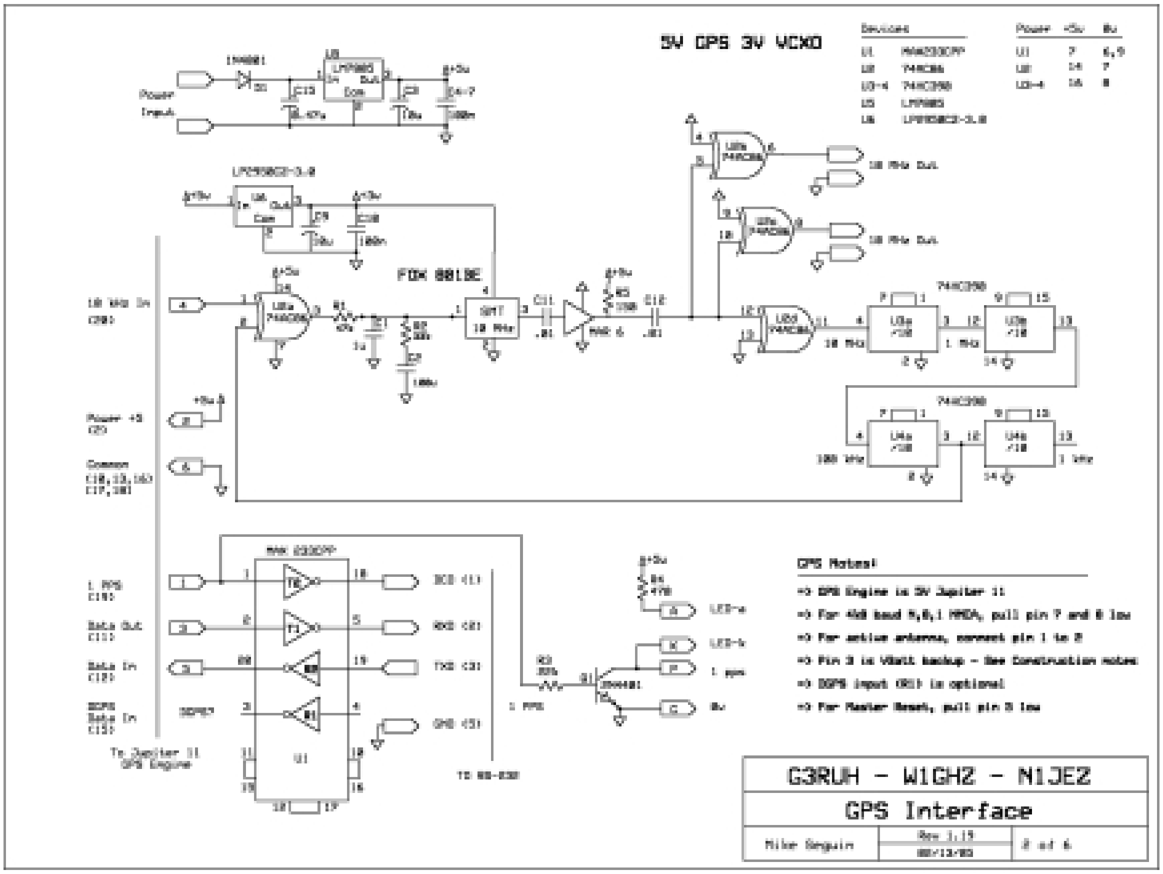
James developed an interface board that contained the VCXO and associated PLL circuitry as well as RS-232 converter for the GPS. I contacted him about acquiring a board, but he no longer had any available. However, he was more than willing to provide the circuit board layout in Gerber format. I gladly accepted his offer!

My next task was to figure out how to reproduce the boards. I was able to view the Gerber files using a program called ViewMate[5]. However, I was unsure how to proceed from there. Shortly thereafter, I contacted Paul, W1GHZ and explained what I was up to. He offered to lay out the interface board using ExpressPCB[6] software and I kindly accepted. Paul had the board done in just a few days. Meanwhile I downloaded the companion software for drawing schematics from ExpressPCB and drew up the basic system.

For those not familiar with ExpressPCB, they offer a web-based circuit board manufacturing service. They supply free downloadable software to design boards and schematics. Once you have your design, you can order boards from them via the web and they will be delivered to you in just a few days.

Now that I had a board layout, I set out to find the various parts. As it turns out, everything was easy to acquire except the 10 MHz VCXO. The interface board called for a 5 volt VCXO DIL-14 part. I was able to find quite a few sources, but none would supply the part in small quantities at a price I was willing to pay… Then Paul spotted a SMD VC-TCXO made by FOX[7]. Mouser Electronics[8] stocked quite a few at a reasonable price. The only problem was the part was a SMD and required 3 volts. After talking with Paul, he redesigned the board to accommodate the SMD within the outline of the DIL-14. There was even enough room to mount a 3-volt low drop out regulator and associated capacitors needed to power the FOX. Now the board could accommodate either type of VCXO. I placed an order for all the parts and also ordered 6 boards from ExpressPCB.

Below is the schematic for the original version I built.



GPS Notes!

=> GPS Engine is 5V Jupiter 11
=> For 4800 baud 8,0,1 NMEA, pull pin 7 and 8 low
=> For active antenna, connect pin 1 to 2
=> Pin 3 is VBatt backup - See Construction notes
=> DGPS input (R1) is optional
=> For Master Reset, pull pin 5 low

As you can see, it utilizes a 5 volt GPS engine and 3 volt FOX801BE TC-VCXO. The circuit is rather simple. A MAX 233 is used for RS-232. While this is a slightly more expensive part, it does have the advantage of not needing any external capacitors as do chips like the MAX 232.

The board features two buffered 10 MHz outputs. It also includes an extra level converter in the MAX 233 chip for use with DGPS if so desired. The 1 PPS output from the Jupiter is brought out on the DB-9 serial connector DCD line and there is an LED that flashes at 1 PPS as a “heartbeat” monitor. The original version I constructed did not have the MAR 6. During testing, I found the FOX had insufficient drive, so the MAR 6 was added.

I now have redesigned the board to accommodate both the MAR 6 option and a 3.3 volt LDO TO-220 regulator that can be used to power a 3.3 volt GPS engine This is in addition to the original 5 volt regulator. If you have a 3.3 volt engine, you'll want both supplies. One is to feed your powered GPS antenna, and the other to power the GPS engine.

There are now 4 different versions of the circuit to accommodate various options:

- 5 volt GPS 5 volt VCXO
- 5 volt GPS 3 volt VCXO
- 3.3 volt GPS 5 volt VCXO
- 3.3 volt GPS 3 volt VCXO

THE LOOP FILTER

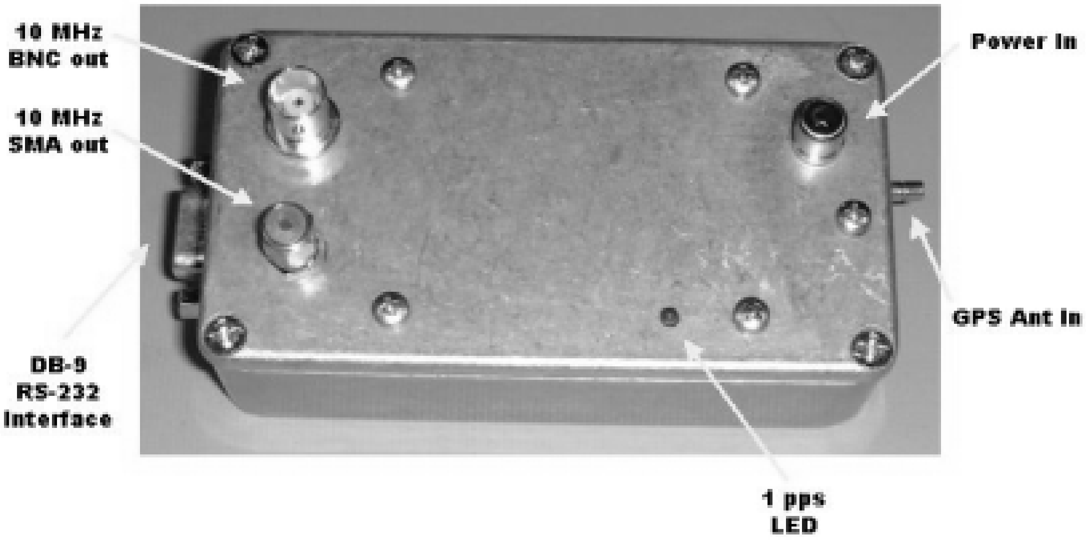
One of the challenges of a PLL is the loop filter. Here again I am indebted to James Miller. He supplied a simple basic program to calculate the necessary components. I did have to do some modification to the code to get it to successfully run under GWBasic, but it was fairly easy. Listed below is the original code from James.

```
REM        CALCULATES PLL LOOP FILTER VALUES
REM                   G3RUH
REM           Original      1985 May 13
REM           Last modified 2004 May 16 [Sun] 0330 utc
REM
REM F   Loop bandwidth
REM Q    ..  Q
REM R1  Series R
REM R2  Shunt  R
REM C   Capacitor in series with R2
REM K   Loop gain = 2*(Total VCO freq swing)
:
@%="f10.2": REM 2 d.p., field 10
R1= 82000 : REM Say
K = 6     : REM 600 Hz/v.  That is: v=0 to +5; N=1/1000 so K=  2*(600*5/1000)
Q = SQR(0.5)
:
PRINT "        F        R1 k      R2 k       C uF"
PRINT "------------------------------------------"
FOR F = 0.01 TO 0.1  STEP 0.01
  W0 = 2*PI*F
  W  = W0/K
  T2 = 1/(Q*W)-1
  T1 = 1/(W^2)-T2
  T1 = T1/K
  T2 = T2/K
  C  = T1/R1
  R2 = T2/C
  PRINT F,R1/1000,R2/1000,C*1E6
NEXT
END
```

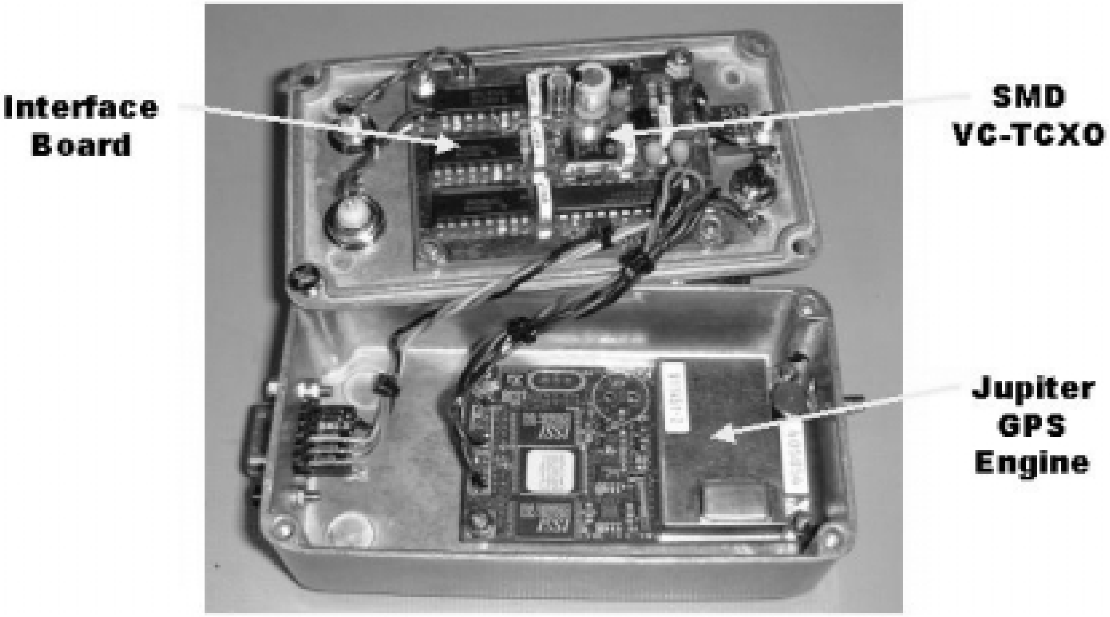
If you consider building a unit, I highly recommend that you measure the actual frequency swing you can achieve with your VCXO. The specifications for the FOX that I used showed ±3 ppm. I measured ±3.5 ppm/v over the 2 volt CVR (control voltage range). Knowing the actual performance of your VCXO helps in calculating a loop filter with reasonable performance.

Construction

Below are pictures of the prototype. The case used is a Hammond 1590B. The whole unit fits comfortably in your hand. The choice of connectors is up to you. I used what was lying around the bench. The GPS antenna input on the original engine I used is a male OSX.



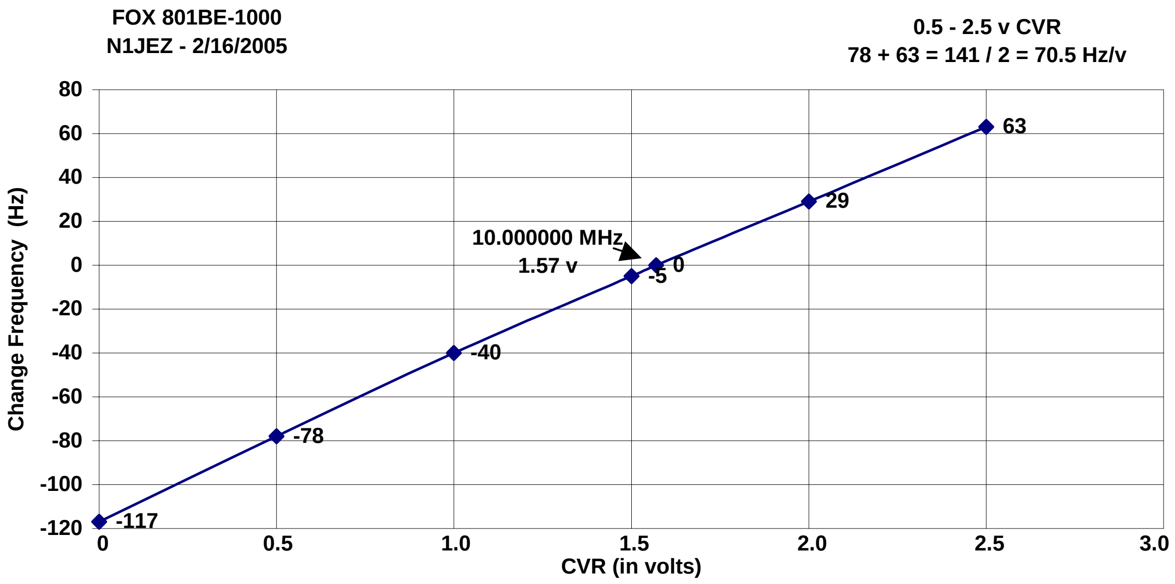
Inside are two boards; the GPS engine and the interface. There is plenty of room to add a small backup battery if you wish.



While it’s possible to use headers on the board and a push-on connector for the 20 pin field connector on the GPS, I opted to wire directly between the boards. The pins on the GPS are easy to remove. I like to eliminate connectors if possible, especially in gear that is going to be exposed to harsh treatment on the road.

The actual construction is quite straightforward. I didn’t use any sockets for the chips. All the parts are through hole except if you need to use the MAR 6 or use a SMD VCXO.

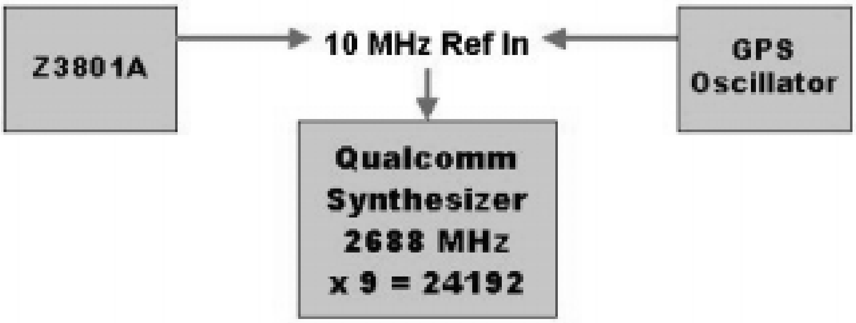
To measure the VCXO tuning range, I assembled the board with the exception of R1, R2, C1, C2 and C8. I then powered the board and hooked a variable power supply to the VC line of the VCXO. While watching my frequency counter, I varied the tuning voltage between 0.5 volts and 2.5 volts (the specified tuning range for the FOX) and recorded the change. This number was used in the loop filter calculation. Below is a graph of the measurements I made on my FOX 801BE.

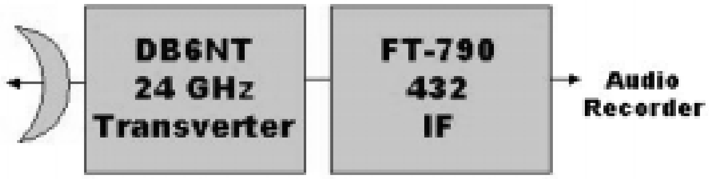


PERFORMANCE

The first time I powered up the oscillator it locked in less than 3 minutes from a cold start. Most of this was GPS acquisition and lock time. I don’t have a backup battery in my system. When I initially powered the unit, the VCXO went to the upper end of its tuning range. As soon as I obtained a valid 3D GPS fix, the counter proceeded to drop. It overshot the mark by a bit and then slowly returned to 10.000000 MHz and didn’t budge.

My frequency counter is locked to a HP Z3801A frequency standard that I use as a reference, so I’m reasonably sure of my accuracy. Even so, my counter can only resolve 1 Hz. It really couldn’t tell me how good this new oscillator was. To better measure its real world performance I used the following setup.





I set up another Qualcomm board I had for 2688 MHz. This frequency multiplied 9 times is 24192 MHz. The initial 10 MHz reference used for the Qualcomm was my HP Z3801A GPS Disciplined Oscillator. Utilizing this reference, the Qualcomm generated a very accurate 24192 MHz marker. I then set up my 24 GHz transverter and hooked the IF audio output to the line input on the soundcard in my laptop. I then adjusted the IF for a convenient audio tone and recorded it for a period of time. While still recording, I then switched the Qualcomm reference over to the new GPS oscillator. I continued recording so I could measure any frequency shift in the audio tone. This would represent the frequency difference between the simple GPS oscillator and the highly stable Z3801A. It also takes into account LO drift in the transverter. As it turned out, the LO was a significant portion of the drift I measured!

What I heard was very encouraging. I analyzed the audio tones before and after switching references using CoolEdit Pro. The worse case shift I measured was 50 Hz, but depending on the sample I used, it was as low as several Hz. I had also expected that the tone might be a bit rough due to some jitter, but it sounded quite pure.

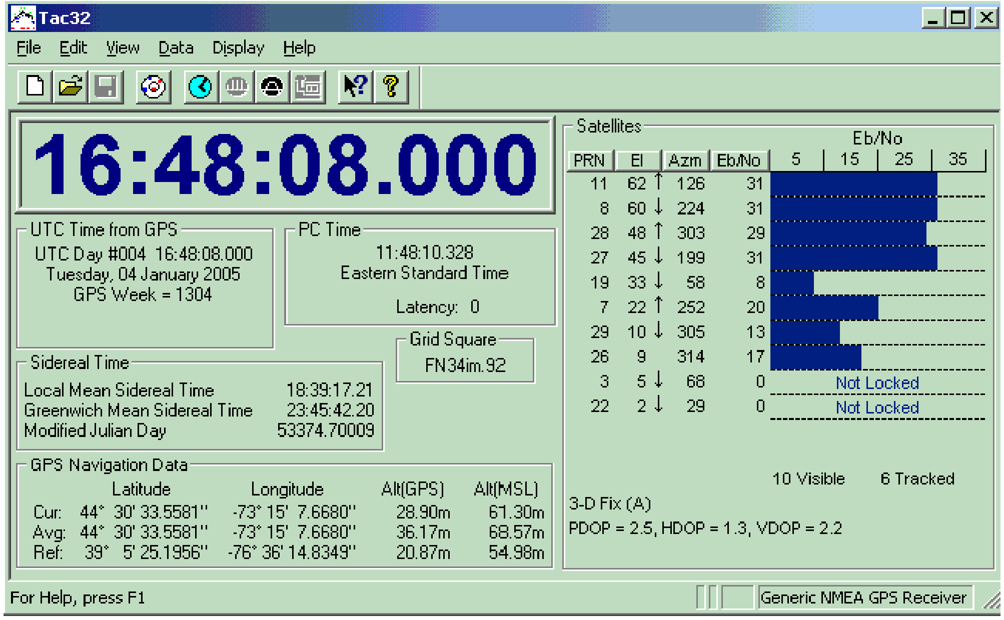
After evaluation by a number of us, we find you can expect the output to be within 1.5 Hz per GHz (± 0.00075 ppm). Don, W1FKF has done extensive measurements and his paper on using the oscillator as a reference for a microwave marker system bears out these findings. Overall, a respectable reference for such a simple design.

GPS ISSUES

One problem with the Jupiter engine is that from the moment it’s powered, it outputs 10 kHz and 1 PPS. Until I had a valid 3D GPS fix (I recommend at least 4 satellites) the output was not accurate. I needed some way to be able to verify that I had a valid 3D GPS fix before I could trust the 10 MHz output. The GPS engine does not provide any type of output such as a line that might go high/low to indicate lock, so I had to think of another solution. I’ve come up with several ways to do this and I’m sure there are more.

1. Use GPS monitoring software that can take advantage of NMEA-0183 output.
2. Use HyperTerminal or other terminal program to monitor the NMEA strings.
3. Measure the Voltage Control (VC) line on the VCXO.

For the first solution, I used Tac32 from CNS Systems[9]. I purchased this software a number of years ago when I was working with a Shera[10] GPS stabilized oscillator. It can easily monitor the Jupiter engine and graphically display lock status and also the quality of the lock. One bonus is that it will also display an 8 digit grid square so I can begin to approach the K2RIW standard of precision. ☺ Finally, it has the ability to set the clock on the host PC.



16:48:08.000

UTC Time from GPS
UTC Day #004 16:48:08.000
Tuesday, 04 January 2005
GPS Week = 1304

PC Time
11:48:10.328
Eastern Standard Time
Latency: 0

Grid Square
FN34im.92

Sidereal Time

Local Mean Sidereal Time	18:39:17.21
Greenwich Mean Sidereal Time	23:45:42.20
Modified Julian Day	53374.70009

GPS Navigation Data

	Latitude	Longitude	Alt(GPS)	Alt(MSL)
Cur:	44° 30' 33.5581"	-73° 15' 7.6680"	28.90m	61.30m
Avg:	44° 30' 33.5581"	-73° 15' 7.6680"	36.17m	68.57m
Ref:	39° 5' 25.1956"	-76° 36' 14.8349"	20.87m	54.98m

Satellites

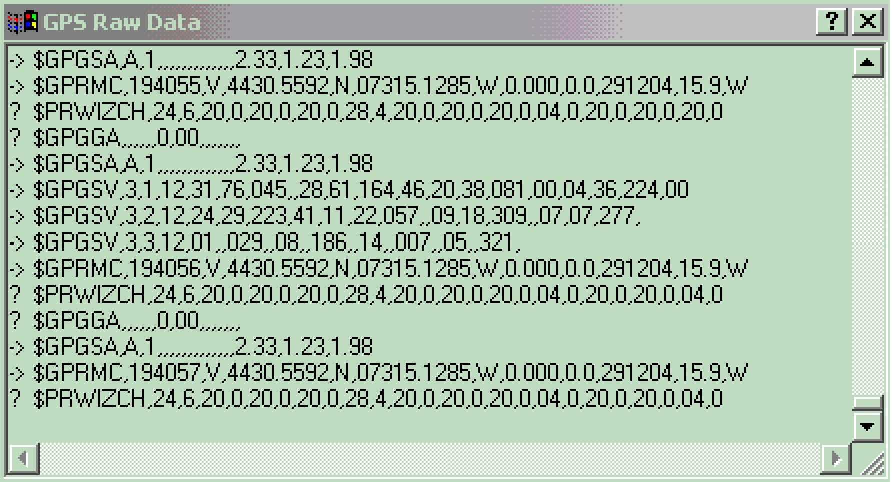
PRN	El		Azm	Eb/No	Eb/No 5 | 15 | 25 | 35
11	62	↑	126	31	
8	60	↓	224	31	
28	48	↑	303	29	
27	45	↓	199	31	
19	33	↓	58	8	
7	22	↑	252	20	
29	10	↓	305	13	
26	9		314	17	
3	5	↓	68	0	Not Locked
22	2	↓	29	0	Not Locked

10 Visible 6 Tracked

3-D Fix (A)
PDOP = 2.5, HDOP = 1.3, VDOP = 2.2

For Help, press F1
Generic NMEA GPS Receiver

Solution number two involves simply setting up HyperTerminal or other terminal program for 4800 baud N,8,1 with no flow control and connecting it to the GPS. What you will see is a series of NMEA strings flow by. Below is an example of this raw output.



```
-> $GPGSA,A,1,,,,,,,,,,,,,2.33,1.23,1.98
-> $GPRMC,194055,V,4430.5592,N,07315.1285,W,0.000,0.0,291204,15.9,W
?  $PRWIZCH,24,6,20,0,20,0,20,0,28,4,20,0,20,0,20,0,04,0,20,0,20,0,20,0
?  $GPGGA,,,,,,0,00,,,,,,,
-> $GPGSA,A,1,,,,,,,,,,,,,2.33,1.23,1.98
-> $GPGSV,3,1,12,31,76,045,,28,61,164,46,20,38,081,00,04,36,224,00
-> $GPGSV,3,2,12,24,29,223,41,11,22,057,,09,18,309,,07,07,277,
-> $GPGSV,3,3,12,01,,029,,08,,186,,14,,007,,05,,321,
-> $GPRMC,194056,V,4430.5592,N,07315.1285,W,0.000,0.0,291204,15.9,W
?  $PRWIZCH,24,6,20,0,20,0,20,0,28,4,20,0,20,0,20,0,04,0,20,0,20,0,04,0
?  $GPGGA,,,,,,0,00,,,,,,,
-> $GPGSA,A,1,,,,,,,,,,,,,2.33,1.23,1.98
-> $GPRMC,194057,V,4430.5592,N,07315.1285,W,0.000,0.0,291204,15.9,W
?  $PRWIZCH,24,6,20,0,20,0,20,0,28,4,20,0,20,0,20,0,04,0,20,0,20,0,04,0
```

These strings are simple comma delimited ASCII text. They can indicate whether you have a lock or not. The three strings we want to look at are GPGGA, GPRMC and GPGSA. In the above example, the GPGGA string starts with a series of commas followed by the number 0. If you have a valid lock, the number will be either 1 for a valid GPS fix or 2 if you're using DGPS and have a solution. This only indicates that you have a fix. What we really want is a 3D fix. When you look at the GPRMC string, the V near the beginning of the string indicates an error. An A in this position would indicate a valid solution. The best string is GPGSA. The second field in the sentence in the above example is a 1. If there were a 3 in this position, it would indicate a valid 3D GPS fix.

Option number three involves simply measuring the VC line of the VCXO. If you have a valid GPS lock and the VCXO is stabilized, the voltage should be at the center of the tuning range. For instance on my FOX, it's ~1.5 VDC. One word of caution, since the VC line is what controls frequency be careful how you measure it. Any disturbance to the line can pull the VCXO!

Another approach has been developed by Tommy, W1AUV. He programmed a PIC to monitor the NMEA GPGSA string and turn on an LED when there was a valid 3D lock. This is the same way I suggested in option #2 above, but doesn't require a computer.

FINALLY…

The system has easily met my design criteria. Based on the performance of this marker system so far, it should be a big help in the 10 GHz and Up Contest. If both stations are utilizing this same reference, we should be able to reliably calibrate frequency and minimize another of the "variables" that can hinder the completion of a successful contact.

During the June 2005 VHF contest, the system was utilized at both ends of a 205 km 24 GHz contact between myself on Mt. Washington (FN44ig) and Don, W1FKF on Mt. Wachusett (FN42bl). After we both calibrated our frequency, Don began to transmit. I instantly heard him with no retuning of my IF. The weather conditions were very poor that day and the signal was extremely weak, but having calibrated frequencies made it easy to work.

If you're interested in this project, please feel free to contact me at n1jez at verizon dot net.

References:

1. http://www.jrmiller.demon.co.uk/projects/ministd/frqstd.htm
2. http://www.ham-radio.com/sbms/sd/projindx.htm
3. http://www.delorme.com/
4. http://pulsar.princeton.edu/~joe/K1JT/
5. http://www.pentalogix.com/
6. http://www.expresspcb.com/
7. http://www.foxonline.com/
8. http://www.mouser.com/
9. http://www.cnssys.com/
10. http://www.rt66.com/~shera/index_fs.htm