Narrow Span High Resolution PLL synthesizer

Kasper Pedersen
kkp2010@kasperkp.dk
rough draft 2, 2010.03.31

Abstract

Classic variable divider and fractional divider PLL synthesizers have wide tuning range but coarse resolution. We describe a digital-domain PLL that is suitable for narrow tuning range (10^{-5}) and arbitrarily fine resolution (10^{-12} or better). It is most suited for oddball frequencies, and not so much for straight integer divisors.

A test implementation was built at a cost of eur.2 for the loop controller. A microcontroller implements all functions.

PLL designs

In the classic PLL, a phase detector is fed with two signals of mostly-identical frequency. Dividers are used to prescale the reference and the VFO to achieve frequency scaling, and the resolution of the PLL system is set by the resolution of these dividers. Where integer dividers won't do, fractional-N dividers are an option. The downside to the fractional-N divider is that, as the resolution goes up, so does the low-frequency noise, and thus close-in noise around the VFO output. If µHz resolution is needed, no analogue filter will work.
An alternative to the fractional-N divider is to replace a divider on either the reference path or the vfo path with a numerically controlled oscillator (DDS). This does work very well but is power hungry and expensive.

A post-mixer offset approach

Compared to a standard PLL, there is one fundamental difference in the arrangement in fig.1: The oscillator to be disciplined, and the reference, are both applied to to the phase detector as-is, at very different frequencies. The phase detector is sampled, and produces an instantaneous phase value. This value is compared against a predicted value, and the error between these two values is what drives the loop. This arrangement moves the frequency control from a divider before the phase detector, to after the phase detector. Here frequency control can be introduced by slewing of the phase.
This arrangement cannot easily be built in the analogue domain, as the output of the phase detector, adder, and offset generator require circular-number-system outputs.

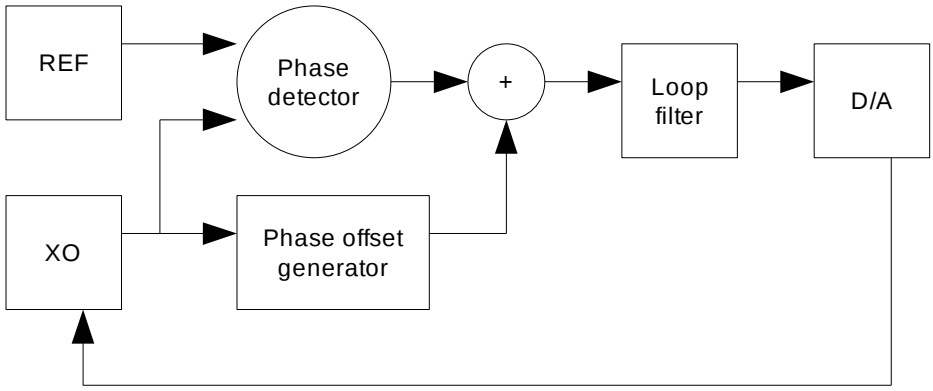


fig.1

Phase detector implementation

An all-digital detector is possible when REF and XO have no large common divisors. In the test implementation, REF is a 14.7456MHz OCXO, and XO is 20MHz.
These have a suitably small greatest common denominator, 6400 Hz. In 1/6400 second, REF will complete 2304 cycles, and XO 3125.
If the phase detector samples 3125 times, it will get 3125 equidistant points on the REF phase circle. The point spacing is then $GCD(F_{XO},F_{REF})/(F_{REF}*F_{XO})$ = 22 ps, leading to decent phase resolution.

The ATMega168 used for the test implementation can not do this. Instead it samples at $3/F_{XO}$ intervals. It is still possible to get the full resolution by staggering sample runs with 1 clock delay, so a slower microcontroller may keep resolution at the expense of time.
When time is not available, a lower number of samples may be used, though picking a good number becomes a manual endeavor.

The selection of sampling interval and number of samples is done by computing the phase point of each sampling point, verifying distribution, and then dividing the sampling points into a suitably large

number of buckets. For the test setup, 321 samples at 3 clock interval was eyeballed to be a neat, acceptably equidistant point constellation.

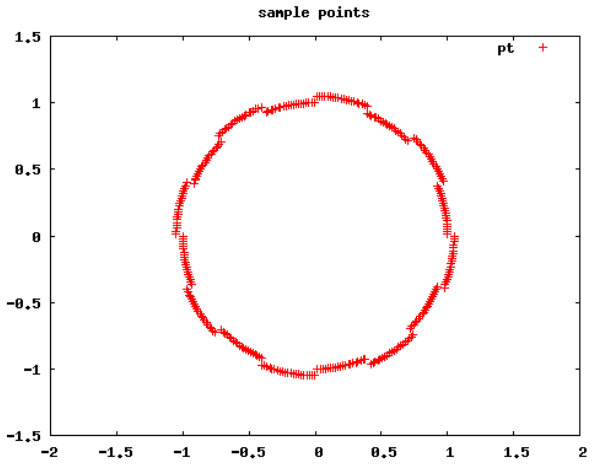


fig.2

The sampling points are here divided into 16 buckets, each bucket being 20 or 21 samples. A run of 321 sets of machine instructions is generated, each set doing a single sample and then increment the bucket-count if REF was found to be high at that instant. After the bins are filled, the phase can be found by finding the center of distribution.
The theoretical single-shot resolution is $1/F_{REF}/321=211$ ps. No external hardware is required, REF is directly applied to an input pin, and the internal synchronizer flip flop is used as the sampling element. An external flipflop will have better performance even at 211ps resolution.

A complication is that the sample run will not start phase synchronously with the REF/XO beat, giving us random-looking phase. This will be handled by the offset generator, and helps move post-filtering resolution closer to the theoretical 22ps.

The output of the detector is arbitrarily chosen to be 2048 divisions.

Phase offset generator

The offset generator/predictor is invoked after the phase detector has done a sample run. The sample run is time stamped, allowing the predictor to simply calculate $\varphi=t^* d\varphi/dt$. If the system is to be run-time adjustable, it is better to calculate $\Delta\varphi=\Delta t^* d\varphi/dt$, $\varphi_n=\varphi_{n-1}+\Delta\varphi$.

To do the above calculation without requiring excessively long integers, we can exploit that the signals are periodic.
$\Delta\varphi=\Delta t^* d\varphi/dt = (\Delta t \bmod (F_{XO}/GCD(F_{XO},F_{REF})))^* d\varphi/dt$
Thus, for the test implementation, a modulus of 3125 can be applied to Δt before multiplication. In a 32-bit integer this leaves 1.3M counts of usable resolution.

In the test implementation, the phase step (in periods) should be 2304/3125=0.73728. In 24.8 scaled fixed point, the per-XO phase increment should be 0.73728 * phase divisions per turn * 2^8 =386547.057, rounded to 386547.
The rounded value corresponds to a phase advance per XO clock of 0.73727989, an error of 2.16Hz, or 0.11 ppm.

If this accuracy is insufficient, or adjustment is needed, additional or higher resolution generators may be used. For the test implementation, an additional generator with variable increment rate was used. A trim value is accumulated into a 32 bit variable at a rate of 78.125kHz, and the topmost 11 bit (for a phase span of 2048) are added to the above compensation. The resolution thus becomes $78.125\text{kHz}/2^{32}$ = 18μHz. This is against REF, so this translates to a resolution of $1.233^* 10^{-12}$.

It is not necessary to run a fast interrupt to do the 78kHz accumulation; It is more efficient to calculate the number of 78kHz periods that have passed, and then compute the accumulated value whenever required.

Filter
When the predicted value is subtracted, the output is the error signal. It will be overlaid with the resolution noise from the phase detector, pulling effects in the microcontroller inputs, phase jitter on the microcontroller clock inputs, and other sources. The filter tau should be chosen so it matches the crossover where the XO's (rising with tau) instability exceeds the (falling with tau) filtered noise from the phase detector.
For the test implementation, the microcontroller does 10k acquisitions per second, so for a 1-second tau the noise is down by about 100x, and found to be equivalent to ~3ps rms before the DAC.

Unlike for an analogue PLL, the filter does not need to be wide to allow lock-in. The input to the filter should not be the raw phase signal. Instead, on every acquisition, transfer the change in phase onto an accumulated phase value. This will provide continuous phase to the filter when the phase detector rolls over.

DAC implementation
The test implementation uses dual 78kHz PWM channels summed in hardware for 15 bit resolution. Linearity and offset do not matter much, as long as it is monotonous. The PWM is filtered with a 2nd order 10ms filter, and applied to the XO varactor. Tuning range is +/- 25 ppm.

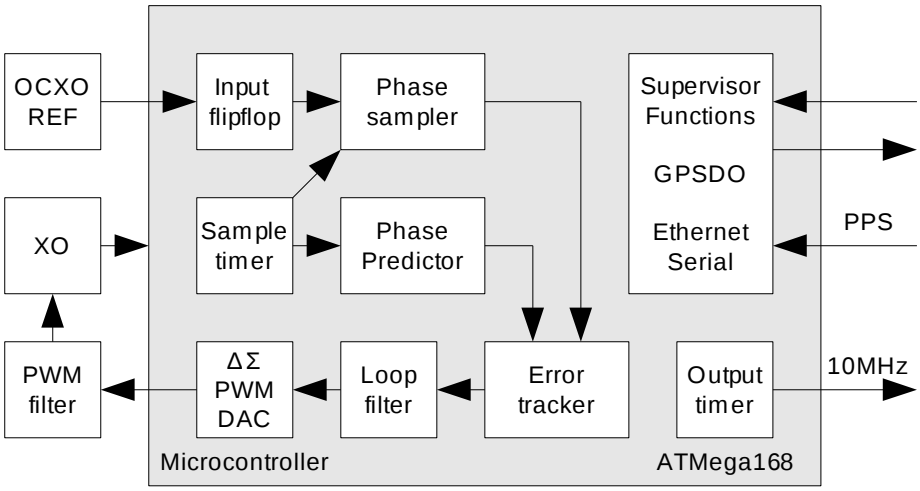
It is possible to add ΔΣ modulation techniques to the PWM. It should be noted, though, that since the phase detector needs the occasional run of ~1000 uninterrupted clock cycles, interrupting at high rate to create a fast software ΔΣ in the same microcontroller will break the phase detector.

Results

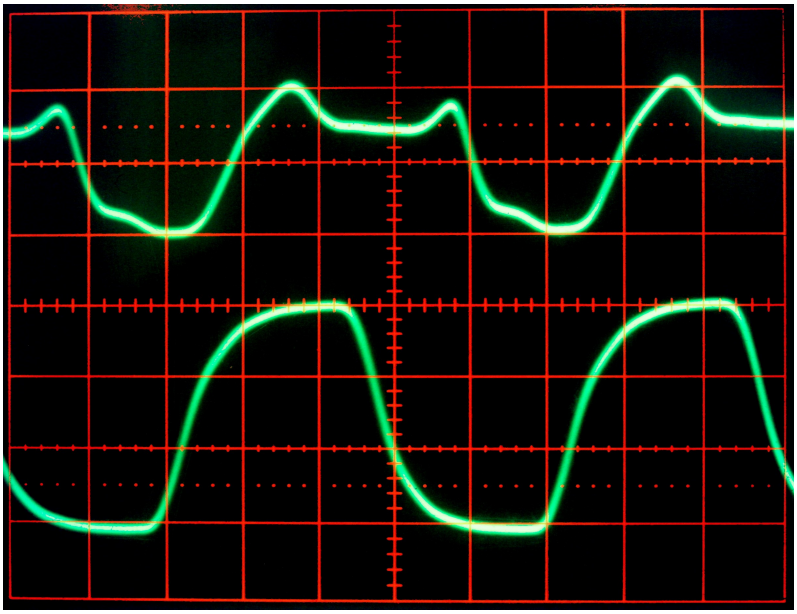
The quite unstable on-chip oscillator of the ATMega168 locks to the OCXO in time set by the filter. 50ms is suitable since it is decidedly noisy, and a lower value would be better but the current external filter prevents that.

A second test setup was built, with a better quality 20MHz XO, and an outer loop locking to an external pulse-per-second signal by adjusting the phase predictor. This setup has the medium term stability of the non-adjustable OCXO, and the close-in noise figure of the XO. Compared to a conventional GPSDO, it has no drift from reference voltage, DAC gain, or varactor path aging.

Putting a narrow bandwidth, high resolution PLL synthesizer in a microcontroller is quite doable, and beats a DDS-based approach on noise figure, power, and cost.



In addition to running the phase detector, PLL, and GPSDO, there is space in the ATMega168 for limited software Ethernet transmission for reporting and time mark generation, an ugly hack that allows commands to be sent to the device over Ethernet, 10MHz output generation, and generation of two marker signals with 50ns step resolution. The complexity is in the design of the phase detector, not in the implementation.



10MHz house standard on top, not properly terminated.

10MHz source under test below. Stability is provided by the 14.7MHz OCXO, and the 'virtual VC-OCXO' disciplined against GPS.

Scope is triggered on the house standard for both channels. Exposure time for this image is 10 seconds.

During a day, the lower trace will move about by about 20 ns, equivalent to 6m; The GPS has a rather poor view of the sky.