"Geared" Oscillator Project
Final Design Review

Nick Edwards
Richard Wright

This paper outlines the implementation and results of a variable-rate oscillating clock supply. The circuit is designed using a 130nm process, and a supply voltage of 3.6V to our internal power regulator, with an average of 1.6V output from our oscillator. Internally, the circuit is composed of 72 states, each of which produces a frequencies ranging from 575 MHz to 3.03 GHz. These frequencies are divided into 6 general ranges by varying the number of inverters in the oscillator. Each of these ranges is further divided into 12 finer-grained ranges by varying the rail voltage to the oscillator. Detailed frequency values are shown in this paper, along with typical power consumption.

Introduction

Our project is a clock generator which can be implemented on-chip, without the use of an external crystal oscillator. This is done through the use of a specialized ring oscillator and a custom variable rail voltage supply for the oscillator. The clock has a variable frequency in order to compensate for fabrication variance, and also to provide dynamic control over the clock frequency. This clock generator can be useful in situations where cost prohibits the use of an external clock, or for cases where a clock with dynamic frequency changes are desired.

One of the best uses for our proposed circuit would be for a locally clocked circuit inside of an asynchronous system. In that case, the oscillator would provide the clock to the single circuit without any outside interference or clock requirements. In addition to locally providing a clock, the frequency of the clock can be increased or decreased to match actual delays rather than being limited to worst case delays. This can make the clocked section more robust to temperature, power, and process variations.

For instance, instead of having a static external clock, this internal oscillator's speed will depend on local process, temperature, and power variations in the region of the circuit that it drives. In addition to that, sensing circuitry can be placed in the critical paths to determine if the outputs changed after the setup and hold time needed for the flip flops. In that case, it can slow the system down and re-compute the previous calculation. Inversely, if the circuit senses that the calculations are completed early for all critical paths, then it can gently speed up the dynamic clock generator. Thus, the clocked system's delay will be related to the actual delay of the circuitry, including the process and environmental variations currently present.

Block level designs

In order to have fine grained control over the output of our oscillator, we have decided to dynamically control the power supplied to the oscillator. We researched similar systems and PLLs and found that a buck converter is frequently used in such a system because of its simplicity and ability to be controlled by a simple set of signals. A simple buck converter is shown in Figure 1, in which the switches S1 and S2 are complimentary to each other (i.e. when one is on the other is off). By just looking at first order effects, the output voltage will be the input voltage scaled by the percent of time that S1 is closed, assuming that the frequency of switching of the switches is high enough. Thus, as long as things are scaled properly, the output frequency depends on the input voltage and the duty cycle of the alternating signal that alternates the signals going to S1 and S2.

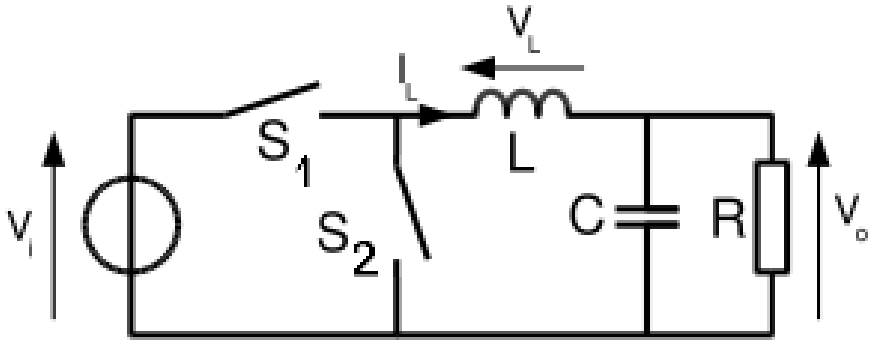

Figure 1. Buck Converter

In order to create a high frequency, alternating signal, we designed a duty cycle generator out of another, smaller, ring oscillator. If the transistors in the inverters are sized to have similar pull up and pull down characteristics, then the duty cycle will be about 50%. However, other duty cycles are obtainable by biasing the rising and falling times on the inverters, as well as by inverting them to double the number of duty cycle signals. In order to get the control we desire, we have 6 duty cycle generators that create different duty cycles. Since only one of the duty cycle generators is needed at any given point in time, we have included an enable signal to disable them while not in use, so as to limit the active power usage of our design. A general example of our duty cycle design is shown in Figure 2.

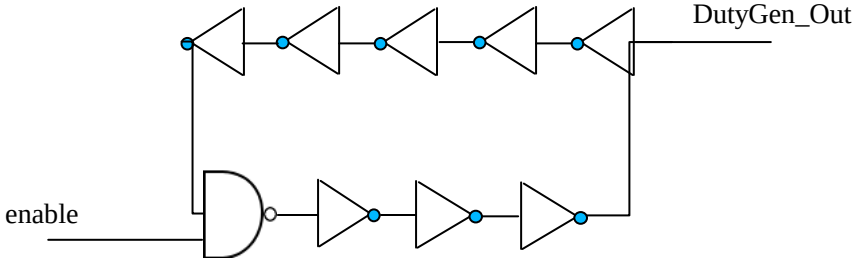

Figure 2. Duty Cycle Generator

Our initial design was to make half of the inverters have a pull up strength that was six times larger than their pull down strength. Then, the other half had pull down strengths six times larger than the pull up strength. Since a ring oscillator has an odd number of

inverters, the output was attached between two ‘large P’ inverters that we connected in a row. With these strengths, the duty cycle generated with a nine inverter oscillator was 86%. This was significantly higher than we expected. Figure 3 shows the output wave form of such an oscillator.

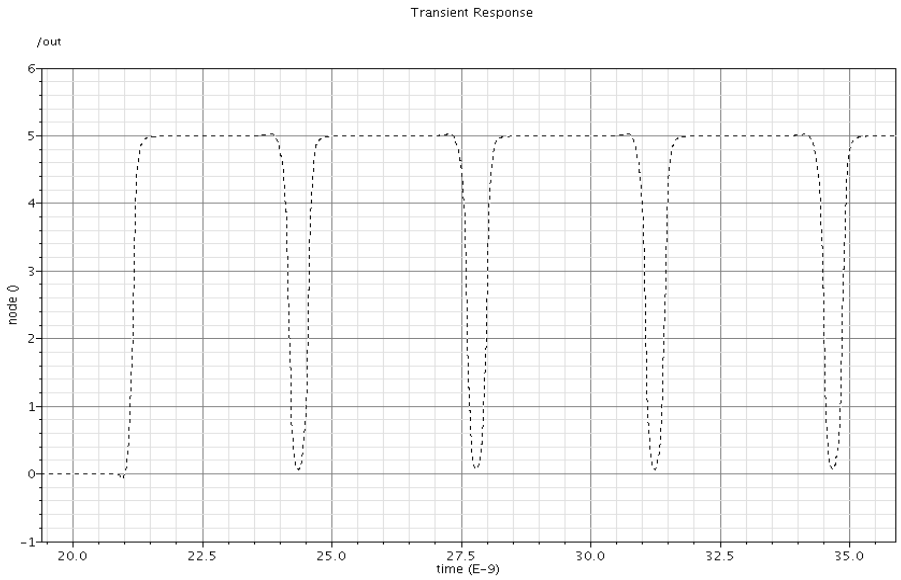

Figure 3. Duty Cycle Output

In order to dynamically change the output voltage on the buck converter, multiple duty cycle generators are connected through an OR gate to its input. Figure 4 shows how this is to be done logically. An un-depicted logic controller will control the enabled signal to the duty cycle generators. If only one is selected at any given time, and if the others are all outputting 0 to the OR gate, then the output of the OR gate will be the same as the selected duty cycle generator. The XOR gate in this circuit is used to invert the duty cycle, basing its functionality upon the fact that if you invert a 60% duty cycle, then you get a 40% duty cycle out of the inverter. In this way, we can get the same number of different duty cycles, with only having to create half as many generators for them.

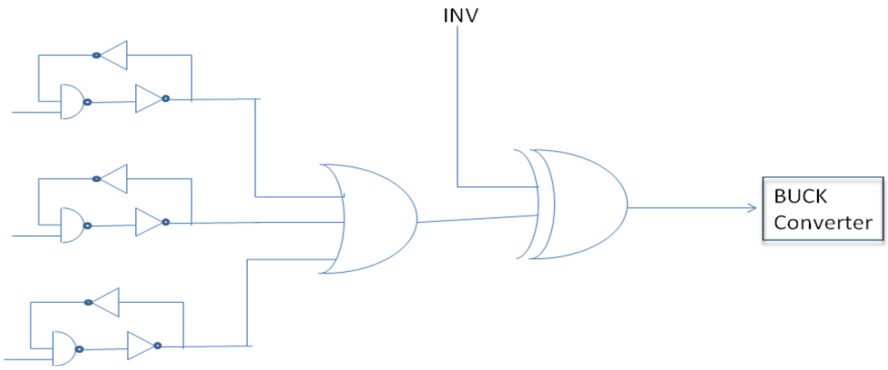

Figure 4. Duty Cycle Selector

The main oscillator used to generate the clock frequency is a “geared” oscillator. The main goal is to be able to dynamically shorten the path in the oscillator to increase the frequency. By removing inverters from the path in pairs, the frequency increases. Figure 5 illustrates the basic principle behind this. Our actual oscillator uses “gears” of length 9, 11, 13, 17, 23, and 31 transistors. In order to select the desired path, we use a set of mux gates built with pass logic.

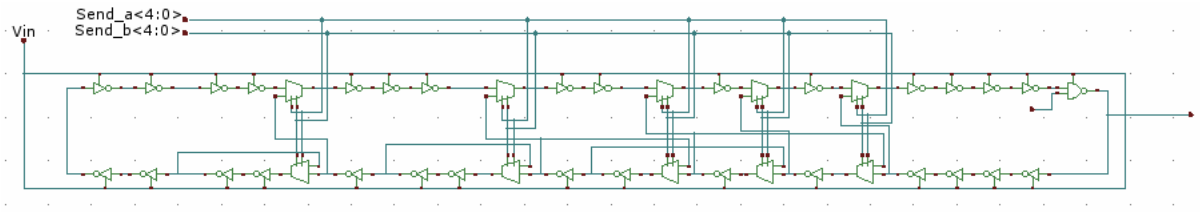

Figure 5. Geared Ring Oscillator

The overall system design is shown in Figure 6. The functionality is that the control logic is able to modify the length of the inverter chain for the main oscillator as well as being able to control the power going to the oscillator. In this way, it will be able to fine tune the frequency by modifying the supply voltage coming out of the buck converter. When it has reached an edge on its control with the supply voltage, it will be able to remove or add extra delay into the oscillator, while at the same time reducing or increasing the supply voltage to compensate. This allows for a gradual frequency change, even while lengthening or shortening the path in the oscillator.

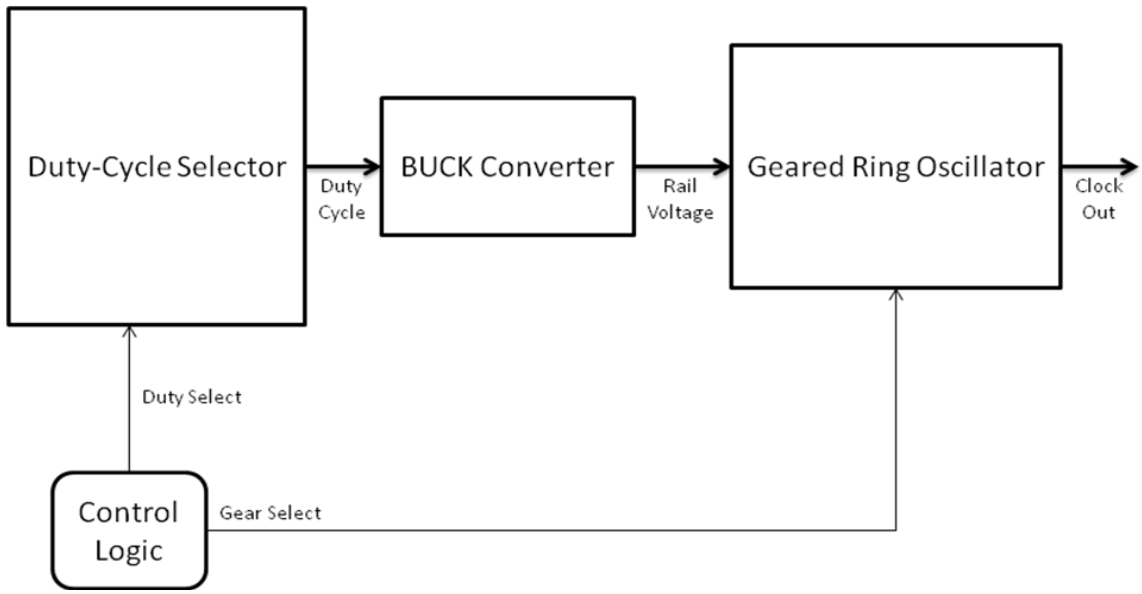

Figure 6. Top-most Block level design

Golden Model

The overall test point of our project is in measuring the clock frequency which is output from the circuit and comparing that to the input selection signal which we are sending into the circuit. This lets us know that the correct duty cycle and oscillator short-circuiting are being used. Since this is not a digital test, we will not be able to have it automatically scripted. Rather, we will need to manually go into Spice and test to see that the correct clock frequency is being generated.

As a mid-point test, we can also check the output voltage from the Buck converter to verify that the converter is working properly, and that the correct duty-cycle generator is being selected.

Area, Performance, and Power (estimates)

Area:

~10 duty cycle generators: (8 inv's + 1 NAND) * 10 = ~200 transistors
OR tree + Invert select: (9 + 1) NORs = 40 transistors
Buck Converter: 2 transistors + 1 resistor + 1 inductor + 1 capacitor
Ring Oscillator: 35 inverters + 10 shorts = (70 + 20) transistors

Total Area: ~332 transistor + 1 resistor + 1 inductor + 1 capacitor

Power:

While we are not sure at this time as to the power estimates of our circuit, we have been keeping this in mind during our design. For example, only one duty cycle generator can be active at a time. While each of these components is not in use, they will not remain in transition. Figure 7 shows the power consumption of the 31 inverter length oscillator running with the different duty cycle generators.

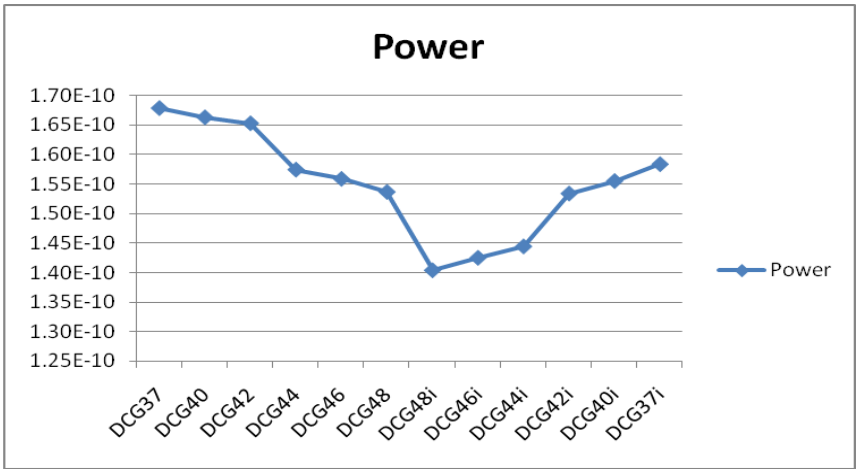

Figure 7. Graph of average frequencies

Performance:

Our oscillation range is between 575MHz and 3.03MHz. If lower clock speeds are desired, clock dividers may be used in conjunction with out circuit.

Testing Methods

The interface for this consists of a completely analog environment. We used timed AC voltage supplies to provide inputs for the select wires. For verification we used hSpice to verify that the correct clock frequencies are being generated. We used two main testing methods to verify our circuit.

First we ran extended tests on each of our 'gears', covering all 72 permutations of our select inputs. This allowed us to test the stability of the system given any input vector. We were able to observe the jitter and noise associated with our output signal. We also were able to quantify the actual response of the circuit, given a particular set of inputs.

Then, we ran tests in order to see how the circuit performs while switching between gears. This allowed us to quantify how well the system would work in a dynamic environment in which it switches between different frequencies. We tested in both directions, going from low frequencies to high frequencies, as well as testing from high frequencies to low frequencies. Both of these directions have their own challenges, especially when changing the length of our oscillator chain.

Results

Table 1 contains the average frequency of 100 clock cycles while running with each of the possible input vectors. We were expecting some overlap between the input vectors when switching from highest power in one gear to lowest power in the next gear, but we didn't see that behavior. In order to get this overlap, we would have needed either more duty cycle generators, or else to have spaced them out further from each other.

	9	11	13	17	23	31
DCG37	2486.069252	1686.917455	1255.351451	930.9622184	711.8532003	575.6408216
DCG40	2535.857893	1720.129934	1279.10938	948.8984373	725.490118	586.870916
DCG42	2596.18483	1757.761153	1306.509928	968.7826184	740.7746973	599.3949101
DCG44	2633.372365	1782.305903	1324.202239	981.9131757	750.7667409	607.5267934
DCG46	2677.938552	1810.903965	1345.216905	997.2965967	762.7561878	617.183013
DCG48	2731.052759	1844.552308	1368.66033	1014.998677	776.1460889	628.0655845
DCG48	2889.073401	1947.152975	1441.817048	1068.334807	816.910793	661.4955703
DCG46	2921.723641	1965.718819	1456.140228	1079.002183	825.2168844	668.4153788
DCG44	2950.561292	1984.263008	1469.345764	1089.042248	832.5191883	674.3494314
DCG42	2970.14981	1995.851385	1477.751797	1094.990312	837.3052883	678.2105146
DCG40	3002.189064	2015.457889	1492.189223	1105.454362	845.344498	684.6141091
DCG37	3031.317848	2033.680873	1504.870876	1114.701376	852.3157343	690.5825124

Table 1. Average frequencies over all 72 input vectors

As you can see in Figure 7 we did get reasonable coverage over a large set of frequencies (between 575 MHz and over 3 GHz). Figure 8 shows these same values graphed together on a logarithmic scale, which better displays the relative frequency jumps. One anomaly that we found was that our XOR gate was skewing the signal between the duty cycle generator and the buck converter. This skew is different when it is inverting and when it is not inverting. This causes the difference in slope between each of the groups of six values that you can see in both of these figures.

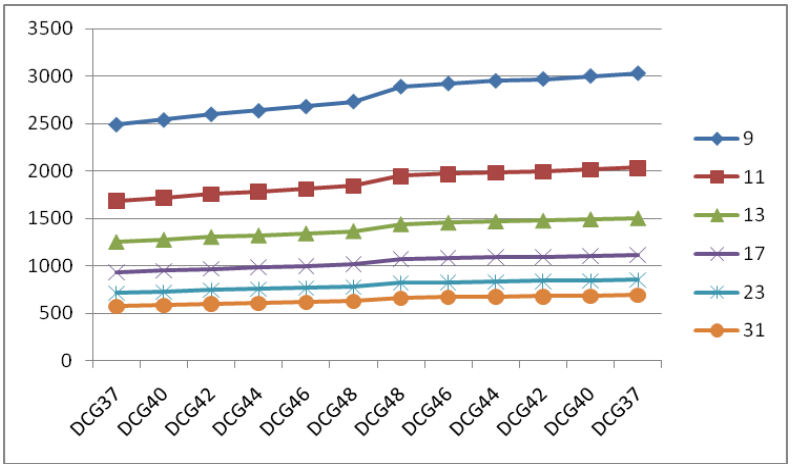

Figure 7. Graph of average frequencies

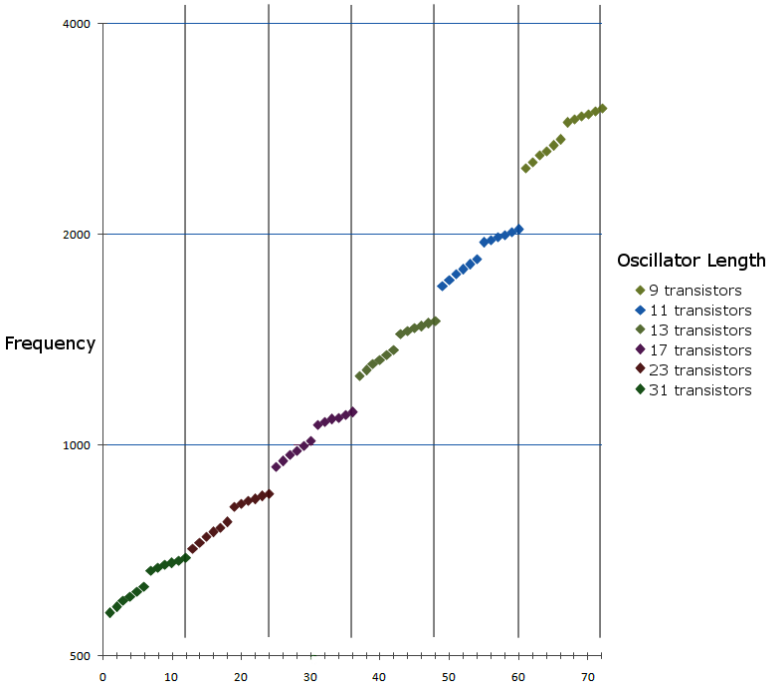

Figure 8. Average frequencies graphed on logarithmic scale

We also wanted to quantify the amount of skew that we have when running with one input vector. Figure 9 shows an example of the jitter between clock cycles. It graphs 100 clock cycles of our circuit with the inputs selecting our longest "gear" in the oscillator (31 inverters) with the 40% duty cycle coming into it. As you can notice, the period varies approximately by 6ps over the 100 cycles. We found 6ps – 8ps variance to be pretty average over our set of 72 simulations.

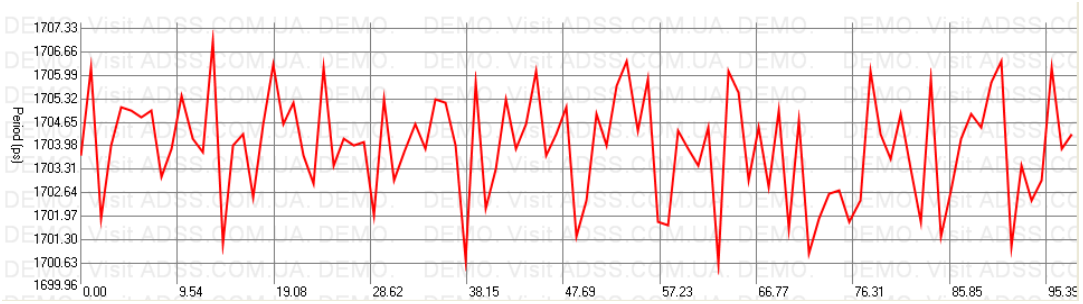

Figure 9. Sample output of 31 inverter chain and 40% duty cycle

Our first simulation when running the full circuit spanning all possible input vectors yielded favorable results. Figure 10 shows the change in output frequencies over the span of time. We allowed 50ns with each input vector before moving on to the next vector. As you can notice, there are distinct peaks when switching gears in our oscillator by removing inverters from the path. The problem is that the oscillator gear switches instantly, but the power takes about 35ns to adjust to the new voltage level. This causes these major jumps when switching gears. There is also a minor jump when switching our duty cycles from non-inverted to inverted signals. This is caused by the skew added by our xor to do the inverting.

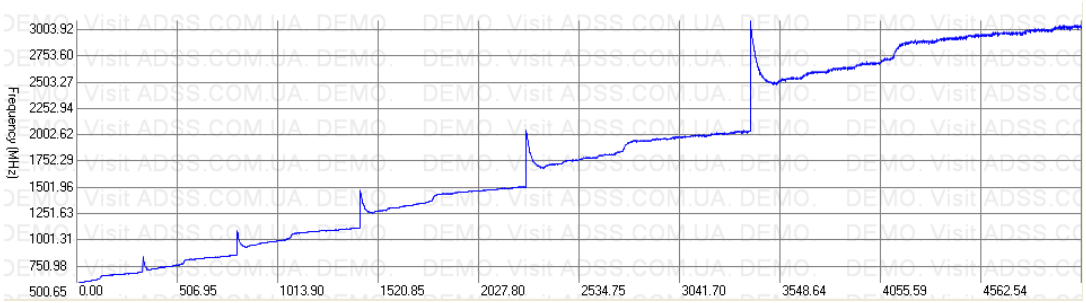

Figure 10. Oscillator moving between 575 MHz and over 3 GHz

In practice, it would be bad to have the frequency jump to a higher frequency that quickly in such a short period of time. In fact, it would be better to slow the clock down for a couple of cycles and then switch to the faster gear so that if the circuit can only handle a little bit faster frequency, it is never over-whelmed by the switch over. Thus, more sophisticated control logic is needed to start dropping the frequency before switching gears. Figure 11 displays this technique. There is still a sharp jump, but it drops down instead of up.

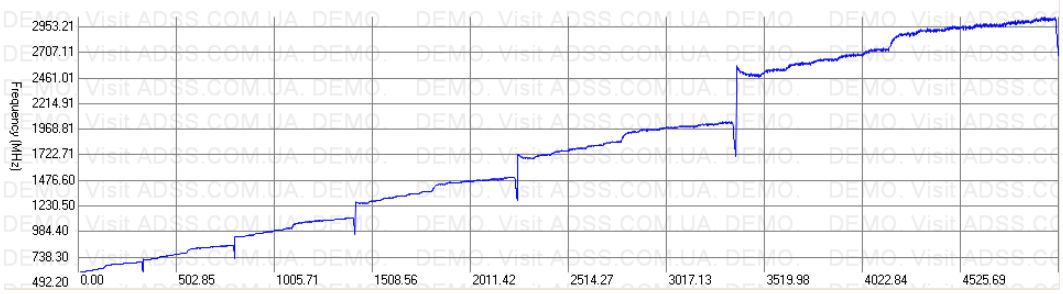

Figure 11. Oscillator that dips the frequency down before changing gears

In general, it is pretty much the same idea when we are going from a higher frequency to a lower frequency. We have worried about these transitions because of the possibility of starting multiple oscillations going around our oscillator chain at the same time. This is a possibility only when extending the chain, and not when shortening it. As you can see in Figure 12, this occurs the third time that it switches gears. This causes it to jump up above 5 GHz for a few cycles until the two internal oscillations merge together. We would need to develop a small sensing circuit to make sure that the phase is correct before inserting the extra inverters when switching gears.

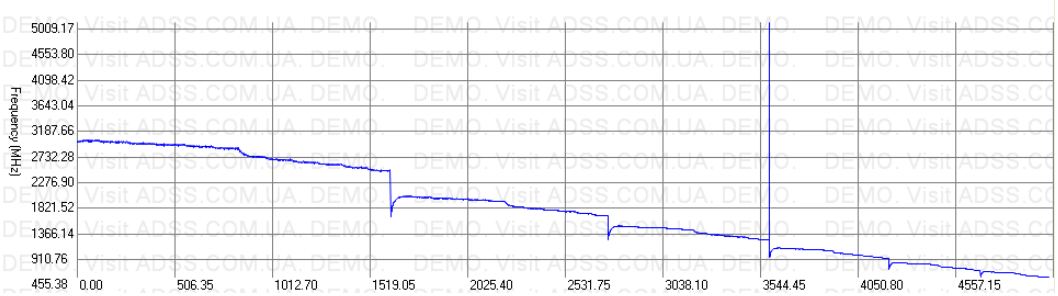

Figure 12. Oscillator moving between 3 GHz and 575 MHz

Conclusions

Overall, the circuit worked as we wanted to and as we expected it to work. The range of frequencies was very wide, covering from 575 MHz to 3 GHz. Although there were a few large gaps between frequencies that we could generate, we felt that the circuit covers the whole range fairly well.

We did find a few limiting factors to this design. The largest factor is the size of the inductor. In order to make a buck converter that is able to supply enough current to allow the oscillator to run at a specified voltage level, the inductor takes up too much area on a chip to make it practical for any real world applications. Also, there would be some smaller issues in reading the output signal and boosting it up to a constant voltage level before it could be used as a clock.

Although we were able to make it so that the jumps in frequency caused by switching gears in our oscillator would drop down and slowly rise up, it is still undesirable to have a large, instantaneous, change in frequency. For this reason, it would probably be best to use this type of a technology in a more design-specific manner. For example, you could design the oscillator to run at a much smaller range of frequencies and not have multiple gears. Instead, you could control the oscillator strictly by the voltage level provided by the buck generator. This would make the circuit much smaller and remove the large frequency jumps. In practice, an actual circuit will generally have a relatively small range of usable frequencies to which you can design the specific oscillator.