# Heating DDS relative phase drift

**original 14 Apr 2011**
**revised 29-Apr-2011**
**JM**

The wish is to change heater frequency under radar controller control, triggering new values from DDS modulation RAM into use via the UPD and STMC radar controller commands. A simple RC test program to do that is Mike's *medium.tlan*, below. This program switches between two frequencies at (almost) one second intervals.

medium.tlan

```
AT   0.4      STMC*,RXSYNC,RXP1ON,RXP2ON,RFON*,TXSYNC
AT   1000000  UPD*,RFOFF*,RXP1OFF,RXP2OFF,RXSYNC
AT   2000000  REP
```

Earlier in the spring Mike had noticed that when doing frequency hopping this way, the relative phase of the DDS output sinusoids sometimes jumped, see Mike's e-mail on 6-Apr-2011. Such jumping destroys any possibility of beamforming and can't be tolerated.

When Topi and myself visited Heating on 12–15-Apr, the problem was studied further. This note describes what Mike, Topi and myself found during the visit. The main conclusion after the visit is that we now understand the problem quite well, and believe that it can be fixed once-and-for-all by re-programming the DDS firmware and modifying slightly the RC programs.

* * * *

From Mike's report, the initial suspicion was that the jumping could be due to the 25 MHz free-running oscillator on the exciter. The free-running feature might somehow allow control signal timing errors to develop, most likely via what I will here call the *RC synchronization PEEL* (see Fig. 1 for the relevant part of the exciter schematics). The synch PEEL is a single chip, but it has enough IO pins to be able to provide the modulation control signals for both DDSs on the exciter board. In the exciter schematics, the PEEL is actually drawn as two separate boxes. However, while it was conceivable that the free-running feature would allow different exciter boards get out of synch, it was peculiar how already the two DDSs on a single exciter board were able to get out of synch, too. There was, after all, only the single oscillator, free-running or not, on any given exciter board. But recalling the very high frequency, 200 MHz, of the DDS system clock, we reasoned that perhaps the control signal timing requirements for the DDSs were so extremely stringent that the system simply was not able to meet them.

Only later during the week did I realize, from the DDS chip documentation like Fig. 3, that the timing requirements, though demanding, are not at all unreasonable, and the unavoidable sub-nanosecond timing jitter could not be the real cause for the phase jumps. Later still, I realized that actually the control signal timing jitter per see was not really the relevant issue at all. (For the impatient, Fig. 4 explains what we now understand is the decisive factor.)

The RC synchronization PEEL is used, among other things, to generate the crucial IOUPDATE signals to the two DDS chips on an exciter board, based on input signals from the radar controller. The IOUPDATE signal activates a new frequency value into use, by causing the value currently in the DDS input buffer latch to be transferred to the DDS frequency tuning word register. Thus it seemed reasonable that the IOUPDATEs should be precisely controlled in time, both within board and between the boards. With separate free-running oscillators controlling IOUPDATE-signaling on the different exciter boards, it appeared hopeless to ever be able to guarantee synchronism between the boards.

Our early idea had therefore been to replace the free running oscillators by phase-locked $\times 2.5$ clock multipliers (PLLs), to be locked to the system's common 10 MHz reference clock, and the PLLs had indeed been bought. But by Topi's suggestion, we now realized that we could eliminate the possible effects of the free-running oscillator more simply, by routing the presumably robustly phase-locked UPD and STMC RC signals essentially unmodified trough the synch PEEL. So the first thing we did was to program a new set of PEELs to that effect, and insert them in place.

The new PEELs indeed changed the character of the phase jumping. The jumps now appeared less random, less jitterlike. We used the *medium.tlan* file to alternate once per second between 2.0 MHz and 4.0 MHz. We noticed that instead of randomly occuring jumps, the DDS output phases drifted systematically with respect to each other, at the rate of about one period in 50 seconds. This was an encouraging change, but not yet the sought-for decisive improvement. Though later, Mike argued that for smaller, and less frequent, frequency hops, the drift would be slow enough to be manageable in realistic experimental situations.

On one exciter board, measurement of the signal timing shown in Fig. 2, revealed that the IOUPDATE signals to the two DDS chips, labeled IOUPDATE1 and IOUPDATE2 in Fig. 2, jittered a few ns. These signals are ultimately derived from the RC UPD and STMC signals. Jitter is expected. But significantly, there was also about 5 ns relative delay between the IOUPDATE generated from UPD and the IOUPDATE generated from STMC even on a single DDS, when measured as near to the DDS chips as we could get (after the TTK/LV drivers on Fig. 1). The signals from the RC to the exciter board appeared, as they should, to be aligned within one or two nanoseconds. So one way or another, the delay seems to develop within the exciter mother board. We did not verify, but it is conceivable, and even expected, that on some other exciter boards the delay could have been different, and even larger.

Topi's original guess was that this delay could be due to the input drivers for the UPD and STMC signals being configured differently. They need to be configured differently due to the UPD being an upgoing 100 ns pulse and the STMC being a downgoing 100 ns pulse, as shown in Fig. 2. We tested this idea by actually inverting the UPD pulses in TARLAN, and modifying the driver configuration for one exciter board, but this did not remove the delay. Instead the hardware fiddling necessary here resulted a certain soldering to break subtly loose on one DDS module. This in turn caused a major scare and a lot of debugging when the system mysteriously ceased to work. Ultimately, the UPD polarity and the modified board was returned to their original state. The actual reason for the relative delay remains unknown.

Mike also noticed that there appeared to be a tendency of the DDS drifts to group in only two groups so that within each group, there is no drift, but between the groups, there is the phase drift of one period per 50 s. We also noticed that if we modified the

*.tlan* files so that there was only a single STMC in the beginning, and then a lot of UPDs, the phase jump only happened once per radar loop.

In the night 14-Apr I sat down to try to really understand how small jitters (as we still thought at that time) could result in such systematic, repeatable drift as we had observed. My first observation, from Fig. 3, copied from the AD9953 data sheet, was that small jitter should not be significant. The frequency transitions happen a fixed amount (namely, one SYNC_CLK cycle, 20 ns), after the IOUPDATE is edge-detected, and the edge detection finds the IOUPDATE at the next rising edge of the SYNC_CLK after the IOUPDATE's rising edge. This arrangements is implemented internally in the DDS chip by a D flip-flop. The arrangement ensures that in most situations, small jitter in the timing of IOUPDATEs has no observable effect.

After realizing that jitter was not the issue, the next question was whether systematically different locations of the IOUPDATEs on different DDSs could cause the drift. So I draw a figure similar to the one shown on the top panel of Fig. 4. To my disappointment, it appeared that if the frequency on one DDS changes earlier than on another, and thus a phase difference starts to accumulate, the accumulated difference is precisely cancelled at the next frequency shift. No net drift could happen this way. The best I could get at by changing the timing of the IOUPDATEs was just to shift the overall frequency pattern, and no matter how much that was shifted, there would be no mean drift.

So, something stronger was necessarily required than just a shift of the pattern. What could that be? I realized that what was needed was to be able to change the pattern itself, not just to shift it. This would result in different mean phase accumulation rates, which is equivalent to a drift of relative phase. After this key observation, the next question was what precisely did it require to be able to change the pattern. It did not take long to notice that one needed two different signals to have enough degrees of freedom for fiddling. And, of course, I did have two timing-wise different signals available, the IOUPDATE derived from STMC, and the IOUPDATE derived from UPD. This could then result in situation like drawn in the bottom panel of Fig. 4, which exhibits the sought-for drift.

Summarising, the cause of the phase drift is that, *timing-delay-wise*, there exists two different kinds of IOUPDATEs. One is derived from STMC, drawn by the red dot in Fig. 4, and one is derived from UPD, marked by blue asterisks in the Fig. 4. The crucial thing is that depending where the IOUPDATEs happen with respect to SYNC_CLK rising edges, we can land into the frequency-wise non-symmetric situation $g$ shown in the bottom panel of Fig. 4, even when we actually programmed to achieve the symmetric situation $d$. There could also be even more skewed situations than $g$ but these would require relative delays over 20 ns between UPD and STMC on a single DDS. These should be rare if we have proper control of the internal timing.

The final step was to check if this idea would also work quantitatively. In our case, $F_1$ is 2.0 MHz and $F_2$ is 4.0 MHz. If one DDS follows the symmetric pattern $d$, and another follows the biased pattern $g$ that has an overdose of the higher frequency $F_2$, the latter has *larger average frequency*, and thus its phase advances faster.

How much faster does the phase advance in $g$ than in $d$ in the bottom panel of Fig. 4? Denote the total phase advance during a 2 s period in the case (d) by $\phi_d$ and the total phase advance during 2 s in the case (g) by $\phi_g$. We have

$$\phi_d = 2\pi F_1 \times 1s + 2\pi F_2 \times 1s\,, \tag{1}$$

and

$$\phi_g = 2\pi F_1 \times (1\mathrm{s} - 20\mathrm{ns}) + 2\pi F_2 \times (1\mathrm{s} + 20\mathrm{ns})\,, \tag{2}$$

so that the accumulated phase difference during 2 s is

$$\Delta\phi = \phi_g - \phi_d = 2\pi(F_2 - F_1) \times 20\,\mathrm{ns} = 360^\circ \cdot 2\,\mathrm{MHz} \cdot 20\,\mathrm{ns} = 14.4^\circ\,. \tag{3}$$

Happily, this rate is consistent with the observation that the phase takes 50 s to drift by 360°, for

$$14.4^\circ/2.0\,\mathrm{s} = 360.0^\circ/50.0\,\mathrm{s}\,. \tag{4}$$

To get rid of the drift we must get rid of the two different delays for the IOUPDATE-signals. And to do this in a robust way, it seems clear that there should be only one type of IOUPDATE-signals in use.

It appears that it is possible to arrange the IOUPDATE-signals to the DDSs to be generated solely by the UPD RC signals, by re-programming the synch PEEL still once. Internally on an exciter board, both the STMC and UPD signals are used for two quite separate purposes: first, generating the IOUPDATE which transfer the currently latched value from DDS input buffer into active use, and second, triggering the transfer of the next value from the modulation RAM to the DDS input buffer. The first of these tasks the two RC signals perform in the same way, except for the troublesome timing differences, and either one could in principle be used in the TARLAN programs. But for the second task, both must be kept available for the TARLAN programmer, because a STMC, in addition of initiating the data transfer, also resets the RAM address counter.

We propose to change the programming scheme so that the TARLAN code always specifies an UPD when an IOUPDATE is required, but in addition, specifies a simultaneous STMC-command when the address reset is required. On the face of it, this would generate an extra, unwanted, IOUPDATE signal and an extra data transfer from the RAM. But it seems possible to detect the simultaneous presence of the two signals at the PEEL inputs, and handle that situation separately on the PEEL, so that *only* the UPD-signal produces the IOUPDATE, and *only* the STMC signal triggers the data transfer. It appears that to be on the safe side timing wise, this requires that the STMC-pulse, which currently is 100 ns long like the UPD, should be made 200 ns long. If we so decide, the TARLAN side of the reprogramming could be done transparently to the user, by letting TARLAN-compiler to generate a 100 ns extra UPD when it sees an STMC-command. Then we only would need to recompile the existing *.tlan* files. If this works, we would in effect only ever use the UPD to generate the time-critical IOUPDATE-signals, and the frequency drift should entirely go away, as in the top panel of Fig. 4.

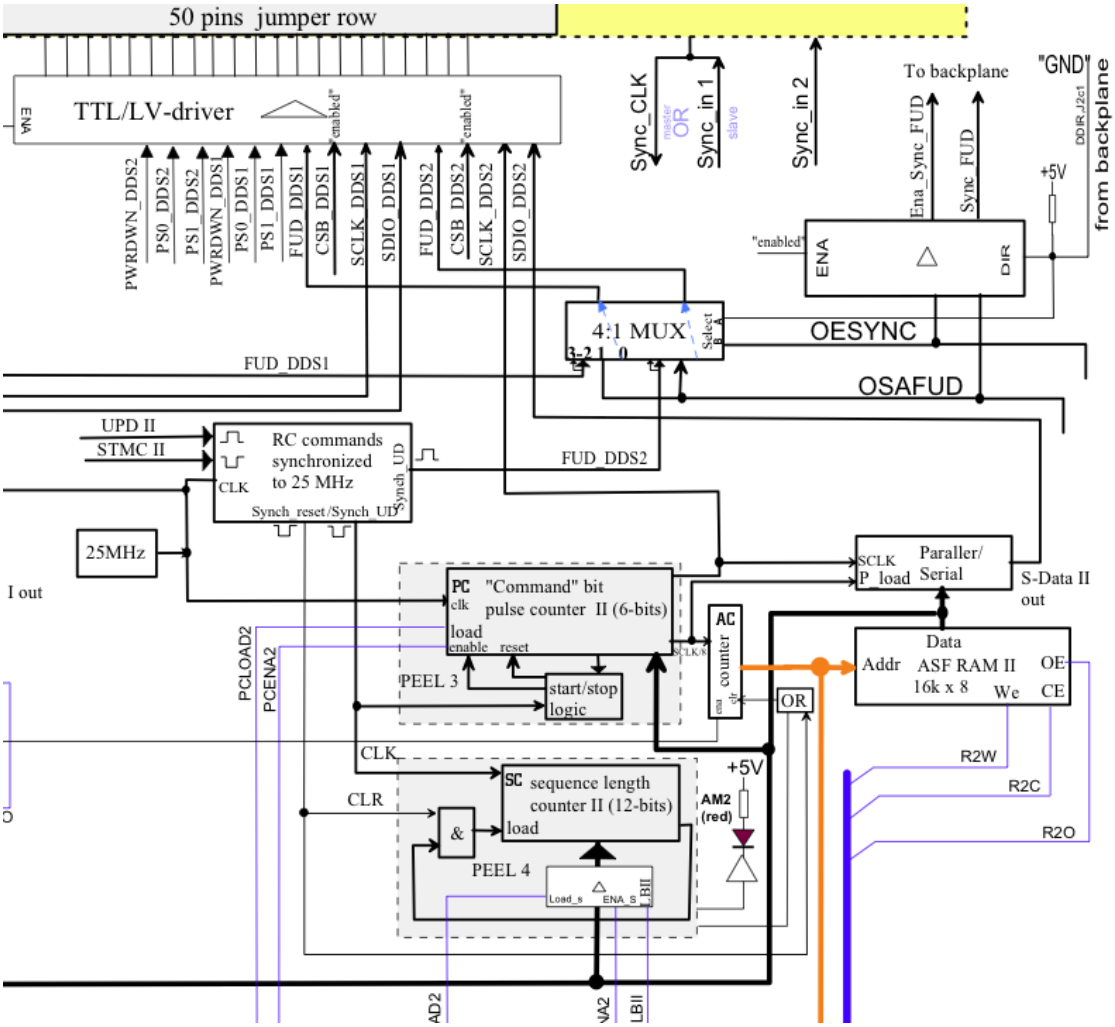



Figure 1: RC synch PEEL and its surroundings. There is, conceptually, a separate "RC command synchronizing PEEL" for both synthesizers on the exciter board, the one shown here is for the DDS unit 2. (Physically, the synch PEEL actually is a single chip, but it can be programmed as if having two independent channels.) The PEEL receives the UPD and STMC signals from the radar controller and a 25 MHz clock from the free-running oscillator, and sends out an IOUPDATE signal towards the DDS unit 2, and two control signals towards the modulation RAM control circuitry. Even though there are the signals UPDii and STMCii marked specifically for DDS unit 2 in the drawing, and (not shown in this cropped view) UPDi and STMCi for DDS unit 1, there actually are only one UPD and one STMC per exciter board coming from the RC. The intermediate 4:1 MUX -circuit has, similarly to the synch PEEL itself, two channels, and is able to feed both DDSs simultaneously. We verified that the MUX did not introduce any noticeble distortion (less than ns) in the relative timing to the throughpassing IOUPDATE1 and IOUPDATE2 signals.

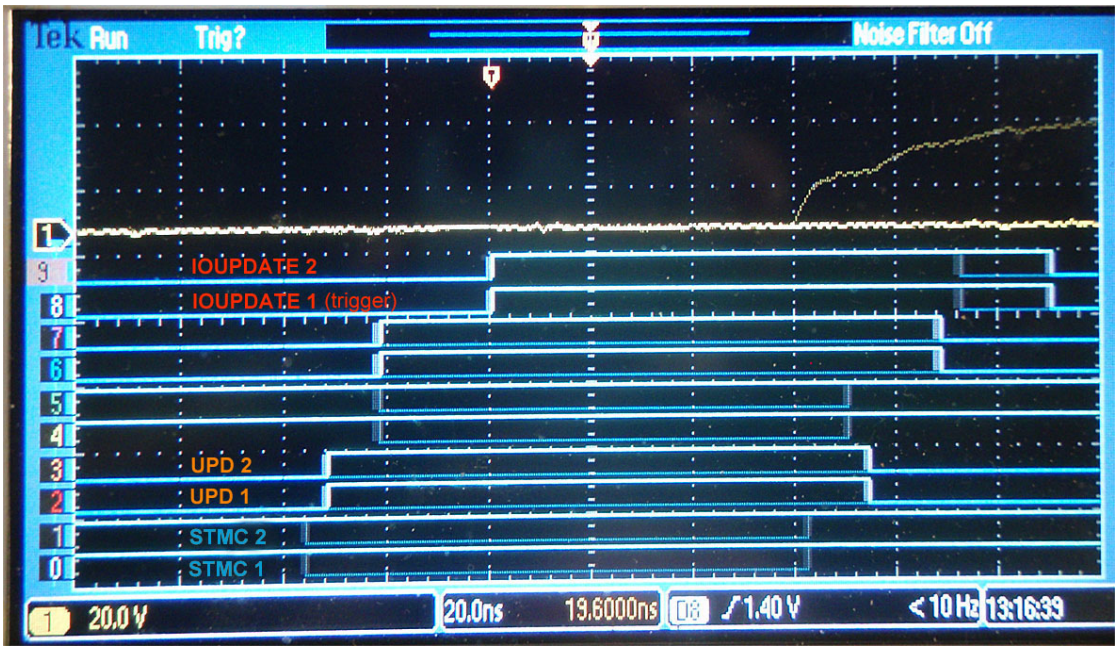



Figure 2: Timing of some internal signals on an exciter board. The signals STMC1|2 (traces 0 and 1) and UPD1|2 (traces 2 and 3) are the incoming signals from the radar controller, measured as early as possible on the exciter mother board. The signals IOUPDATE1|2 (traces 8 and 9) are signals going to the two AD9953 DDS chips on the exciter board, measured as late as possible on the exciter mother board. In the RC program, the STMC1|2 pair is issued at the AT-time 0.4 $\mu$s and the UPD1|2 at the AT-time 1.0 s, while the REP is 2.0 s. Both pairs result in the pair IOUPDATE1|2, of which IOUPDATE1 has been used as the trigger for the logic analyser display. Due to the long exposure time of the photo, and|or afterglow of the screen, both the STMC1|2 (bluish) and UPD1|2 (white) pairs are most conveniently shown in the same picture here, even though in reality only the UPD1|2 pair does occur at this particular 1-second boundary. (The analyser traces 4-7 are the signals 0-3 measured immediately after input drivers on the exciter board. Possibly due to different load on the signal path of the two signals, necessitated by their different polarity, the signals are stretched differently by the drivers. But the trailing edges should not matter here.) The important thing is that there is a difference in time from the UPD1|2 leading edge to the IOUPDATE1|2 leading edge, compared to the corresponding interval from STMC1|2 leading edge to IOUPDATE1|2 leading edge. The difference is about 5 ns. A separate measurement seemed to indicate that both the STMC and the UPD occur "very precisely" (means what?) at its commanded 100 ns boundary. Therefore, it must be the leading edge of the IOUPDATE that vary by about 5 ns with respect to common system clock, depending on whether it is generated by STMC or by UPD.

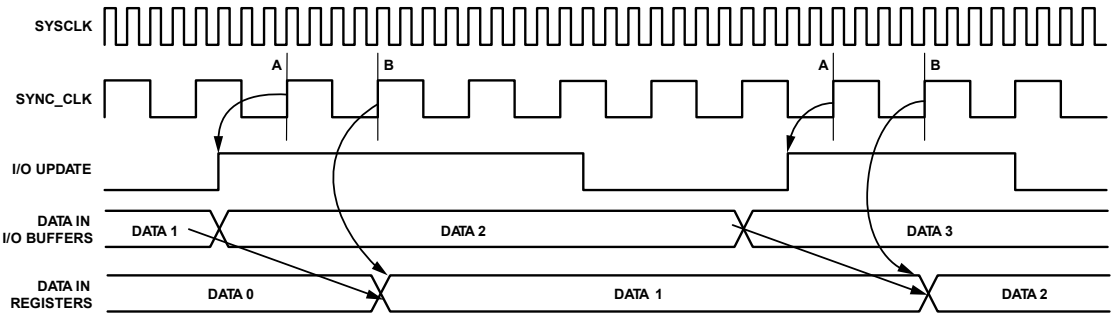
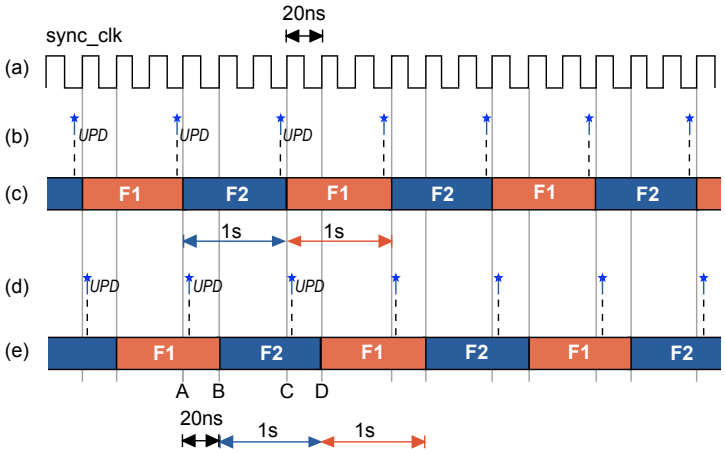
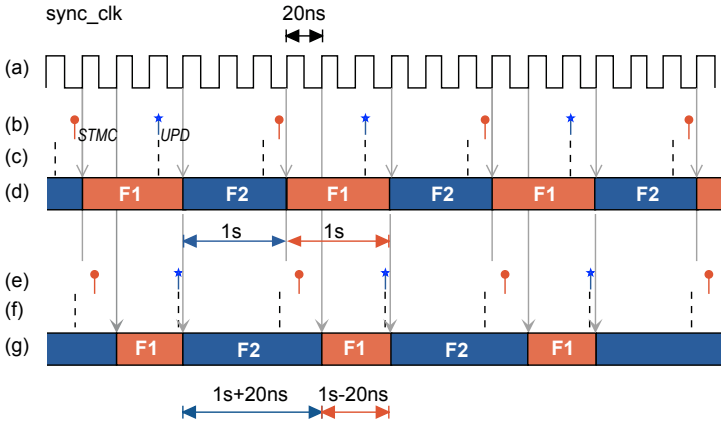



Figure 3: IOSYNC of AD9953. There is, internally on the AD9953 DDS chip, circuitry that aligns the possibly slightly asynchronous input IOUPDATE signals to the 50 MHz SYNC_CLK, and thus, ultimately, to the 200 MHz DDS SYSCLK. In EISCAT application, the SYSCLKs of all the DDS chips are derived from, and phase-locked to, a common 10 MHz reference. All SYSCLKs are initially aligned by the EROS DDS SYNC command to the SYSCLK of the master DDS. Once this initial alignment has been done, the on-chip IOSYNC feature makes it possible to update the synthesized frequencies synchronously across multiple DDSs, without placing too much demands to the precise timing of the IOUPDATE pulses. The IOSYNC allows some jitter in the IOUPDATE on a given DDS, and even allows systematic, but constant, differences in the IOUPDATE timing between separate DDSs, without the DDS output phases starting to drift systematically with respect to each other. But the IOSYNC cannot prevent problems that *may* arise if in the situation pictured, the, say, first IOUPDATE is delayed every now and then, in some systematic fashion, so much that it crosses the time marked 'A' in the diagram. That kind of large, systematic timing variation would not happen in a well-defined phase-locked system on any given DDS, if there would be only a single source for the incoming IOUPDATE signals. But in our case, there were two timing-wise different sources for the IOUPDATE signals for any DDS, and this ultimately allowed the phase-drift problem to occur. The figure is reproduced from the AD9953 data sheet, `http://www.analog.com/static/imported-files/data_sheets/AD9953.pdf`.




(a) Single type of IOUPDATE—no phase drift between DDS $c$ and $e$.




(b) Two types of IOUPDATE—phase drift between DDS $d$ and $g$.

Figure 4: Origin of the DDS phase drift. We assume that an EROS DDS SYNC has been issued so that in effect there is a common SYNC_CLK across all DDSs. As a simplication to Fig. 3, we also assume that the frequency change occurs immediately at the next rising edge of SYNC_CLK after the IOUPDATE's rising edge. In panel (a), there is only one type of IOUPDATE signal, marked here by UPD. The UPD signal may jitter a little, but on the average, it will occur at a constant phase with respect to the 50 MHz SYNC_CLK on any given DDS. Different DDSs, lines $c$ and $e$ on panel (a), can well have their UPD sequences $b$ and $d$ systematically shifted with respect to each other, and this shifts the whole output frequency pattern by a multiple of 20 ns. But as both patterns have the same amounts of $F_1$ and $F_2$, the output phase of both DDSs advances at the *same mean speed* and therefore, there will be no systematic phase drift. What happens is only that during a 20 ns interval such as $AB$ around the frequency change, one DDS accumulates phase faster than the other, but at the next switch back $CD$, this speed difference is the other way round and for the same amount of time, and therefore, there will no net drift. The situation is quite different on panel (b), where there are two timing-wise different types of IOUPDATEs, marked here as STMC and UPD. Small, but systematic, differences between the update sequences $b$ and $e$ may (but need not) place these signals significantly differently with respect of SYNC_CLK on different DDSs. The overall output frequency patterns, $d$ and $g$, can then differ more seriously than by a constant shift. The mean speed of phase accumulation can thus be different on the two DDSs, implying a systematic drift of their relative phase.