Principles of Cascaded Diophantine Frequency Synthesis

Paul Peter Sotiriadis

pps@ieee.org

Abstract—Cascaded Diophantine Frequency Synthesis (CDFS) is a new systematic methodology for developing and programming modular multi-loop frequency synthesizers with high frequency resolution, fast frequency hopping and potentially very low spurs, especially near-in. CDFS results in significantly reduced frequency ranges of the intermediate signals in the frequency mixing stages compared to the predecessor, Diophantine Frequency Synthesis methodology. This simplifies the design and frequency planning for the synthesizer and allows for improved spectral purity of the output signal.

I. INTRODUCTION

Cascaded Diophantine Frequency Synthesis (CDFS) is based on Diophantine equations [1] and shares the same mathematical principles with its predecessor, Diophantine Frequency Synthesis (DFS) methodology, that was introduced in the IEEE Frequency Control Symposium of 2006, [2].

Both methods employ two or more Frequency Synthesis Blocks (FSB), like Phase Locked Loops (PLL), Direct Digital Synthesizers (DDS) or others [3], which are driven by the same reference frequency signal and whose output frequencies are added or subtracted to give the output frequency of the DFS / CDFS synthesizer.

Both methods use only exactly-periodic signals, without employing any dithering, interpolation, pulse removal or other approximately-periodic waveform that may corrupt the near-in spectrum [3], and, they result in high-level architectures whose output fractional-frequency resolution is equal to the product of the FSB’s fractional-frequency resolutions. This allows for the output frequency step to be made (arbitrarily) small while using a relatively high frequency reference for the FSBs.

CDFS has all desirable properties of DFS and in addition it offers the following significant advantage: it minimizes the frequency ranges of all intermediate signals involved in frequency mixing which allows for improved spectral purity and lower design complexity compared to DFS, [4]-[5].

II. NOTATION

The Frequency Synthesis Blocks (FSB)s used to compose a CDFS or DFS synthesizer are shown in abstract form in Figure 1a as frequency multipliers by $\hat{n}/R$.

$$f_{in} \times \frac{\hat{n}}{R} \rightarrow f_{out}$$

(a)

$$f_1 \sum_{\hat{n}} f_1 \pm f_2$$

(b)

$$f_{out} = \frac{\hat{n} + n}{R} f_{in}. \quad (1)$$

An additional, fixed, frequency divider, $Q$, may also be be present in (1), i.e. $f_{out} = \frac{\hat{n} + n}{QR} f_{in}$. Moreover, the range of $n$ can also be expanded to multiples of $R$.

The mixing of two signals at frequencies $f_1$ and $f_2$ is denoted as in Figure 1b where the outcome can be either $f_1 + f_2$ or $f_1 - f_2$ (fixed choice). The context in the paper indicates whether the sum or the difference is considered. The mixing process here includes pre-and post-filtering to remove all unwanted intermodulation products (including $f_1 + f_2$ when the desirable output is $f_1 - f_2$, and vice versa). Mixing of three or more signals has a similar interpretation.

It is instructive to summarize DFS and discuss the frequency properties of the intermediate signals involved in the frequency mixers, first, in order to introduce and motivate CDFS.

---

1Patent Pending

2Almost any frequency synthesizer can be used as a constituent block of DFS or CDFS synthesizers; in general one would prefer simple(r) structures like Integer-N PLLs.

3However, one may choose FSBs that do so.
Moreover, the positive integers $R_1, R_2, \ldots, R_k$ are chosen to be *pairwise relatively prime*. Then DFS guarantees that by programming the values of $n_i$'s using the DFS algorithm [6], $f_{out}$ can take (all) values within the range

$$ [ \bar{f}_{out} - f_{in}, \bar{f}_{out} + f_{in} ] $$

with uniform frequency step equal to

$$ f_{step} = \frac{f_{in}}{R_1 R_2 \cdots R_k} \tag{7} $$

where $f_{out}$ is given by (3). More specifically, $f_{out}$ can take any of the values

$$ f_{out} = \bar{f}_{out} + \frac{n}{R_1 R_2 \cdots R_k} f_{in} \tag{8} $$

where

$$ n = -R_1 R_2 \cdots R_k, \ldots, R_1 R_2 \cdots R_k, \tag{9} $$

by programming the values of $n_i$'s within their ranges,

$$ -R_i \leq n_i \leq R_i \tag{10} $$

$i = 1, 2, \ldots, k$, (using the DFS algorithm [6]).

The central output frequencies of the FSBs, $\bar{f}_i$'s have to be chosen appropriately to result in the desirable value of $f_{out}$ and allow for a spectrally clean output - based on the chosen frequency mixing process. Note that $\bar{f}_{out}$ can be set with the same resolution, (7), as $f_{out}$.

DFS implies that with relatively small values of $k$ and $R_1, R_2, \ldots, R_k$, the frequency step, $f_{step}$, can be made very small, while at the same time, the required frequency steps of the FSB, i.e. $f_{in}/R_i$, $i = 1, 2, \ldots, k$, can be large. The fractional resolution of the DFS scheme is $R_{GI}$ times larger than the geometric mean of the fractional resolutions of the constituent FSBs, where

$$ R_{GI} = (R_1 R_2 \cdots R_k)^{1/k} \tag{11} $$

**IV. MOTIVATING CDFS**

CDFS methodology requires the same frequency ranges and resolutions from FSBs, and results in the same frequency range and resolution of the synthesizer’s output signal with its predecessor DFS [6].

The advantage of CDFS is the significantly reduced (and from certain aspects - minimal) frequency ranges and resolutions from FSBs, and results in the same signal with its predecessor DFS [6].

The following examples clarify the situation and motivate the new approach.
A. DFS Example 1

Consider the DFS scheme in Figure 3 using three FSBs with parameters $R_1 = 4$, $R_2 = 3$ and $R_3 = 5$ (the values of $\bar{n}_i$'s are not important here as they only set $\bar{f}_i$'s, $f_I$ and $f_{out}$).

Using the DFS algorithm, [6], to program the values of $n_i$'s, within their ranges (10), we generate all output frequencies (8) for $n = -R_1 R_2 R_3, \ldots, R_1 R_2 R_3$. The normalized output frequency $(f_{out} - \bar{f}_{out})/f_{in}$ ranges from $-1$ to $+1$, as shown in the last graph of Figure 4. The rest of the graphs show the normalized FSB's frequencies and normalized intermediate frequency $f_I$.

Note that the normalized $f_I$'s range within $-1$ and $+1$ as expected, however, the normalized intermediate frequency $f_I$ ranges from $-1.75$ to $+1$ (fourth graph). Moreover, changing the values of $R_1$, $R_2$ and $R_3$ to 41, 56 and 61 respectively results in the normalized intermediate frequency $f_I$ ranging from $-1.98$ to $+1$.

This indicates that in general, the ranges of the intermediate frequencies $f_I$ in the mixing stages can be significantly larger that $(\bar{f}_{f_{in}} - \bar{f}_{in}, \bar{f}_{f_{in}} + \bar{f}_{in}]$.

The ranges of the intermediate frequencies can increase with the number of FSBs, $k$, in the DFS synthesizer.

B. DFS Example 2

The ranges of the intermediate frequencies can be even larger in schemes with more FSBs, e.g. when the scheme in Figure 5 is programmed using the DFS algorithm, it results in normalized intermediate frequencies $f_{in}$ and $f_{out}$ ranging from $-2.87$ to $1$ and from $-0.78$ to $1.89$ respectively.

In contrast to DFS, the CDFS methodology and algorithm result in all normalized intermediate frequencies ranging from $-1$ to $+1$, i.e. the intermediate frequencies $f_I$ in the mixing are always within $[\bar{f}_{f_{in}} - \bar{f}_{in}, \bar{f}_{f_{in}} + \bar{f}_{in}]$.

V. CDFS ARCHITECTURES & ALGORITHM

The high-level general CDFS scheme is shown in Figure 6. Parameters $R_i$, $n_i$ and $\bar{n}_i$ are fixed and such that $\bar{n}_i > R_i$ and $n_i$ is a variable taking values within the range $-R_i \leq n_i \leq R_i$, for all $i = 1, 2, \ldots, k$. 
As in DFS, all FSBs are driven by the same reference signal $f_{in}$ and their output frequencies are added or subtracted (in any chosen but fixed pattern) to produce $f_{out}$. Again, $\alpha_i \in \{-1, 1\}$, $i = 1, 2, \ldots, k$, are the corresponding frequency-weighting coefficients in the mixers.

The resulting output frequency, $f_{out}$, of the synthesizer is given by (2) and (3) and the intermediate frequencies $f_{i_j}$ involved in the mixing process are given by

$$f_{i_j} = \tilde{f}_{i_j} + \left( \sum_{i=1}^{j} \alpha_i \frac{n_i}{R_i} \right) f_{in}. \quad (11)$$

where the fixed part of it, $\tilde{f}_{i_j}$ is

$$\tilde{f}_{i_j} = \left( \sum_{i=1}^{j} \alpha_i \tilde{n}_i \right) f_{in}. \quad (12)$$

For convenience we define $f_{i_1} \equiv f_1$ as well.

The foundation of CDFS is the following theorem whose proof and discussion can be found in [7].

**Theorem 5.1:** Let $R_1, R_2, \ldots, R_k$ be pairwise relatively prime positive integers (i.e., no pair of them has common divisor other than ±1), and $\alpha_i \in \{-1, 1\}$, $i = 1, 2, \ldots, k$, be a fixed set of mixing weights, then:

For every integer $n$ with $|n| \leq R_1 R_2 \cdots R_k$ we can find integers $n_1, n_2, \ldots, n_k$ with $|n_i| \leq R_i$, for all $i = 1, 2, \ldots, k$, satisfying Diophantine equation

$$\sum_{i=1}^{k} \alpha_i \frac{n_i}{R_i} = \frac{n}{R_1 R_2 \cdots R_k} \quad (13)$$

and inequalities (14) for all $j = 2, 3, \ldots, k$

$$\left| \sum_{i=1}^{j} \alpha_i \frac{n_i}{R_i} \right| \leq 1. \quad (14)$$

Moreover, There exists no constant $\beta < 1$ with the property that for every $n$ with $|n| \leq R_1 R_2 \cdots R_k$ we can find a solution of (13) satisfying $|n_i| \leq \beta R_i$ for all $i = 1, 2, \ldots, k$. \Box

Interpreting the Theorem into frequency synthesis we have that: given the stated conditions on $R_i$’s and $\alpha_i$’s we can synthesize all frequencies given by (8) and (9), by appropriately setting the values of $n_i$’s within their ranges (10), and, at the same time satisfy

$$f_i \in \left[ \tilde{f}_i - f_{in}, \tilde{f}_i + f_{in} \right], \quad (15)$$

for all $i = 1, 2, \ldots, k$, and

$$f_{i_j} \in \left[ \tilde{f}_{i_j} - f_{in}, \tilde{f}_{i_j} + f_{in} \right] \quad (16)$$

for all $j = 1, 2, \ldots, k - 1$.

A. **CDFS Algorithm**

The CDFS algorithm is summarized below (details can be found in [7]) and uses the parameterized function $f$ defined as follows:

Let $A, B$ be two relatively prime positive integers; given two integers, $x_1$ and $x_2$ we define $z_1 = x_1 \mod A$, $z_2 = x_2 \mod B$, and, $\mu = x_1 \ \text{div} \ A + x_2 \ \text{div} \ B$. We set

$$f_{(A,B)}(x_1, x_2) = \begin{cases} \frac{(z_1 - A, z_2 - B)}{\mu} & \text{if } \mu = -2 \\ \frac{(z_1 - A, z_2)}{\mu} & \text{if } \mu = -1 \\ (z_1, z_2) & \text{otherwise} \end{cases} \quad (17)$$

**CDFS Algorithm**

- **STEP 0:** If $n = \pm R_1 R_2 \cdots R_k$ then set $n_1 = \pm R_1$, respectively, set $n_i = 0$ for $i = 2, \ldots, k$ and STOP; otherwise proceed to **STEP 1**.
- **STEP 1:** For $i = 1, 2, \ldots, k - 1$ derive, using the Euclidean algorithm, and store a solution $(z_i, w_{i+1})$ of the Diophantine equation

$$\frac{z_i}{R_1 R_2 \cdots R_i} + \frac{w_{i+1}}{R_{i+1}} = \frac{1}{R_1 R_2 \cdots R_{i+1}} \quad (18)$$

- **STEP 2:** Set $x_k = n$ and derive sequentially the $k-1$ vectors $(x_{k-i}, n_{k-i+1})$, $i = 1, 2, \ldots, k - 1$, using

$$f \left( \prod_{i=1}^{k-i} R_i, R_{k-i+1} \right) (x_{k-i+1} z_{k-i}, x_{k-i+1} w_{k-i+1}) \quad (19)$$

- **STEP 3:** Set $n_1 = x_1$. 

747
Vector \((n_1, n_2, \ldots, n_k)\) is a solution of (13), as specified in Theorem 5.1 (and satisfies (14)) when \(\alpha_i = 1\) for all \(i = 1, 2, \ldots, k\). For the general case one should replace \((n_1, n_2, \ldots, n_k)\) with \((\alpha_1 n_1, \alpha_2 n_2, \ldots, \alpha_k n_k)\).

B. Using the CDFS Algorithm to program the (C)DFS synthesizer in Figure 3

The CDFS algorithm is applied to the 3-FSB scheme in Figure 3 where \(k = 3, R_1 = 4, R_2 = 3, R_3 = 5\) and \(\alpha_1 = \alpha_2 = \alpha_3 = 1\).

Based on the previous Section, the CDFS algorithm provides solutions of the Diophantine equation

\[
\frac{n_1}{4} + \frac{n_2}{3} + \frac{n_3}{5} = \frac{n}{4 \cdot 3 \cdot 5}
\]

(20)

that satisfy the desirable inequalities (10) and (14) - translated here as

\[-4 \leq n_1 \leq 4\]

\[-3 \leq n_2 \leq 3\]

\[-5 \leq n_3 \leq 5\]

respectively - for all integers \(n\) such that \(|n| \leq 4 \cdot 3 \cdot 5\).

The steps of the CDFS algorithm are:

- **STEP 0:** If \(n = \pm 4 \cdot 3 \cdot 5 = \pm 60\) then set \(n_i = \pm 4\), respectively, set \(n_i = 0\) for \(i = 2, 3\) and STOP; otherwise proceed to **STEP 1**.

- **STEP 1:** Derive (some) solutions \((z_1, w_2)\) and \((z_2, w_3)\) of the following Diophantine equations using the Euclidean algorithm.

\[
\frac{z_1}{4} + \frac{w_2}{3} = \frac{1}{4 \cdot 3},
\]

\[
\frac{z_2}{4 \cdot 3} + \frac{w_3}{5} = \frac{1}{4 \cdot 3 \cdot 5}
\]

Using the "gcd" function in MATLAB we get \((z_1, w_2) = (-1, 1)\) and \((z_2, w_3) = (5, -2)\).

- **STEP 2:** Set \(x_3 = n\) and derive sequentially vectors \((x_2, n_3)\) and \((x_1, n_2)\) using "two parameterized versions" of function \(f\).

\[
(x_2, n_3) = f(4, 3, 5)(x_3 z_2, x_3 w_3)
\]

\[
(x_1, n_2) = f(4, 3)(x_2 z_1, x_2 w_2)
\]

- **STEP 3:** Set \(n_1 = x_1\).

The **CDFS algorithm** is used to program the values of \(n_i\)'s, within their ranges (10) and generate all output frequencies (8) for \(n = -4 \cdot 3 \cdot 5, \ldots, 4 \cdot 3 \cdot 5\).

As expected the normalized output frequency \((f_{\text{out}} - f_{\text{in}})/(f_{\text{in}})\) ranges from \(-1\) to \(+1\), as shown in the last graph of Figure 7. The rest of the graphs show the normalized FSB and intermediate \(f_i\) frequencies.

In contrast to DFS programming resulting in Figure 4, here the normalized intermediate frequency \(f_i\) (shown in the fourth graph of Figure 7) ranges from \(-1\) to \(+1\) as expected.

VI. CONCLUSIONS

The Cascaded Diophantine Frequency Synthesis (CDFS) has been presented and compared to its predecessor, Diophantine Frequency Synthesis methodology (DFS).

CDFS leads to fine frequency-step, fast frequency-hopping frequency synthesis architectures with potentially very low spurs, especially in the vicinity of the carrier.

CDFS results in significantly reduced frequency ranges of the intermediate signals in the frequency mixing stages compared to (DFS) simplifying the design of the mixers and offering improved spectral purity of the output signal.
REFERENCES