Efficient C and D sums calculation and decimation for least square estimation of phase, frequency and PDEV

Magnus Danielson [1] proposed a very useful technique which can be used in optimizing data processing of the Ω-frequency counters as well as PVAR calculations. Using that technique timestamps can be processed by small blocks and processing results can be combined to get the frequency or PVAR for the longer measurements time. But there is one nuance which can introduce additional difficulties or compromise the performance of the timestamps preprocessing.

Let's look at the C and D sums formulas:

$$C = \sum_{n=0}^{N-1} x_n \qquad (1)$$

$$D = \sum_{n=0}^{N-1} n x_n \qquad (2)$$

If the reference clock is 400MHz the phase x_n will grow by 4e8 each second. It can be easily shown that D sum (for the block size of 65536 timestamps, $n \in \{0,1,..65535\}$) will not fit in 64bit integer for the measurement time greater than 20s (but it is common case if we are going to calculate PDEV). This results in block calculation performance penalty (because of the need to use floating point or more than 64bits integer math), and, possibly, loss of precision if the floating point math is used.

So, let's modify decimation rule and block processing routines to fix this nuance. The main idea is to calculate C and D sums for each new block restarting the phase from 0. Of cause the starting phase of the current block should be tracked along with the current C and D sums, so the proper decimation can be done. An interesting side effect of such implementation is simplified phase unwrapping (if events frequency is high enough the phase counter will never overflow collecting timestamps for the one block).

The decimation rules in the original paper are:

$$C_{12} = C_1 + C_2 \qquad (3)$$

$$D_{12} = D_1 + N_1 C_2 + D_2 \qquad (4)$$

Using the (1) we can rewrite (3) as:

$$C_{12} = C_1 + \sum_{n=0}^{N_2-1} x_{N_1+n} = C_1 + \sum_{n=0}^{N_2-1} x_{N_1} + \sum_{n=0}^{N_2-1} x_n^0 = C_1 + N_2 x_{N_1} + C_{02} \qquad (5)$$

$$C_{02} = \sum_{n=0}^{N_2-1} x_n^0 \qquad (6)$$

where x_{N_1} is the starting phase of the current block and x_n^0 are current block timestamps with the zero starting phase. Using the same technique we can rewrite (4) as:

$$D_2 = \sum_{n=0}^{N_2-1} n x_{N_1+n} = \sum_{n=0}^{N_2-1} n(x_{N_1} + x_n^0) = x_{N_1} \sum_{n=0}^{N_2-1} n + \sum_{n=0}^{N_2-1} n x_n^0 \qquad (7)$$

$$D_{12} = D_1 + N_1 C_2 + x_{N_1} \sum_{n=0}^{N_2-1} n + \sum_{n=0}^{N_2-1} n x_n^0 = D_1 + N_1 C_2 + x_{N_1} E + D_{02} \qquad (8)$$

$$E = \sum_{n=0}^{N_2-1} n \qquad (9)$$

$$D_{02} = \sum_{n=0}^{N_2-1} n x_n^0 \qquad (10)$$

The sum in (9) is a constant for a constant N_2, it can be precalculated to improve the performance.

The modified decimation rules are:

$$E = \sum_{n=0}^{N_2-1} n \qquad (11)$$

$$C_{02} = \sum_{n=0}^{N_2-1} x_n^0 \qquad (12)$$

$$D_{02} = \sum_{n=0}^{N_2-1} n x_n^0 \qquad (13)$$

$$C_2 = N_2 x_{N_1} + C_{02} \qquad (14)$$

$$C_{12} = C_1 + C_2 \qquad (15)$$

$$D_{12} = D_1 + N_1 C_2 + x_{N_1} E + D_{02} \qquad (16)$$

$$N_{12} = N_1 + N_2 \qquad (17)$$

$$x_{N_{12}} = x_{N_1} + x_{N_2}^0 \qquad (18)$$

They might look complicated, but (11), (12) and (13) can be coded very efficiently using standard integer math, (11) can even be precalculated (it is constant for the given N_2) and the other calculations should be done only once per block.

So, now the block processing routine should calculate the C_{02} and D_{02} sums only. Both sums fit in 64bits integers, the phase inside one block can be represented by the 32bits integer. It greatly improves the performance and simplifies block processing code (the experimental code was able to process up to 43MSPS on general purpose 32bit ARM MCU).

Oleg Skydan
UR3IQO

[1] "Least square estimation of phase, frequency and PDEV," Magnus Danielson, Francois Vernotte, Enrico Rubiola