Markus Appel, Felix Wermke, Frank Winkler, Beate Meffert

Frequency Synchronization for Wireless Networks using Field Programmable Gate Arrays

Final version published as:

Markus Appel, Felix Wermke, Frank Winkler, Beate Meffert: Frequency Synchronization for Wireless Networks using Field Programmable Gate Arrays. In: 2016 IEEE International Conference on Electronics, Circuits and Systems (ICECS). DOI: 10.1109/ICECS.2016.7841238

Frequency Synchronization for Wireless Networks using Field Programmable Gate Arrays

Markus Appel, Felix Wermke, Frank Winkler, and Beate Meffert
Humboldt-Universität zu Berlin
Email: { mappel | wermkefx | fwinkler | meffert } @informatik.hu-berlin.de

***Abstract*—High-precision sensor networks and localization systems require precise time and frequency synchronization. In this paper, we present a novel high-precision frequency synchronization approach for wireless network devices. It adapts the local oscillator frequency of a receiver to the frequency of a transmitter and can be integrated into existing wireless communication systems. The measurement of frequency differences as well as the frequency adjustment is realized in Field Programmable Gate Arrays (FPGAs). Using a 60 GHz wireless experimental setup, the receiver clock is aligned to the transmitter clock with a precision of 37 picoseconds.**

I. Introduction

A precise and accurate time and frequency synchronization between remote devices is essential for many applications including wireless sensor networks and modern communication or localization systems. These applications need time synchronization accuracies ranging from microseconds down to nanoseconds.

Oscillators provide the time bases in almost every electronic system. However, each oscillator has an unintended and generally arbitrary offset from its nominal frequency. Additionally, they have a frequency drift – an undesired change in frequency with time, which can be caused by component aging and environmental changes. Frequency drift can occur in either direction and is not necessarily linear [1]. Thus, in systems requiring a highly precise and accurate synchronization it is fundamental to deal with frequency synchronization which is also termed syntonization. To distinguish the terms, we follow the definition of the National Institute of Standards and Technology [1]:

> *Synchronization* is the process of setting two or more clocks to the same time.
>
> *Syntonization* is the process of setting two or more oscillators to the same frequency.
>
> *Precision* has several meanings in time and frequency metrology. In the context of this paper, it refers to the degree of mutual agreement among a series of individual measurements, values, or results. Precision is analogous to standard deviation.

This paper presents a novel syntonization approach for wireless network devices using Field Programmable Gate Arrays (FPGAs). FPGAs are integrated circuits that can be reprogrammed to a desired application. The syntonization is realized in the receiver only. Therefore, an unidirectional connection is sufficient.

The main contributions of this work are:

1) We present a high-precision syntonization approach for wireless networks.
2) Our solution extends common wireless receiver architectures in order to combine data transmission and syntonization.
3) We use FPGA-internal structures for a precise receiver syntonization.
4) Our solution is not limited to a specific frequency range.

The remainder of this paper is structured as follows: the next section presents related work in this field of research. Our method for syntonization is explained in Section III. Section IV describes the experimental setup and the achieved results. This paper is concluded in Section V.

II. Related work

The state-of-the-art solutions in high-precision wireless networks use the Global Position System (GPS) for syntonization [2], [3], [4] - optionally in conjunction with Oven Controlled Crystal Oscillators or atomic clocks. In this section, we limit the related work to systems realized without GPS, because our approach does not use it.

The White Rabbit (WR) project [5] is one solution in the field of wired networks. The syntonization is done by Synchronous Ethernet (SyncE) and synchronization is realized by the Precision Time Protocol (PTP). The guaranteed sub-nanoseconds accuracy and picoseconds precision among a few thousand nodes is an excellent benchmark for all systems combining syntonization and synchronization. However, due to the fact that WR uses SyncE for syntonization, the solution is not transferable to wireless networks. A wireless system with a comparable performance is not yet available.

In [6], syntonization is used to improve the synchronization in wireless sensor networks. The synchronization error is in the order of microseconds. However, the author's aim was to extend the synchronization intervals and decrease the offset error between adjacent synchronization points. Furthermore, a decrease of power consumption was achieved.

A precise solution was developed by Locata [7]. It is a localization system that works with radio stations independent of GPS. In LocataNet, all slaves are syntonized to a master using the so called TimeLoc algorithm. Over a distance of 15 m, they measured the frequency stability with a precision of 51 ps. LocataNet uses a special hardware that is not made for data transfers.

Compared to our preliminary work [8], we present a vastly improved method aligning the clock of a receiver to the clock of a transmitter. Our new solution achieves syntonization results comparable to LocataNet.

III. Method for Syntonization

The standard wireless communication infrastructure consists of an analog radio frequency (RF) front end and a digital baseband system. In order to be compatible with wireless communication systems, we extend this structure by two components as shown in Fig. 1:

1) A Frequency Analyzer that determines the frequency difference Δf between Clock_T and Clock_R.
2) A Clock Synthesizer that adjusts the frequency of Clock_R with the result that $\Delta f = 0$.

Clock_T and Clock_R are used as reference clocks for the RF front ends. With this arrangement, the receiver clock Clock_R can be tracked to the transmitter clock Clock_T.

The following subsections describe the process of measuring frequency differences, a solution for frequency adjustment using FPGAs, and our algorithm for syntonization.

A. Measurement of Frequency Differences

The frequency difference Δf between Clock_R and Clock_T can be described by:

$$\Delta f = f_{\text{Clock}_\text{R}} - f_{\text{Clock}_\text{T}} \tag{1}$$

We determine Δf with a coarse and a fine measurement.

1) Coarse measurement: The transmitter sends unique network packets every a clock cycles of Clock_T. The receiver detects them and counts b clock cycles between the occurrence of two adjacent packets using Clock_R. This can be expressed as $\frac{a}{f_{\text{Clock}_\text{T}}} = \frac{b}{f_{\text{Clock}_\text{R}}}$. Using Eq. (1), Δf_coarse can be calculated with:

$$\Delta f_\text{coarse} = \left(1 - \frac{a}{b}\right) \cdot f_{\text{Clock}_\text{R}} \tag{2}$$

The resolution of Δf_coarse and the residual error depend on the observation period. A fine resolution needs long-term measurements. Since it is only the mean that is calculated over the observation period, short-term frequency drifts are not detectable with this method.

2) Fine measurement: The approach is based on the fact that minimal frequency differences of Clock_T and Clock_R cause phase differences of the carrier waves. This directly affects the phasing of the baseband data as well. It can be seen in the result of the In-Phase Quadrature Demodulation (IQ Demodulation) which is used in almost all modern digital radio transceivers. The I and Q signals, typically represented as complex pointers, rotate depending on the phase relation of the carrier frequencies f_c of the transmitter and the receiver. Wireless receivers normally compensate this relation in the digital baseband processing. Here, the phase relation is used to calculate the frequency difference Δf_fine. The phase change $\Delta\phi$ of well-known sequences (e.g., preambles of transmitted packets) is observed by the receiver during a time period Δt which is determined by counting d clock cycles of Clock_R ($\Delta t = \frac{d}{f_{\text{Clock}_\text{R}}}$). Δf_fine can be determined by:

$$\Delta f_\text{fine} = \left(\frac{\Delta\phi}{c \cdot 360° \cdot d}\right) \cdot f_{\text{Clock}_\text{R}}, \qquad c = \frac{f_\text{c}}{f_{\text{Clock}_\text{R}}} \tag{3}$$

The sensitivity of this method makes frequency drifts detectable after a short period of time. To detect the drift properly, the difference of the phases has to be less than $180°$. Otherwise, the direction of rotation cannot be determined.

B. Frequency Adjustment using FPGAs

There are different approaches for frequency adjustment. Commonly, Voltage-controlled Crystal Oscillators (VCXO) are used. Modern FPGAs from Xilinx including Virtex-6 and 7 Series FPGAs are suitable as well. They contain Clock Management Tiles [9] that include a Mixed-Mode Clock Manager (MMCM) and a Phase-locked Loop (PLL). Both components serve as clock synthesizers for frequencies up to 800 MHz. In this context, the MMCM with its phase-shift capability is the most interesting feature. A periodic increase or decrease of the phase results in a slightly modified output frequency. The step size of a phase shift depends only on the frequency f_VCO of the internal Voltage-controlled oscillator (VCO) and is defined by:

$$PS_\text{step} = \frac{1}{56 \cdot f_\text{VCO}} \tag{4}$$

After 12 cycles of the programming clock $f_\text{ps\_clk}$ a phase shift is completed. A hardware module utilizes the periodic shift operations. Eq. (5) is used to calculate the time interval in which a phase shift is performed periodically. The parameters k and l are variable during runtime, m is constant.

$$T_\text{ps} = \left(k + \frac{l}{2^m}\right) \cdot \frac{1}{f_\text{ps\_clk}} \quad \begin{cases} k \in \mathbb{N}, & k \geq 12 \\ l \in \mathbb{N}_0, & l < 2^m \\ m \in \mathbb{N}_0 \end{cases} \tag{5}$$

Finally, the tuning frequency Δf_tune of a MMCM is determined by:

$$f_\text{tune} = (-1)^h \cdot \frac{PS_\text{step}}{T_\text{ps}} \cdot f_\text{nom} \begin{cases} h = 0 : \text{phase increment} \\ h = 1 : \text{phase decrement} \end{cases} \tag{6}$$

with f_nom being the nominal output frequency of MMCM.

The phase noise of a MMCM was investigated using a ZedBoard including Zynq FPGA [10]. A schematic drawing of the setup and the results of the phase noise measurements are shown in Fig. 2. In comparison to VCXOs, the phase jitter generated by MMCMs is high, particularly in the range from 10 kHz to 100 MHz. Therefore, the usage of an additional jitter cleaner [11] is reasonable in applications that require a very low phase noise.

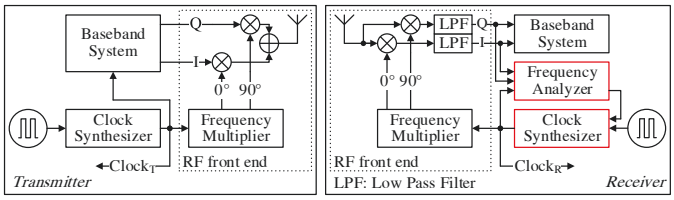

Fig. 1. Transmitter and receiver architectures for syntonization.

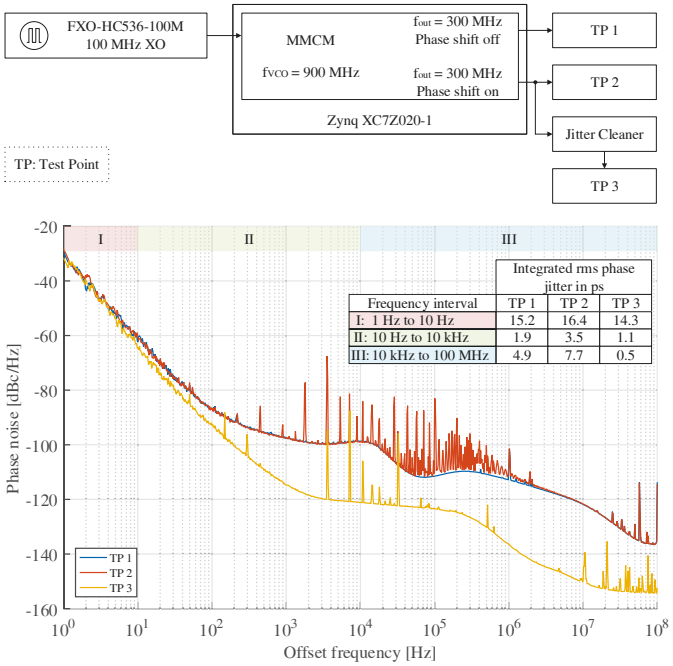

Fig. 2. Setup and measured phase noise of MMCM using a signal source analyzer [12]. The integrated phase jitter was calculated based on [13].

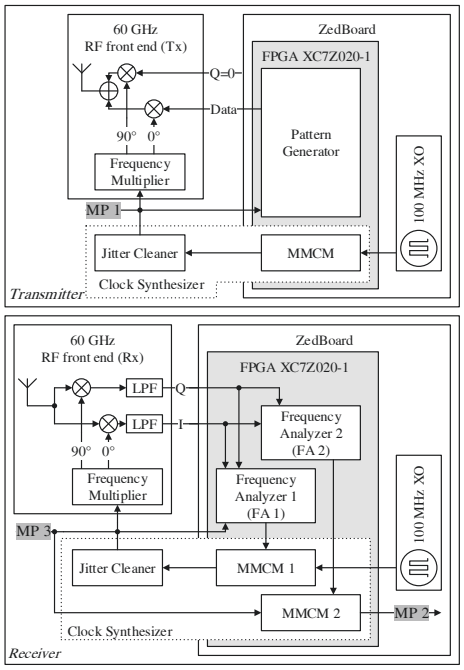

Fig. 3. Simplified schematic of the 60 GHz experimental setup including the measuring points MP 1, MP 2, and MP 3.

C. Algorithm

Our algorithm in the receiver consists of four steps:

1) Measurement of Δf_{coarse} using Clock_R
2) Frequency adjustment of Clock_R using Δf_{coarse}
3) Measurement of Δf_{fine} using Clock_R
4) Frequency adjustment of Clock_R using Δf_{fine} and back to 3)

The first two steps are used to reduce a possibly large frequency offset. The remaining error and the frequency drift are periodically compensated by the last two steps of the algorithm.

The parameters for realizing the frequency adjustment can be determined as follows: Clock_R is generated by a MMCM. The frequency $f_{\text{Clock}_\text{R}}$ used in Eq. (2) and Eq. (3) can be described by:

$$f_{\text{Clock}_\text{R}} = f_{\text{nom}} + f_{\text{tune}} \tag{7}$$

The generalized form of Eq. (2) and Eq. (3) is:

$$\Delta f_i = z \cdot f_{\text{Clock}_\text{R}} \tag{8}$$

After measuring Δf_i, a new frequency $(f_{\text{Clock}_\text{R}})^*$ is determined by:

$$(f_{\text{Clock}_\text{R}})^* = f_{\text{Clock}_\text{R}} - \Delta f_i \tag{9}$$

Using Eq. (9) in combination with the Eq. (5), Eq. (6), Eq. (7), and Eq. (8), the new parameters h^*, k^*, and l^* can be calculated with:

$$\frac{(-1)^{(h)^*}}{(k)^* + \frac{(l)^*}{2^m}} = \frac{(-1)^h \cdot (1-z)}{k + \frac{l}{2^m}} - \frac{z}{PS_{\text{step}} \cdot f_{\text{ps\_clk}}} \tag{10}$$

PS_{step} and $f_{\text{ps\_clk}}$ are constant. h^*, k^*, and l^* depend on the previous parameters h, k, and l and on the measurement z.

IV. Experimental Setup and Results

A schematic drawing of our 60 GHz experimental setup is presented in Fig. 3. The main hardware components were:

- ZedBoard [10] including a Xilinx Zynq FPGA and a 100 MHz crystal oscillator [14]
- Jitter cleaner [11]
- 60 GHz transceiver [15]

The used oscillators had a nominal frequency of 100 MHz and a frequency stability of ± 25 ppm [14]. Since they were sensitive to environmental conditions, especially to temperature changes, they were sufficient to demonstrate the benefit of the proposed method. The parameters of our experimental setup were determined by the properties of the crystals.

Clock_T and Clock_R were created by MMCMs using the external crystal oscillators. They had a nominal frequency of 300 MHz. Both jitter reduced clocks were used in the radio modules to generate the carrier frequency f_c of 58.8 GHz. To achieve a difference of less than 0.05 ppb for two adjacent programmable frequencies in the relevant range of ± 50 ppm, we chose f_{VCO} = 900 MHz, $f_{\text{ps\_clk}}$ = 300 MHz, and $m = 10$.

The parameter c used in Eq. (3) was approximately 200. This means, a 1° phase change of Clock_T or Clock_R resulted in 200° phase change in the carrier band. Therefore, our experimental setup was sensitive to phase noise and the usage of jitter cleaners was indispensable.

A known sequence was transmitted every second. Frequency Analyzer 1 (FA 1) used these sequences to measure Δf_{coarse} and to adjust the frequency of Clock_R.

When the coarse frequency adjustment was completed, our measurements have shown that the duration of a 90° phase change in the carrier band was at least 200 µs. In this time

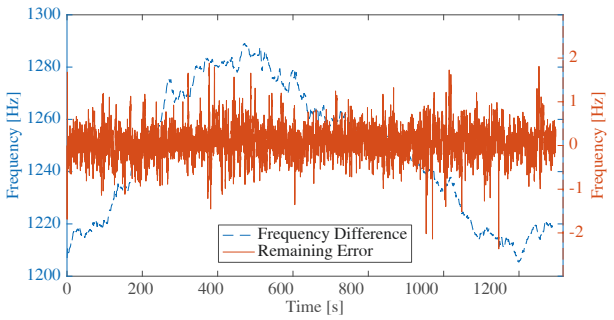

Fig. 4. Frequency difference compensated by MMCM 1 and remaining frequency error at MP 3. Please note the different scaling of the y-axes.

interval, a new phase was estimated. In order to calculate the frequency difference, several phase changes had to be detected to eliminate errors. Using a linear regression, the angular velocity was approximated and the output frequency of MMCM 1 was adjusted by FA 1. This solution achieved a precision of 3 ns measured at points MP 1 and MP 3 in Fig. 3.

In order to achieve better results, the jitter cleaner output was used as the reference input for MMCM 2. Running MMCM 2 with $f_{\mathrm{VCO}} = 1050\,\mathrm{MHz}$, one phase shift was equivalent to one period of the carrier wave. Therefore, MMCM 2 was able to correct the phase offset of one clock cycle of the carrier wave immediately after its detection by FA 2. Consequently, MMCM 1 corrected the major part of the frequency drift, whereas MMCM 2 regulated the phase changes immediately.

In order to get a better impression of the internal functionality, the frequency difference compensated by MMCM 1 and the remaining error at MP 3 were recorded (Fig. 4). The dashed line shows the process of frequency drift between the two clocks. It reveals a frequency offset of about 1200 Hz. The solid line shows that frequent adjustments of MMCM 2 were necessary to compensate the remaining error. Up to 500 phase shifts per second were required to keep the clocks aligned.

The syntonization success was measured with a sampling oscilloscope (Fig. 5). Under laboratory conditions, the precision σ was 36.63 ps. 68.9 % of the measurements fell into the range of $\pm\sigma$ and 100 % into the range of $\pm 3\sigma$.

V. Conclusion

In this paper, we have shown that high-precision syntonization of wireless network devices can be realized with FPGAs. The frequency difference can be determined in the receiver by observing the phase change of well-known sequences. Using MMCMs, the local clock of a receiver can be adapted to the clock of a transmitter with a precision of 37 ps. The presented approach is suitable for an integration into wireless data transmission systems.

Acknowledgments

This work was partially supported by the European Regional Development Fund (ERDF) under Grant Reference Number #10159424.

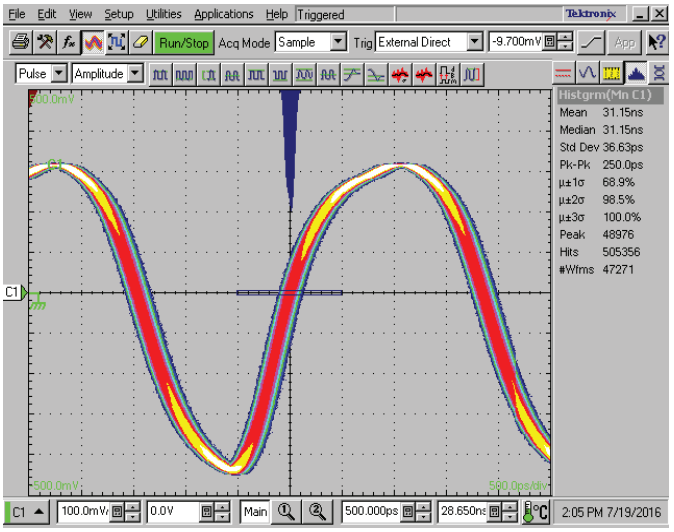

Fig. 5. Phase alignment measurement of MP 1 and MP 2 using a sampling oscilloscope [16]. MP 1 was used as trigger. The 300 MHz signal generated at MP 2 is shown as infinite persistence drawing.

References

[1] (accessed June 30, 2016) Time and Frequency from A to Z. The National Institute of Standards and Technology (NIST). [Online]. Available: http://www.nist.gov/pml/div688/grp40/enc-index.cfm

[2] K.-Y. Tu, F.-R. Chang, C.-S. Liao, and L.-S. Wang, "Frequency Syntonization Using GPS Carrier Phase Measurements," *IEEE Transactions on Instrumentation and Measurement*, vol. 50, no. 3, pp. 833–838, Jun. 2001.

[3] K. Y. Tu and C. S. Liao, "Application of ANFIS for Frequency Syntonization Using GPS Carrier-Phase Measurements," in *2007 IEEE International Frequency Control Symposium Joint with the 21st European Frequency and Time Forum*, May 2007, pp. 933–936.

[4] C. L. Cheng, F. R. Chang, L. S. Wang, and K. Y. Tu, "Highly-Accurate Real-Time Syntonization Using A GPS Single-Frequency Receiver With An Intelligent Atmosphere Forecasting Model," in *Frequency Control Symposium and Exposition, 2004. Proceedings of the 2004 IEEE International*, Aug. 2004, pp. 527–535.

[5] M. LipiÅĎski, T. WÅĆostowski, J. Serrano, and P. Alvarez, "White Rabbit: a PTP Application for Robust Sub-nanosecond Synchronization," in *Precision Clock Synchronization for Measurement Control and Communication (ISPCS), 2011 International IEEE Symposium on*, Sep. 2011, pp. 25–30.

[6] J. Reis and N. B. Carvalho, "Synchronization and Syntonization of Wireless Sensor Networks," in *Wireless Sensors and Sensor Networks (WiSNet), 2013 IEEE Topical Conference on*, Jan. 2013, pp. 151–153.

[7] E. Powers and A. Colina, "Wide Area Wireless Network Synchronization Using Locata," Locata, Tech. Rep., 2015.

[8] M. Appel, F. Wermke, and F. Winkler, "Präzise Synchronisation von Funkstationen im 60-GHz-Band," in *DASS 2016 - Tagungsband*, Brandenburgische Technische Universtät. Fraunhofer Verlag, Mar. 2016, pp. 31–34.

[9] Xilinx, *7 Series FPGAs Clocking Resources*, Jun. 2015, UG472 (v1.11.2).

[10] Avnet, *ZedBoard Rev D.2*, Dec. 2015.

[11] SiliconLaboratories, *ITU-T G.8262 Synchronous Ethernet Jitter-Attenuating Clock Multiplier*, Jul. 2013.

[12] Agilent, *Agilent E5052B Signal Source Analyzer 10 MHz to 7 GHz*, Nov. 2013.

[13] W. Kester, "Converting Oscillator Phase Noise to Time Jitter," Analog Devices, Tutorial, Aug. 2008, MT-008.

[14] F. Electronics, *Model Description: FXO-HC536-100M*, Feb. 2012.

[15] Hittite, *60 GHz Antenna-in-Package Transceiver Evaluation Kit*, Apr. 2013.

[16] Tektronix, *Digital Serial Analyzer Sampling Oscilloscope - DSA8200 Data Sheet*, Aug. 2011.