# APS007 APPLICATION NOTE

# WIRED SYNCHRONIZATION OF ANCHOR NODES IN A TDOA REAL TIME LOCATION SYSTEM

## How to synchronize anchor nodes using a wired scheme in an RTLS based on DecaWave's DW1000 IC

**Version 1.10**

**This document is subject to change without notice**

## TABLE OF CONTENTS



## LIST OF TABLES

## LIST OF FIGURES

# 1 INTRODUCTION

This is one in a series of notes on the application of DecaWave's DW1000 technology.

This note assumes the reader is familiar with the concepts and principles behind RTLS in general and Time Difference of Arrival (TDOA) RTLS in particular – for more information on the principles of RTLS see the DecaWave website www.decawave.com.

This note considers how to synchronize anchor nodes in a Real Time Location System (RTLS), implemented using DecaWave's DW1000 IC, by using wired synchronization techniques.

The topics covered include:

- Wired Sync RTLS architecture
- Hardware features for Wired Sync RTLS with the DW1000
- Software features for wired Sync RTLS with the DW1000

# 2 WIRED SYNC ARCHITECTURE

## 2.1 Introduction

An RTLS consists of tagged mobile items that are located with reference to fixed anchor nodes. One of the most power efficient techniques to do this in terms of the power consumption of the tags is where the tag sends a periodic "blink" message that is received at multiple anchors and the difference between the message's arrival time at different anchors is used to determine the location of the tag.

This technique is called Time Difference of Arrival (TDOA). Typically the arrival times of blink messages at each anchor are reported to a central location engine (CLE) which computes the time differences for pairs of anchors and then solves the TDOA data in a mathematical process called multilateration.

For the time difference to make sense and allow the tag to be located the anchors receiving the message must have the same concept of time and their timebases must be synchronised.

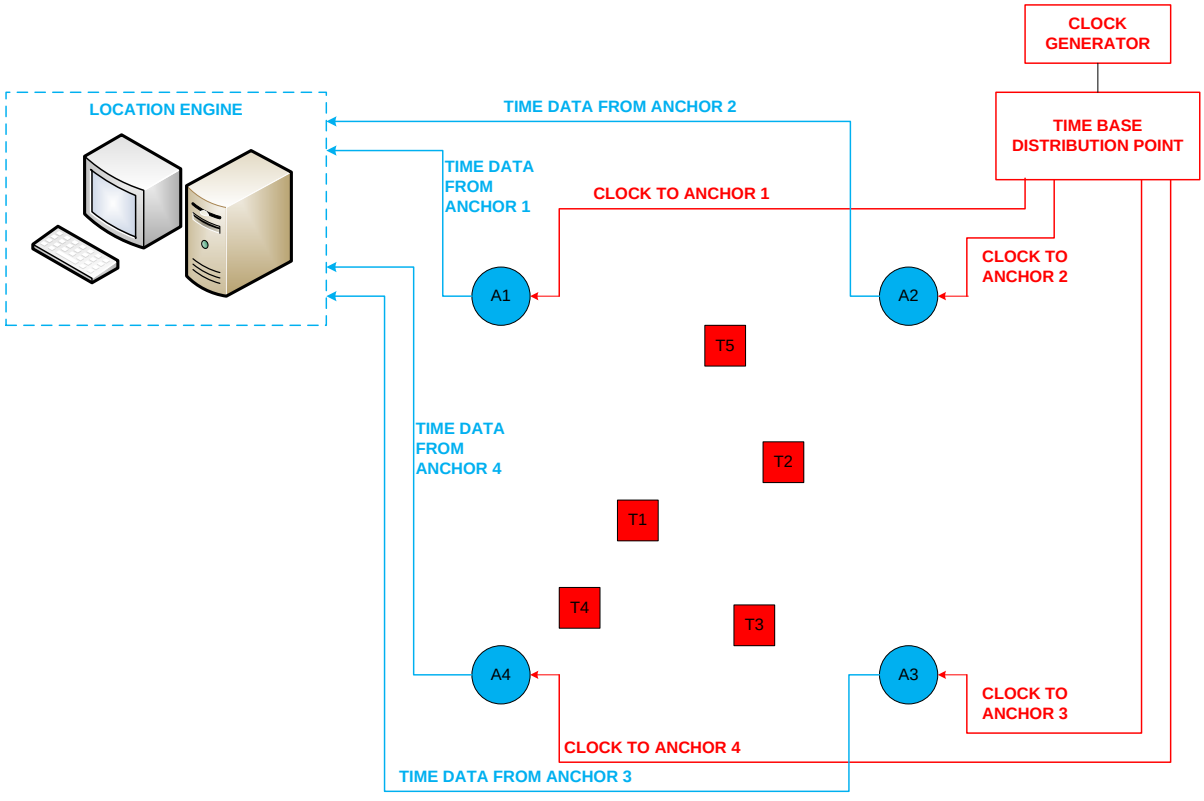


**Figure 1: Basic concept of a wired synchronization scheme**

If each anchor were to use its own free-running clock source then the anchor clocks would drift out of synchronization very quickly to the point where the system would fail. Even using temperature controlled crystal oscillators (TCXOs) with a very tight frequency tolerance (1 ppm) the time-bases in different anchors will rapidly drift apart; for example a 1 ppm error results in a 1 ns error over a 1 ms interval – this is well outside the limit required for accurate RTLS.

Therefore, synchronization of the anchor nodes is essential. There are a number of methods of achieving the required synchronization; this application note concerns itself with one of those methods known as “wired synchronization”.

## *2.2 Architecture options*

### 2.2.1 Introduction

In a wired synchronisation scheme using the DW1000 the minimum requirement is that a common 38.4 MHz clock is distributed to all anchor nodes and used to drive each anchor node’s on-board DW1000 IC. Inside the DW1000 this 38.4 MHz clock is multiplied by 13 to 499.2 MHz and then divided by 4 to give the 124.8 MHz system clock which is used for counting time in the DW1000’s (system clock) counter.

Distributing the 38.4 MHz clock to all anchors means that time is counted at the same rate for all the anchors and, when the IC receiver timestamps a tag blink arrival, the timestamps at each anchor will have a consistent meaning as long as any fixed offsets between anchors are determined.

Assuming this basic requirement for a common clock is met then there are two system architectures that can be deployed: -

- Clock only
- Clock and SYNC signal

**Table 1: Effect of different wired sync implementations**

| Parameter | Type of wired synchronization employed | |
|---|---|---|
| | **Clock but no SYNC** | **Clock & SYNC** |
| Offsets between anchors due to: - | • Differences in cable propagation delays<br>• Differences in initial values of system clock counters in each anchor | • Differences in cable propagation delays only |
| Calibration method | Using reference tags to determine the offset between the anchors | Using reference tags to determine the fixed offset between the anchors |
| How often is calibration required? | Every time a system synchronization is carried out | Only at installation / commissioning time or when system is physically expanded |

### 2.2.2 Synchronizing anchor nodes without using a synchronization signal

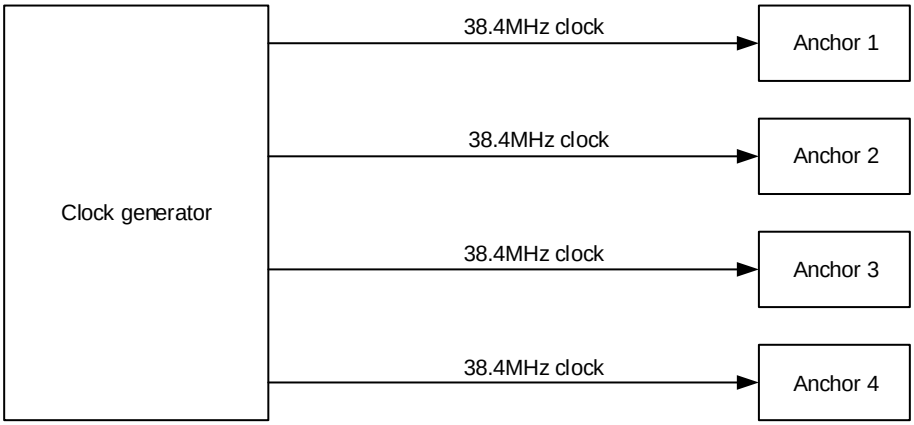


**Figure 2: Clock signal distribution**

This scheme fulfils only the minimum requirement of providing a common clock to all the anchors in the system; it does not actually synchronize the system clock counters in each anchor, it merely discovers the offsets between the system clock counters in each anchor due to two contributory factors: -

- the differences between the initial values of the system clock counters in the anchor nodes which will be different each time anchors start-up or are reset
- the differences in the propagation delays of the clock signal from the central clock source to each of the anchors which should be fixed assuming the distribution network remains unchanged

Once these offsets are determined and known, they can be removed from any subsequent TDOA calculations.

Typically the determination of these offsets is done by using reference tag transmitters positioned at known locations with respect to the anchors.

For any pair of anchors, the reference tag blink time difference of arrival is averaged for a period (of 1000's of blinks say), and the difference between this average and the TDOA that is mathematically predicted (based on the tag's known position relative to the anchors) is saved in the system as the fixed offset between that pair of anchors.

Later, in normal operation of the system, when that pair of anchors is used to yield a TDOA figure for any arriving blink message, the fixed offset determined during the installation learning phase is used to calculate a corrected TDOA figure for multilateration.

The challenge with this implementation is that if a resynchronization of the system should ever be required, for whatever reason, the system must go through this complete process again since the offsets between system clock counters in the anchor nodes are due not only to the fixed delay in the distribution network but also the unknown initial value in each of the system clock counters.

### 2.2.3 Synchronizing the anchors using a synchronization signal

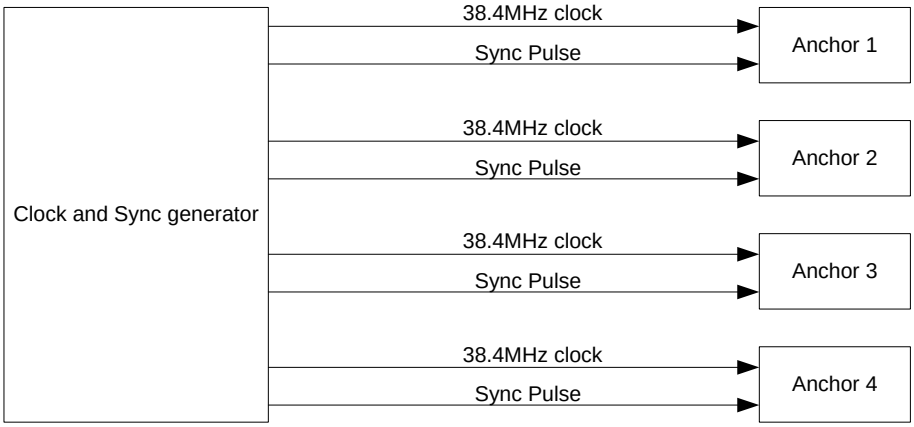


**Figure 3: Clock and SYNC signal distribution**

The DW1000 has a SYNC input pin that can be used to reset the on-chip system clock's counter value to zero. This can be used to solve the initial starting-time problem. A SYNC control signal can be distributed to all the anchor nodes so that they all have their system clocks reset by the same event in time (albeit at a different fixed offset per anchor due to the different propagation delay of the SYNC signal to each anchor). The SYNC signal can be distributed alongside the 38.4 MHz clock.

Alternatively to reduce the number of wires required it is possible to send the SYNC signal by embedding it in the CLOCK signal as a violation in the clock that is distributed to the anchors. At each anchor the violation is detected to recover the SYNC signal to feed into the IC.

The reset of the DW1000 system counter using the SYNC input is a function that needs to be enabled via an IC feature called One Shot Timebase Reset (OSTR) where the DW1000 is configured under software control to accept the SYNC input and reset the system clock counter.

The benefit of this scheme is that the offsets between the anchor nodes are now due purely to the difference in the propagation delays between the central clock & SYNC source and each anchor node because the application of the SYNC signal resets the on-chip system clock counters to a known value. Once the fixed offsets between anchor nodes are known then whenever synchronization is necessary all that is required is the issuing of a SYNC signal from the central controller.

It is only necessary to use reference tags to determine the fixed offsets due to the propagation delays in the clock and SYNC distribution cables once at installation / commissioning time (or whenever the system is physically expanded by the addition of anchor nodes) because these delays should not change unless the physical structure of the distribution network is modified in some way.

# 3 WIRED SYNC HARDWARE

## *3.1 Hardware to distribute clock and SYNC*

To implement a wired sync system certain hardware is required: -

- a central unit that generates the clock and SYNC reference signals
- a distribution network
- repeater units

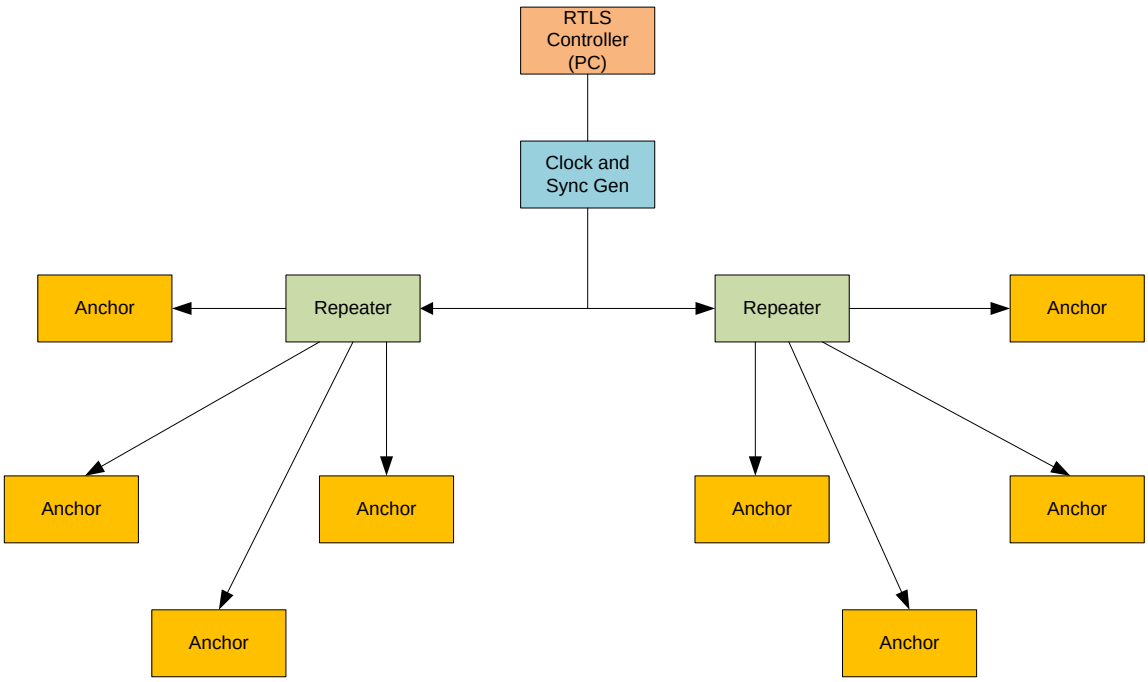


**Figure 4: Wired Sync infrastructure hardware**

### 3.1.1 Clock & SYNC generation unit

This is the central time-base generator that generates the clock and SYNC reference signals for the entire system. The clock and SYNC pulses must be generated to meet the requirements of the DW1000 detailed in section 0; see also ref [2].

The unit should be capable of driving the clock and SYNC signals (either as a combined signal where the SYNC is a violation of the normal clock signal, or as a pair of signals) over the distribution network which is typically constructed using CAT5 cable unshielded twisted pairs. This requires differential signalling.

As the SYNC signal needs to be controlled by the RTLS system software this unit also requires an interface to connect to the RTLS server.

### 3.1.2 Distribution network

To distribute the clock and SYNC signal to the anchor nodes commonly available infrastructure such as CAT5 cables are typically used. A typical limit for a run length of CAT5 is 100 m.

### 3.1.3 Repeater units

A single clock and SYNC generation unit is unlikely to be sufficient for all but the smallest of RTLS installations. Therefore a number of regeneration /splitter units will most likely be required. These units clean up both the clock and SYNC signals before re-transmitting them with a deterministic delay.

## *3.2 Hardware Implementation*

### 3.2.1 Transmitting the 38.4 MHz clock over CAT5 cable

100 m of CAT5 cable results in up to 12.5 dB attenuation. While this does not affect the frequency of the clock signal it does affect the amplitude which must be high enough to meet the specification and any recovery of the clock must provide the phase noise profile required for the EXTCLK of the DW1000.

The recommended approach for this is to use the Si5317 jitter cleaner. See ref [4]. Of the methods DecaWave has tested this allowed the 38.4 MHz clock to be transmitted over 100 m of CAT5 cable and be recovered sufficiently to have no impact on the receiver sensitivity performance of the DW1000 receiver.

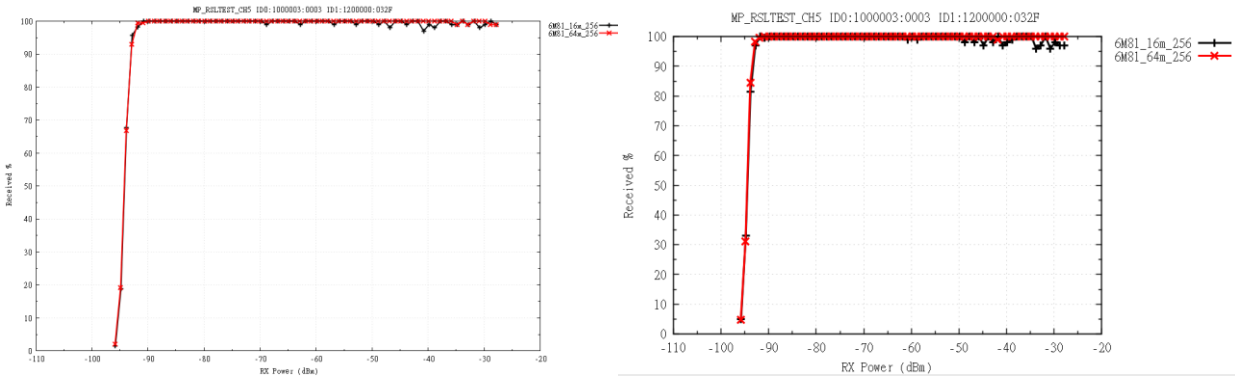


**Figure 5: Receiver sensitivity measurements for local clock (left) and 100 m CAT5 / Si5317 clock (right)**

The recommended implementation schematic for the Si5317 is given in ref [4]. Be sure to configure the device for the correct operating frequency of 38.4 MHz.

### 3.2.2 Transmitting the SYNC signal over CAT5 cable

The recommended approach for this is to use the ADCMP600 family of fast comparators which can accept a differential input signal and generate a CMOS level output signal. See ref [5].

### 3.2.3 Transmitting the SYNC signal as a violation of the clock signal

To avoid having to transmit the SYNC signal over a separate twisted pair in the CAT5 cable it is possible to incorporate it into the clock signal as a violation of that clock signal e.g. by removing one of the clock pulses. Circuitry in the central clock & SYNC source can combine the clock and SYNC signals and circuitry in the anchor nodes can detect the absence of this clock pulse and regenerate the SYNC signal for application to the SYNC pin on the DW1000 as shown in Figure 6 below.

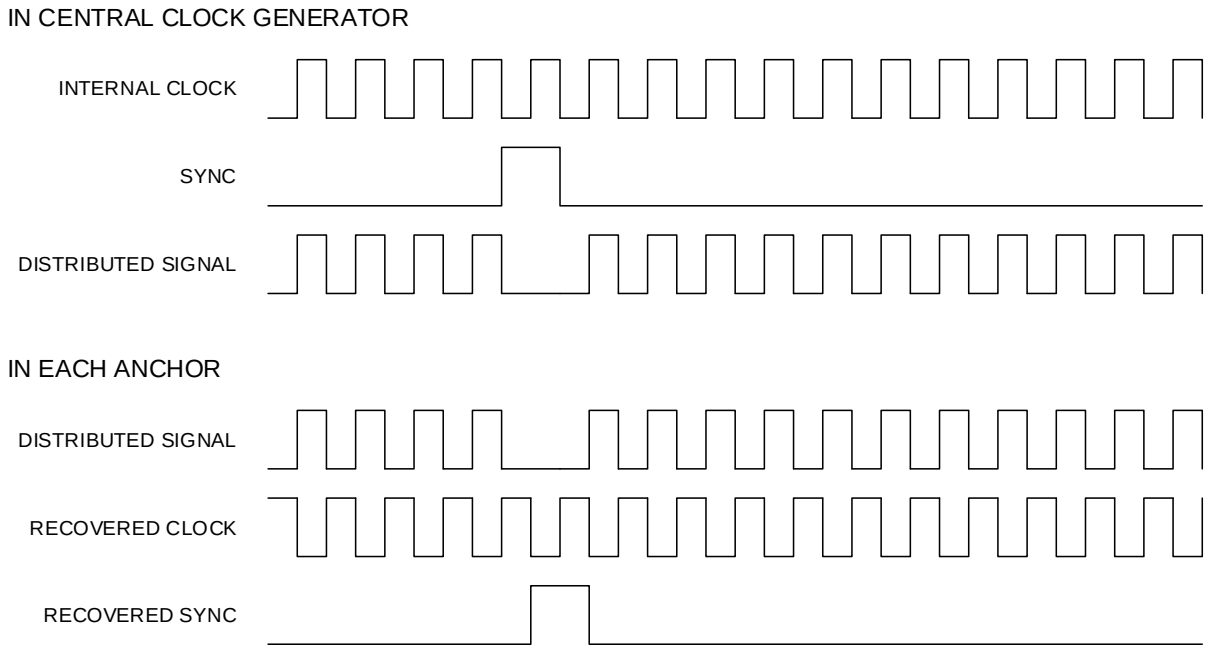


**Figure 6: Using a clock violation to distribute the SYNC signal**

In such a scheme care should be taken to maintain the phase relationship between the recovered clock and SYNC signals such that the setup and hold times of the SYNC signal at the DW1000 are respected. Various circuit implementations of this scheme are possible but in all cases it is important that the clock jitter of the recovered clock be kept within the required clock specifications of the DW1000 otherwise performance will be degraded.

## *3.3 DW1000 and Wired Sync*

Wired sync with the DW1000 is made possible through the use of the external clock (38.4 MHz clock signal), the SYNC pin and a special mode called One Shot Timebase Reset (OSTR) Mode.

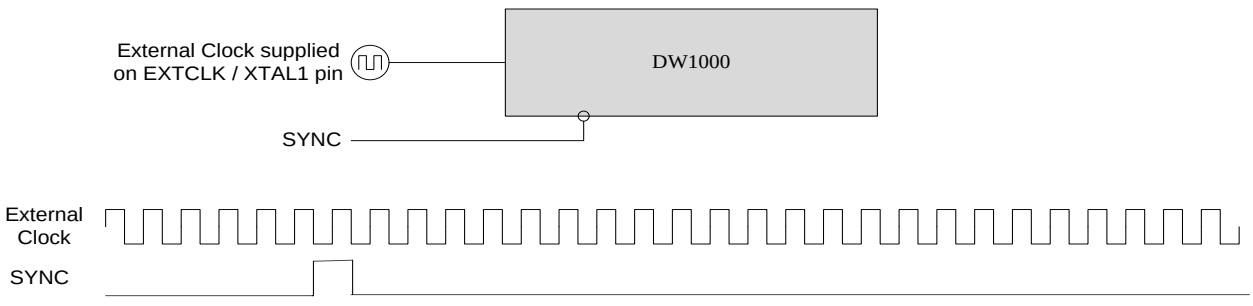


**Figure 7: OSTR Clock and Sync**

The operation of this mode is controlled via the EC_CTRL register in the DW1000 with bit definitions as follows: -

| ID | Length (octets) | Type | Mnemonic | Description |
|---|---|---|---|---|
| 24:00 | 4 | RW | EC_CTRL | External clock synchronisation counter configuration |

The EC_CTRL register contains the following sub-fields:

| REG:24:00 –EC_CTRL– External clock synchronisation counter configuration | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | OSTRM | WAIT | | | | | | | | - | OSRSM | OSTSM |

The fields of the EC_CTRL register identified above are individually described below: -

| Field | Description of fields within Sub-Register 0x24:00 EC_CTRL |
|---|---|
| OSTSM bit:0 | External transmit synchronisation mode enable. |
| OSRSM bit:1 | External receive synchronisation mode enable. |
| WAIT bits:10:3 | Wait counter used for external transmit synchronisation and external timebase reset. |
| OSTRM bit:11 | External timebase reset mode enable. |

**Figure 8: EC_CTRL Register**

One Shot Timebase Reset (OSTR) mode allows a reset to be applied to the main timebase counter in the digital baseband section of the DW1000 at a deterministic and predictable (tolerance specified at <300 ps, real world measurements at the IC level indicate a much lower standard deviation of 12-14 ps) time relative to a sync event. By applying the same sync event to multiple DW1000 devices whose clocks all come from the same source their internal timebases can be synchronised very accurately provided any fixed delays in the clock and sync distribution network are taken into account.

To enter this mode the OSTRM bit in the EC_CTRL register is set and the WAIT value set to a suitable value. When a rising edge of the EXT_CLK captures the SYNC pulse high the OSTR process is initiated. This causes some counters in the PLL to be reset (to establish a fixed phase relationship, this makes it necessary that the transmitter and receiver are disabled to prevent unexpected behaviour) and starts counting the EXT_CLK cycles. When the count of the EXT_CLK cycles equals the OSTS_WAIT value the reset of the system counter is initiated. This does involve a clock domain transfer, but this is forced to be predictable through the aforementioned reset in the PLL. Testing was carried out using a WAIT setting of 33.

**Table 2: External clock specifications**

| Parameter | Min. | Typ. | Max. | Units | Condition/Note |
|---|---|---|---|---|---|
| External Reference | | | | | |
| Amplitude | 0.8 | | | $V_{pp}$ | Must be AC coupled. |
| SSB phase noise power density | | | -132 | dBc/Hz | @1 kHz offset. |
| SSB phase noise power density | | | -145 | dBc/Hz | @10 kHz offset. |
| Duty Cycle | 40 | | 60 | % | |

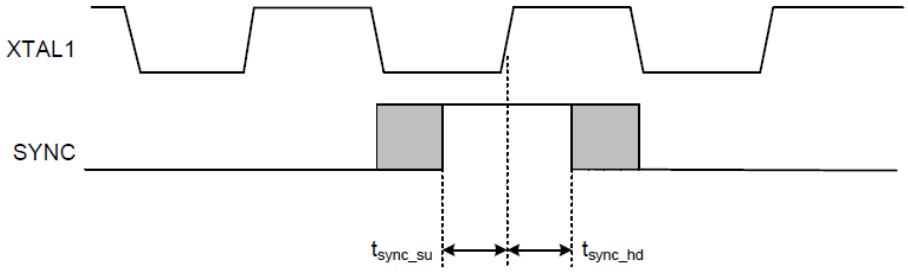


**Figure 9: SYNC signal timing diagram**

**Table 3: SYNC signal timing specifications**

| Parameter | Min | Typ | Max | Unit | Description |
|---|---|---|---|---|---|
| $t_{SYNC\_SU}$ | 10 | | | ns | SYNC signal setup time before XTAL1 rising edge |
| $t_{SYNC\_HD}$ | 10 | | | ns | SYNC signal hold time after XTAL1 rising edge |

# 4 WIRED SYNC SOFTWARE

## 4.1 Prerequisites

For a wired sync system to work correctly the software needs to know the following: -

- The position of each anchor
- The position of a reference tag or tags during installation (NB: it may be possible to use anchors as reference transmitters for other pairs of anchors that are in range)

## 4.2 Determining when to synchronize

There are a number of different situations in which a resynchronization of the system is required: -

- If an anchor restarts or a new anchor is introduced to the system then it will have an arbitrary clock counter value with respect to all the other anchors and so a system wide re-synchronization will be required.
- During installation / calibration of the system, before the system can begin to learn the fixed offsets between the anchor nodes, a re-synchronization of the system is required to set all anchors into the starting state for learning the fixed offsets.
- For commissioning the system and for system debug it is useful to have a user system command that initiates the re-synchronization process.

## 4.3 Synchronization process

A number of steps are required to synchronize all anchors in the system: -

- The system controller central location engine (CLE) issues a command to all the anchor nodes to prepare for a One Shot Timebase Reset (OSTR) input on the SYNC input pin.
- Each anchor receives the command to prepare for OSTR. Each anchor typically discontinues all tag blink time of arrival reporting (TOA), configures the DW1000 for the OSTR, and reports back to the CLE that it is “ready” for OSTR (denoted RFOSTR).
- When the CLE receives RFOSTR reports from all anchors in the system it then commands (by whatever means is appropriate depending on the specific system design) the external clock and SYNC generation hardware to generate the physical SYNC signal that traverses the clock / sync distribution network and eventually drives the DW1000 SYNC input in each anchor node.
- The micro-controller in each anchor node receives an interrupt from the DW1000 indicating that the OSTR has occurred. The anchor node reports this event back to the CLE and then resumes reporting tag blink time of arrival (TOA) to the CLE.

## 4.4 Calibrating out fixed offsets

As discussed previously it is necessary for the CLE to know the fixed offsets between anchor nodes caused by differences in cable lengths from the central clock / sync source to each anchor node so that these can be removed from TDOA calculations.

In “Learning” mode the CLE averages TOA for reference tags (or anchors). Because the system knows the location of the anchor nodes and the reference tags it can calculate the expected TDOA

assuming no offset between the anchor nodes. After a predetermined period (or a number of TOA) the system can accurately calculate the fixed TDOA offset for each pair of anchors and can save these results in its configuration store (or database) for use during operational mode.

## *4.5 In operation*

When “Learning” mode is complete and the system switches to “Operational” mode, the CLE calculates TDOA data for pairs of anchors (from their TOA reports) and corrects the TDOA values using the fixed offset values (previously determined in Learning mode).

These corrected TDOA values are then passed to the multilateration function which solves between groups of TDOA values to yield the estimates of the tags’ locations.

# 5 REFERENCES

## *5.1 Listing*

Reference is made to the following documents in the course of this Application Note: -

**Table 4: Table of References**

| Ref | Author | Version | Title |
|---|---|---|---|
| [1] | DecaWave | Current | DW1000 Data Sheet |
| [2] | DecaWave | Current | DW1000 User Manual |
| [3] | IEEE | 2011 version | IEEE 802.15.4-2011 or "IEEE Std 802.15.4™-2011" (Revision of IEEE Std 802.15.4-2006). IEEE Standard for Local and metropolitan area networks— Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). IEEE Computer Society Sponsored by the LAN/MAN Standards Committee. Available from http://standards.ieee.org/. |
| [4] | Silicon Labs | Rev 1.1 | Si5317 pin-controlled 1–711 MHz jitter cleaning clock data sheet |
| [5] | Analog Devices | Rev A | Rail-to-Rail, Very Fast, 2.5 V to 5.5 V, Single-Supply TTL/CMOS Comparators data sheet |

# 6 FURTHER INFORMATION

Decawave develops semiconductors solutions, software, modules, reference designs - that enable real-time, ultra-accurate, ultra-reliable local area micro-location services. Decawave's technology enables an entirely new class of easy to implement, highly secure, intelligent location functionality and services for IoT and smart consumer products and applications.

For further information on this or any other Decawave product, please refer to our website www.decawave.com.

# 7 APPENDIX 1: DECAWAVE TESTING OF OSTR AND SYNCHRONIZATION PERFORMANCE

## 7.1 IC Level Testing of OSTR mode

The testing of OSTR mode is intended to ensure that the delay between the application of the SYNC event at the SYNC pin and the reset of the DW1000 internal system counter is consistent.

To do this the DW1000 is placed into OSTR mode and an independent counter is also primed to be reset by the SYNC event. While the independent counter is reset at the SYNC event the system counter will be reset after the WAIT counter has elapsed. A number of receive events are then time-stamped using both counters. If the OSTR mode is behaving correctly then the offset between the counters should remain consistent regardless of additional SYNC events.

**Table 5: OSTR performance over process and temperature**

| DW1000 Device # | Silicon Manufacturing Process Variation | Temperature | OSTR Mean Offset (ns) | OSTR STD Dev Offset (ns) |
|---|---|---|---|---|
| 709 | Slow | Room | 867.4099254 | 0.012220933 |
| 709 | Slow | +85 | 867.4103287 | 0.012001977 |
| 806 | Fast | Room | 864.4068097 | 0.011940458 |
| 806 | Fast | -40 | 864.4073158 | 0.011803035 |
| 616 | Typical | Room | 866.4090273 | 0.012080554 |

Testing was carried out over semiconductor manufacturing process and temperature variations. Temperature does not have a significant impact on the measured delay, but process does. In a wired sync RTLS this has no effect because the individual DW1000 device variations will be taken into account as part of the calibration procedure used to compensate for the propagation delay in the SYNC and clock cables to each of the anchor nodes.

## 7.2 Testing of a wired sync RTLS

A full implementation of a wired sync system has not yet been tested by Decawave.

Some preliminary testing has been performed using existing anchor nodes modified to accept an external (transported over co-axial cable) and incorporated into our RTLS Demo system.

A SYNC pulse is not distributed in this system so it is necessary to compensate for the offset between devices using a training phase with wireless transmissions between anchors.

Initial measurements between two anchors resulted in TDOA measurements with a standard deviation of 140 ps, as compared with 250-300 ps in the equivalent wireless sync based RTLS Demo System.