DWM1001 System Overview And Performance

Version 2.1

This document is subject to change without notice

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

DOCUMENT INFORMATION

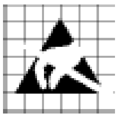
Disclaimer

Decawave reserves the right to change product specifications without notice. As far as possible changes to functionality and specifications will be issued in product specific errata sheets or in new versions of this document. Customers are advised to check with Decawave for the most recent updates on this product.

Note: For the purpose of this document, "**DWM1001**" can also refer to "**DWM1001C**". The only difference between both modules is that **DWM1001C** is certified and its OTP memory is calibrated to comply with FCC/ETSI regulations.

LIFE SUPPORT POLICY

Decawave products are not authorized for use in safety-critical applications (such as life support) where a failure of the Decawave product would reasonably be expected to cause severe personal injury or death. Decawave customers using or selling Decawave products in such a manner do so entirely at their own risk and agree to fully indemnify Decawave and its representatives against any damages arising out of the use of Decawave products in such safety-critical applications.



Caution! ESD sensitive device. Precaution should be used when handling the device in order to prevent permanent damage.

DISCLAIMER

(1) This Disclaimer applies to the software provided by Decawave Ltd. (“Decawave”) in support of its DWM1001 module product (“Module”) all as set out at clause 3 herein (“Decawave Software”).

(2) Decawave Software is provided in two ways as follows: -
(a) pre-loaded onto the Module at time of manufacture by Decawave (“Firmware”);
(b) supplied separately by Decawave (“Software Bundle”).

(3) Decawave Software consists of the following components (a) to (d) inclusive:

(a) The **Decawave Positioning and Networking Stack** (“PANS”), available as a library accompanied by source code that allows a level of user customisation. The PANS software is pre-installed and runs on the Module as supplied, and enables mobile “tags”, fixed “anchors” and “gateways” that together deliver the DWM1001 Two-Way-Ranging Real Time Location System (“DRTLS”) Network.

(b) The **Decawave DRTLS Manager** which is an Android™ application for configuration of DRTLS nodes (nodes based on the Module) over Bluetooth™.

(c) The **Decawave DRTLS Gateway Application** which supplies a gateway function (on a Raspberry Pi ®) routing DRTLS location and sensor data traffic onto an IP based network (e.g. LAN), and consists of the following components:

- DRTLS Gateway Linux Kernel Module
- DRTLS Gateway Daemon
- DRTLS Gateway Proxy
- DRTLS Gateway MQTT Broker
- DRTLS Gateway Web Manager

(d) **Example Host API functions**, also designed to run on a Raspberry Pi, which show how to drive the Module from an external host microprocessor.

(4) The following third party components are used by Decawave Software and are incorporated in the Firmware or included in the Software Bundle as the case may be: -

(a) The PANS software incorporates the Nordic SoftDevice S132-SD-v3 version 3.0.0 (production) which is included in the Firmware and is also included in the Software Bundle;

(b) The PANS software uses the eCos RTOS which is included in the Software Bundle. The eCos RTOS is provided under the terms of an open source licence which may be found at: http://ecos.sourceware.org/license-overview.html;

(c) The PANS software uses an open source CRC-32 function from FreeBSD which is included in the Software Bundle. This CRC-32 function is provided under the terms of the BSD licence which may be found at:

https://github.com/freebsd/freebsd/blob/386ddae58459341ec567604707805814a2128a57/COPYRIGHT;

(d) The Decawave DRTLS Manager application uses open source software which is provided as source code in the Software Bundle. This open source software is provided under the terms of the Apache Licence v2.0 which may be found at http://www.apache.org/licenses/LICENSE-2.0;

(e) The Decawave DRTLS Gateway Application uses the following third party components: -

(i) The Linux Kernel which is provided as source code in the Software Bundle. The Linux Kernel is provided under the terms of the GPLv2 licence which may be found at: https://www.gnu.org/licenses/old-licenses/gpl-2.0.en.html and as such the DWM1001 driver component of the DRTLS Gateway Application is provided under the same license terms;

(ii) The three.js JavaScript library, the downloadable version of which is available here https://threejs.org/, is provided under the terms of the MIT Licence which may be found at https://opensource.org/licenses/MIT.

Items (a), (b), (c), (d) and (e) in this section 4 are collectively referred to as the “Third Party Software”

(5) Decawave Software incorporates source code licensed to Decawave by Leaps s.r.o., a supplier to Decawave, which is included in the Firmware and the Software Bundle in binary and/or source code forms as the case may be, under the terms of a license agreement entered into between Decawave and Leaps s.r.o.

(6) Decawave hereby grants you a free, non-exclusive, non-transferable, worldwide license without the right to sub-license to design, make, have made, market, sell, have sold or otherwise dispose of products incorporating Decawave Software, to modify Decawave Software or incorporate Decawave Software in other software and to design, make, have made, market, sell, have sold or otherwise dispose of products incorporating such modified or incorporated software PROVIDED ALWAYS that the use by you of Third Party Software as supplied by Decawave is subject to the terms and conditions of the respective license agreements as set out at clause 4 herein AND PROVIDED ALWAYS that Decawave Software is used only in systems and products based on Decawave semiconductor products. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER DECAWAVE INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY THIRD PARTY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT, IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which Decawave semiconductor products or Decawave Software are used.

(7) Downloading, accepting delivery of or using Decawave Software indicates your agreement to the terms of (i) the license grated at clause 6 herein, (ii) the terms of this Disclaimer and (iii) the terms attaching to the Third Party Software. If you do not agree with all of these terms do not download, accept delivery of or use Decawave Software.

(8) Decawave Software is solely intended to assist you in developing systems that incorporate Decawave semiconductor products. You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your systems and products. THE DECISION TO USE DECAWAVE SOFTWARE IN WHOLE OR IN PART IN YOUR SYSTEMS AND PRODUCTS RESTS ENTIRELY WITH YOU AND DECAWAVE ACCEPTS NO LIABILTY WHATSOEVER FOR SUCH DECISION.

(9) DECAWAVE SOFTWARE IS PROVIDED "AS IS". DECAWAVE MAKES NO WARRANTIES OR REPRESENTATIONS WITH REGARD TO DECAWAVE SOFTWARE OR USE OF DECAWAVE SOFTWARE, EXPRESS, IMPLIED OR STATUTORY, INCLUDING ACCURACY OR COMPLETENESS. DECAWAVE DISCLAIMS ANY WARRANTY OF TITLE AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS WITH REGARD TO DECAWAVE SOFTWARE OR THE USE THEREOF.

(10) DECAWAVE SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY THIRD PARTY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON DECAWAVE SOFTWARE OR THE USE OF DECAWAVE SOFTWARE. IN NO EVENT SHALL DECAWAVE BE LIABLE FOR ANY ACTUAL, SPECIAL, INCIDENTAL, CONSEQUENTIAL OR INDIRECT DAMAGES, HOWEVER CAUSED, INCLUDING WITHOUT LIMITATION TO THE GENERALITY OF THE FOREGOING, LOSS OF ANTICIPATED PROFITS, GOODWILL, REPUTATION, BUSINESS RECEIPTS OR CONTRACTS, COSTS OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION), LOSSES OR EXPENSES RESULTING FROM THIRD PARTY CLAIMS. THESE LIMITATIONS WILL APPLY REGARDLESS OF THE FORM OF ACTION, WHETHER UNDER STATUTE, IN CONTRACT OR TORT INCLUDING NEGLIGENCE OR ANY OTHER FORM OF ACTION AND WHETHER OR NOT DECAWAVE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, ARISING IN ANY WAY OUT OF DECAWAVE SOFTWARE OR THE USE OF DECAWAVE SOFTWARE.

(11) You acknowledge and agree that you are solely responsible for compliance with all legal, regulatory and safety-related requirements concerning your products, and any use of Decawave Software in your applications, notwithstanding any applications-related information or support that may be provided by Decawave.

(12) Decawave reserves the right to make corrections, enhancements, improvements and other changes to its software, including Decawave Software, at any time.

Mailing address: Decawave Ltd.,
Adelaide Chambers,
Peter Street,
Dublin D08 T6YA
IRELAND.

1 INTRODUCTION

1.1 Overview

This document describes the details and performance of DWM1001 Two-Way-Ranging Real Time Location System (DRTLS). The major components are:

- Decawave DWM1001 module hardware
- Decawave Positioning and Networking Stack (PANS) software

1.2 Audience

This document is suitable for customers (engineers, RTLS system designers) of the DWM1001 [1], DWM1001-DEV [2] and MDEK1001 [3] products who are integrating the technology into their own products.

1.3 The DWM1001 module and RTLS

The DWM1001 is a module product that comes complete with firmware to allow system developers to quickly implement an RTLS to suit their particular end application, or add RTLS capability to an existing system. The module may be configured to behave as an "anchor" one of the fixed nodes in the system, "bridge" a fixed node that routes the data between the UWB network and IP network or a "tag" one of the mobile located nodes in the system. The module configuration may be achieved either via Bluetooth using the companion application (Decawave DRTLS Manager [4]); via an SPI or UART connection from an external host [5] or remotely from a Web client via the UWB backhaul.

The module incorporates Decawave's DW1000 UWB transceiver which the module's on-board firmware drives to implement the network of anchor nodes and perform the two-way ranging exchanges with the tag nodes enabling each tag to compute its own location.

The module also incorporates the Nordic Semiconductor nRF52832 IC providing the Bluetooth connectivity used for configuration and the microprocessor that runs the firmware which drives the DW1000 and provides the RTLS enabling functionality. A more complete description of this may be found in section 3.

The module is typically mounted on a PCB, such as the DWM1001-DEV product. The MDEK1001 kit provides 12 DWM1001modules already mounted on "development" boards enabling system developers evaluate the product and/or begin their system development before embarking on their own designs.

1.4 More Information

Information about the DWM1001 and related products can be found in the following documentation:

- DWM1001 (module)
    - DWM1001 Product Brief
    - DWM1001 Hardware Datasheet
    - DWM1001 Firmware User Guide
    - DWM1001 Firmware API Guide
- DWM1001-DEV (development board)
    - DWM1001-DEV Product Brief
    - DWM1001-DEV Hardware Datasheet
    - DWM1001-DEV Quick Start Guide (in the box)

- MDEK1001 (development and evaluation kit)
  - MDEK1001 Product Brief
  - MDEK1001 System User Manual
  - MDEK1001 Quick Start Guide (in the box)
  - MDEK1001 Application Manager Source Code Guide
- www.decawave.com

2 SUMMARY PERFORMANCE

The table below summarises the performance of the DRTLS. See the DWM1001 datasheet for more detailed information about the module hardware.

Table 1: Summary System Performance

Parameter	Description	Notes
RTLS System Performance		
X-Y location accuracy	<10 cm (typical)	Line-of-Sight (LOS)
System capacity / cluster	150 Hz	750 tags @ 0.2 Hz 150 tags @ 1 Hz 15 tags @ 10 Hz etc.
Max. Location Rate / Tag	10 Hz	
Min. Location Rate / Tag	0.0167 Hz	Every 1 minute
Max # Anchors (theoretical)	Area Dependent	See section 5.1.1
Max. # Tags / cluster (theoretical)	9000	@ min. rate of 0.0167 Hz (every 1 minute)
Available Memory		
Flash Memory available to user	40kB	
RAM available to user	5kB	
Data Throughput		
Tags to gateway (sensor data)	Uplink or Downlink: 340 Bytes per second	Each tag
Anchors to gateway (sensor data)	Uplink: 34 Bytes per 12 s Downlink: 34 Bytes per 12 s	Each anchor
System Latency	100 ms	From tag to gateway
Encryption	AES encryption of UWB frames	Mutually exclusive with BLE
UWB Parameters		
UWB Channel	Channel 5 (6.5 GHz)	Fixed
Data Rate	6.81 Mbps	Fixed
PRF	64 MHz	Fixed
Preamble Length	128	Fixed
Preamble Code	9	Fixed

3 TARGET SYSTEM ARCHITECTURE

The DWM1001 hardware and software is designed to enable users to quickly build a system like the one shown in Figure 1.

The main components of such a system are:

- DWM1001 Two-Way-Ranging Real Time Location System (DRTLS) Network:
  - A UWB network consisting of anchors and tags based on the DWM1001 module.
  - A gateway to connect the anchors and tags UWB network to the broader IP network. This consists of a DWM1001 configured as a bridge node and a Linux host (e.g. Raspberry PI 3 model B)
  - Local configuration and visualisation via an Android application or Web client application
- Backhaul infrastructure e.g. Ethernet or Wi-Fi
- End Points (consumer of data):
  - DWM1001 Web Clients
  - MQTT Clients
  - Local Bluetooth-connected Tablets/Smartphones

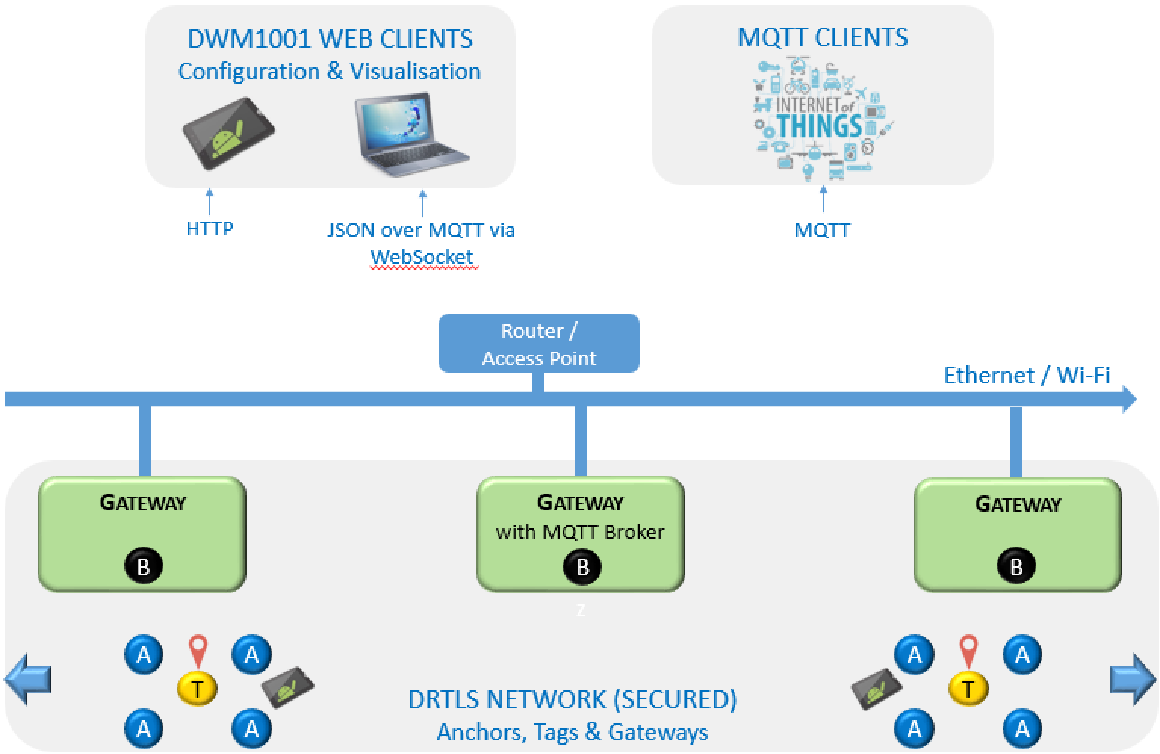


Figure 1: Target System Architecture

3.1 DRTLS Network

3.2 DWM1001 Two-Way-Ranging Real Time Location System (DRTLS)

The DRTLS network components are shown in Figure 2. The DRTLS consists of:

- A number of DWM1001 devices which can be configured as an anchor, a tag or a bridge node on a gateway. The tags are usually mobile, and anchors fixed in place.
- An optional gateway unit, containing a DWM1001 module ("bridge node"), which connects the system to the outside network (LAN/WAN).
- External applications (PC/server and tablet/phone – e.g. Decawave DRTLS Manager) which can be used to install, commission and configure the DRTLS. The tag's location can then be displayed at either the local application (tablet) or remote application.

The operation of the DRTLS network is described in Section 4.

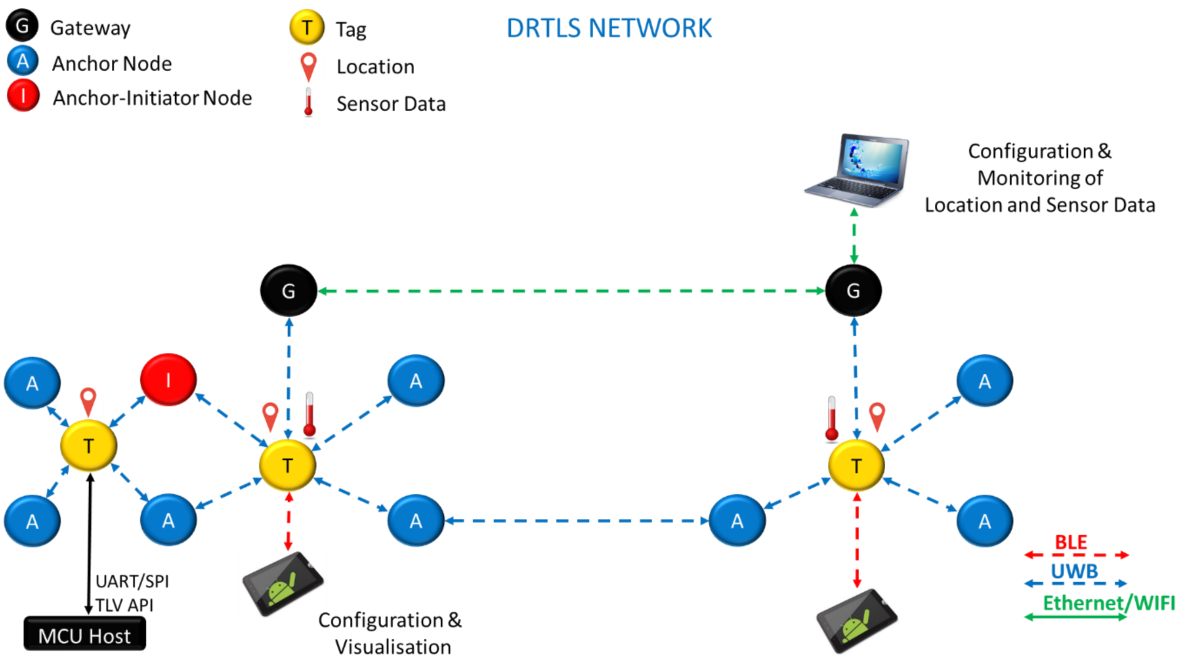


Figure 2: DRTLS network

3.3 Gateway

The gateway is a hardware and software solution allowing configuration and monitoring of a network from the LAN/WAN network.
The gateway hardware components are:

- DWM1001-Dev
- Raspberry Pi 3 model B

The gateway software components are shown in Figure 3.
The gateway consists of a combination of software components running either on the DWM1001-DEV or on the Raspberry Pi 3 Model B.

DWM1001-DEV software component:

- When a unit with PANS firmware is configured as bridge node, it acts as a bridge between the UWB network and Linux host. It collects information about the surrounding anchors, passes the *downlink* data from the Linux host to the anchors or tags and *uplink* data from the anchors and tags to the Linux host.

Raspberry Pi 3 B software components are:

- Linux kernel module
- DWM daemon
- DWM proxy
- MQTT broker
- HTTP server
- Linux Host

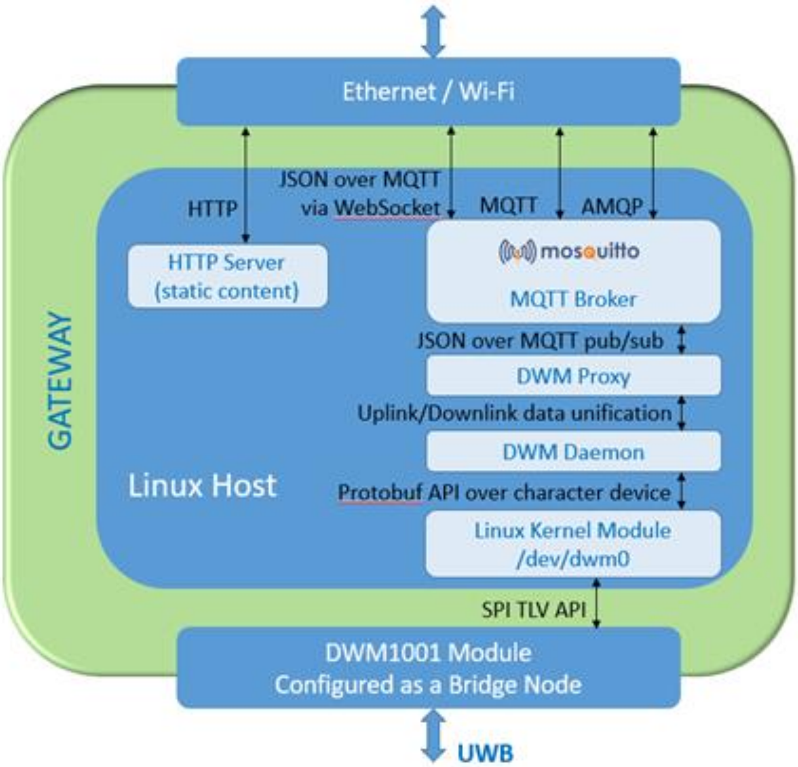


Figure 3: Gateway software components

3.4 End Points

The data from the system is visualised via Web, MQTT or Bluetooth clients.

3.4.1 DWM1001 Web Clients

Web-based clients on a PC or a mobile device such as a tablet will provide the interface for configuration and visualisation of the location data. Figure 4 shows a mock-up of this interface.

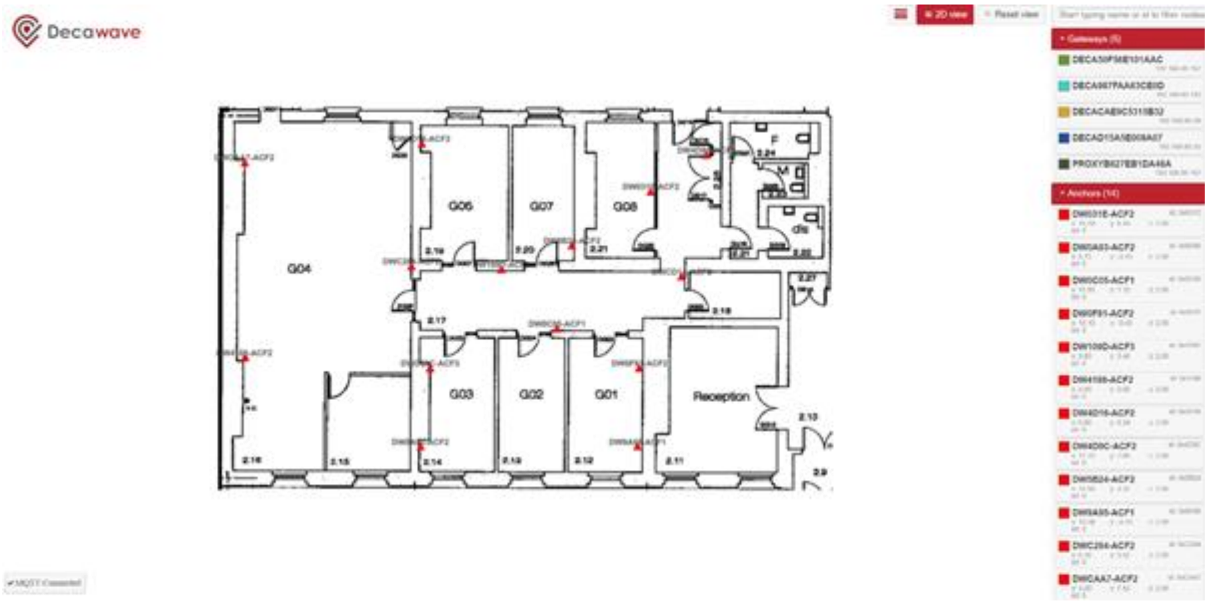


Figure 4: Web-Client interface mock up

MQTT Clients
MQTT clients will be able to connect to the MQTT Broker to access the location and IOT data sent from nodes.

3.4.2 Local Bluetooth-connected Tablets/Smartphones

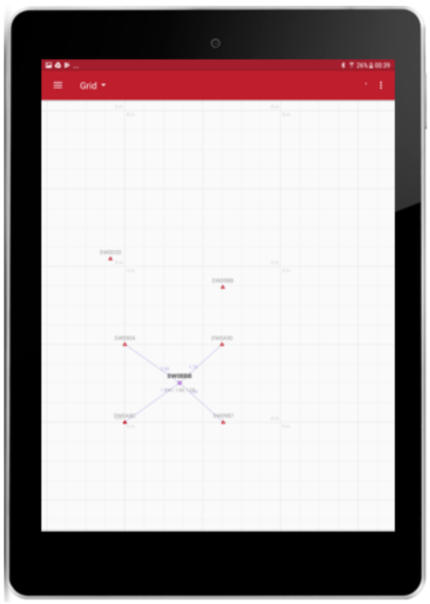


Figure 5: Local Configuration & Visualisation on the Android Application

4 DRTLS NETWORK, CONFIGURATION AND CONTROL

4.1 DRTLS network operation

A UWB DRTLS system/network allows location of tags as well as sending the data from the "cloud" (MQTT Client) to and from the tags and anchors via the bridge nodes. The network is made up of fixed nodes (anchors), mobile nodes (tags) and bridge nodes (gateway to IP network). The network installation is covered in section 4.2 below.

The DRTLS uses TDMA channel access. The nodes operate using a repeating "superframe" structure of 100 ms duration. This structure is shown in Figure 6. The initiator controls the timing and the superframe consists of 30 Beacon Message (BCN) slots, in which anchors send *Beacon* messages (9.1.1). This is followed by two Service (SVC) slots which are used for *Almanac*, network service messages (e.g. network join request) and uplink/downlink data to the anchors. There are also 15 Two-Way-Ranging (TWR) slots which are used for tag to anchor two-way ranging exchanges and also for uplink/downlink data to the tag. There are 30 additional bridge node beacons which are used to inform the tags if there is any downlink data for them. There is guard/idle time at the end of superframe which completes the superframe.

The DRTLS uses a single-sided TWR scheme, in which tag ranges with up to four anchors, and then calculates its own location with respect to the anchors' positions, which it has learnt from the *Beacon* messages. It needs a minimum of three anchors to calculate a location. The Location Engine operation is described in 6.The tags will initially listen for *Beacon* and *Almanac* messages and learn about the network topology, and then select the four anchors to range with. The details of ranging are given in 4.7. The tag's position is then sent via Bluetooth to the Decawave DRTLS Manager application (Bluetooth is enabled when tag is in *Responsive* mode, see section 4.6), where it can be visualized. The location data can also be passed via the bridge nodes, to MQTT clients for further processing/visualization.

As well as the location data tags and anchors can send up to 34B of other data in the IOT data messages. At the end of the ranging exchange the tags can either receive 34B data from the network or send 34B to the network through the gateways. The downlink takes precedence over the uplink. IOT data to/from the anchors is sent during the SVC slots.

Note, the location and ranging information can also be output over UART, when a tag is connected to a host device or a PC. This is further described in [5].

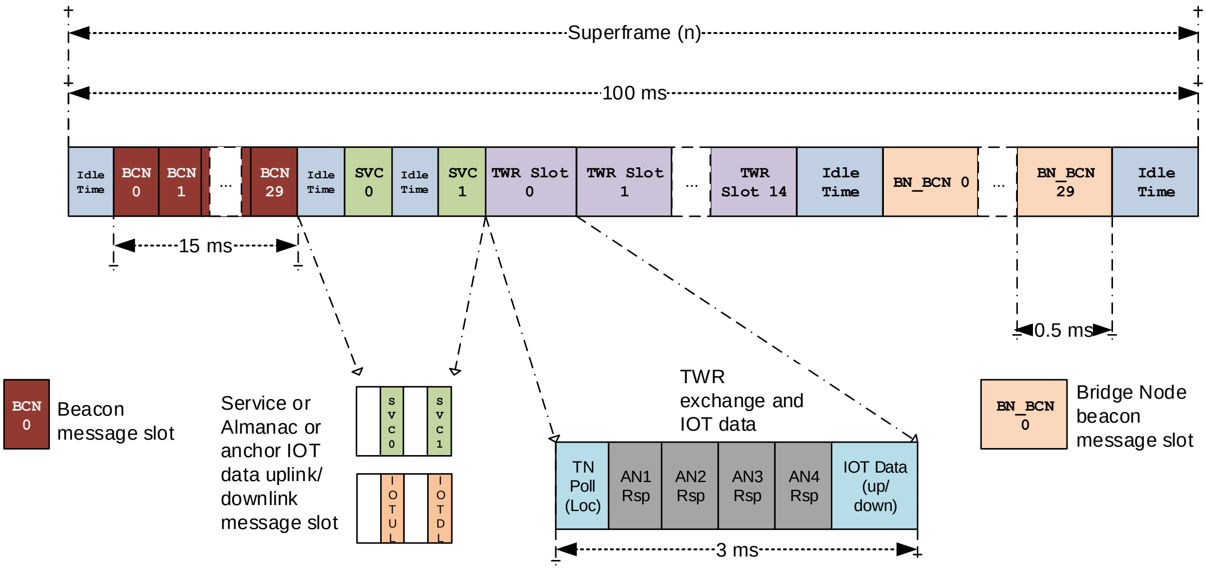


Figure 6: Superframe structure

4.2 Network initialisation

A group of nodes that form a UWB network must have the same PANID to be able to range and pass data. A network PANID (a unique 16-bit number), needs to be set by the user either via the Shell command or via the DRTLS manager as part of node configuration.

The initiator anchor will start and control the network. It is also possible to configure more than one anchor as an initiator, but only one will be active and the others will take a role of ordinary anchor. Initially every initiator begins by listening for UWB messages for a number of superframes, and if it receives none it takes a role of the initiator. It initialises the network by choosing the seat number 0, meaning that it will transmit its *Beacon* messages in BCN slot 0, and it will start a DRTLS network by broadcasting *Beacons* and *Almanacs*.

If the initiator hears messages belonging to another network (i.e. they will have different PANID), the initiator will take a role of ordinary anchor. It will join the existing network. Note the PANS FW supports cooperative network coexistence – this is described in 5.4.

When an anchor begins its operation it similarly initially listens for beacons from other anchors and then tries to join the network. Once they join the DRTLS network they will also send *Beacon* and *Almanac* messages in the assigned slots. Which contain the anchor's coordinates, general network information and anchor's TWR involvement with tags.

Each anchor uses one BCN slot and since there are 30 BCN slots in the superframe this means that at most 30 anchors can operate in the same area, each occupying its own BCN time slot.

Note: This does not mean that the system is limited to 30 anchors, but rather that across the network no new anchor can be added (take a BCN slot) if any of its in-range anchors can hear another anchor already occupying that slot. This mechanism is described in detail below, and further in section 5.1.1

4.2.1 Criteria for anchor sign-up

Before an anchor can join an existing network, the following criteria must be satisfied:

- The cluster map and cluster neighbor map (sent as part of *Beacon* messages) must indicate there is a free seat to occupy. (i.e. the number of occupied BCN slots in the superframe is < 30)
- Clock level of the in-range anchors is lower than 127. The anchors which can hear initiator's *Beacons* are at clock level 1. The next ones are clock level 2, etc.
- All in-range anchors have confirmed that the requested seat (requested by the anchor wishing to join) is free and no collisions with the other devices occurred during the sign-up process.
- The firmware must be compatible (if FW update is enabled), i.e. the version must match the version in the initiator. The firmware version is sent in the *Almanac* messages.

Note: anchors do not need to have same PANID to join the network, but unless they do they will not be able to perform TWR or transfer IOT data to/from tags.

4.2.2 Sign-up process

The new anchor continues listening to the *Beacons* and creates a list of networked anchors which it can hear. After it hears each of the anchors at least 3 times, and sees that there is a free seat, it starts looking for a free SVC slot to send a *Cluster Join Request* message. The networked anchors indicate in their *Beacons* when the SVC slot can be used for Up-Link, i.e. is available for joining anchor to send the *Cluster Join Request* message to the infrastructure.

The *Cluster Join Request* message contains information about the joining anchor (hardware version, firmware version, firmware checksum, and other node capabilities) and the requested seat number. The joined anchors send a response to the request using the extended part of the *Beacon* message. They embed a *Cluster Join Confirmation* message at the end of the *Beacon* messages. They repeat this for 3 cycles (superframes) since the last reception of the *Cluster Join Request* message. During this exchange the anchors in the network and the joining anchor are "locked" and no new anchors will be able to join. If any new anchor sends a join request it will be ignored, this means only one anchor can join at a time. The first *Cluster Join Request* message will contain seat number 0xFF which means, the joining anchor is probing if the networked anchors already have it on their lists.

During the sign-up process the following scenarios can happen:

- All networked anchors which the joining anchor can hear will respond with a valid seat number (0 to 29), that means the joining anchor was connected to the network recently, and is reconnecting. It would not need to be assigned a new seat and can continue to use the previous seat if all networked anchors replied with the same number. If the replied seat numbers are not the same, then the joining anchor will need to choose a free seat (i.e.one which none of the networked anchors are using). If no free seat is available the new anchor cannot join the network, and will have to wait and try to connect later.
- Some networked anchors will respond with a valid seat number and some with seat number 0xFF. The seat number 0xFF means those anchors don't know about the joining anchor yet. The joining anchor will need to choose a free seat (i.e.one which none of the networked anchors are using).
- Some networked anchors don't respond after the *Cluster Join Request* message. The joining anchor will mark these as gone (i.e. out of range). When all the other anchors are locked to it, it can choose a free seat.

Once the joining anchor has chosen a seat it will send *Cluster Join Request* message with the requested seat number and then when all networked anchors confirm the seat, it will consider itself connected and the sign-up process has completed successfully. The anchor will then start sending the Beacon and Almanac messages, and participating in the DRTLS.

If the over-the-air (OTA) firmware update is enabled (disabled by default in DRTLS), an anchor will firstly listen for the *Almanac* messages and checks its hardware version, firmware version and firmware size are compatible with the other devices on the network. If the version is different, then a firmware update process is initiated, see section 8, otherwise the anchor can continue the sign-up process as described above. Note: the initiator firmware should be updated first by SWD or BLE, after which each node will be updated by initiator over UWB.

4.3 Infrastructure collision detection and resolution

An *infrastructure collision* occurs when two anchors are transmitting in the same BCN slot (i.e. occupying the same seat). Each anchor participating in the network, receives *Beacon* messages from all other anchors that are in range, and at the same time monitors for any collisions. Each anchor then maintains a list of anchors for which it has detected collisions/conflicts, i.e. it detects that the same seat is used by two different anchors, it will report this if the collision counter reaches a threshold during a defined period. The anchor receiving a collision report addressed to it will leave the network and attempt to re-join which should result in it occupying free seat number.

The collision is reported using an SVC slot in the *Service* message with parameters indicating the address of the colliding anchor and the address of the reporting anchor.

4.4 Operation of a connected anchor/anchor-initiator

An anchor which can hear the initiator keeps its clock synchronised with it, so that it is aligned to the initiator's superframe timings. This anchor's clock level is said to be 1. An anchor which cannot hear the initiator will keep its clock synchronised with the anchor which is closer to the initiator (e.g. if it can hear two anchors, one with level 1 and other with level 3, it will use level 1 anchor's clock for its clock synchronisation estimation). The further the anchor is from the initiator the greater its clock level will be. The DRTLS supports a maximum clock level of 127.

Each anchor will send *Beacons* in its reserved BCN slot, based on its seat number. And also send *Almanacs* in the SVC slot during its reserved superframe. Each anchor, based on its seat number, has a reserved SVC slot in which to send the Almanac. The *Beacons* are sent every superframe and *Almanacs* every 120th superframe.

The anchor will listen during other BCN slots for any *Beacons* and during SVC slots for any *Almanacs* or other *Service* messages. It will listen at the start of the TWR slot for *Group Poll* messages. If the *Group Poll* contained its address, then it will respond with a *Response* message and perform TWR with the tag.

Upon reception of *Firmware Update Data Request* message, the anchor will provide firmware to requesting device over UWB (if the firmware update is enabled), for more information on this process see section 8.

Joined anchors are also listening to bridge node beacons (BN_BCN) in order to receive the list of IOT data being available for anchor or tags within the system. If downlink IOT data is available for a given tag, then anchor will inform this tag of the data availability in the TWR resp message.

4.5 Operation of a bridge node / gateway

For further information regarding gateway configuration and deployment, please refer to the DWM1001_Gateway_Quick_Deployment_Guide [6].

4.6 Operation of a tag

Initially a tag sleeps and periodically wakes up to listen for anchor's *Beacon and Almanacs* messages. It listens for a period of 5 superframes before returning to sleep for a set period. It will automatically wakeup and try again. The sleep period will initially be 10 s and will extend to 60 s.

When the tag receives valid *Beacons* and *Almanacs* messages, it firstly checks that it has compatible hardware version, firmware version and firmware size with the networked anchors. If the versions are incompatible then a firmware update process is initiated (if enabled), as per 8. If the versions are compatible then the tag will continue with TWR slot reservation and TWR process.

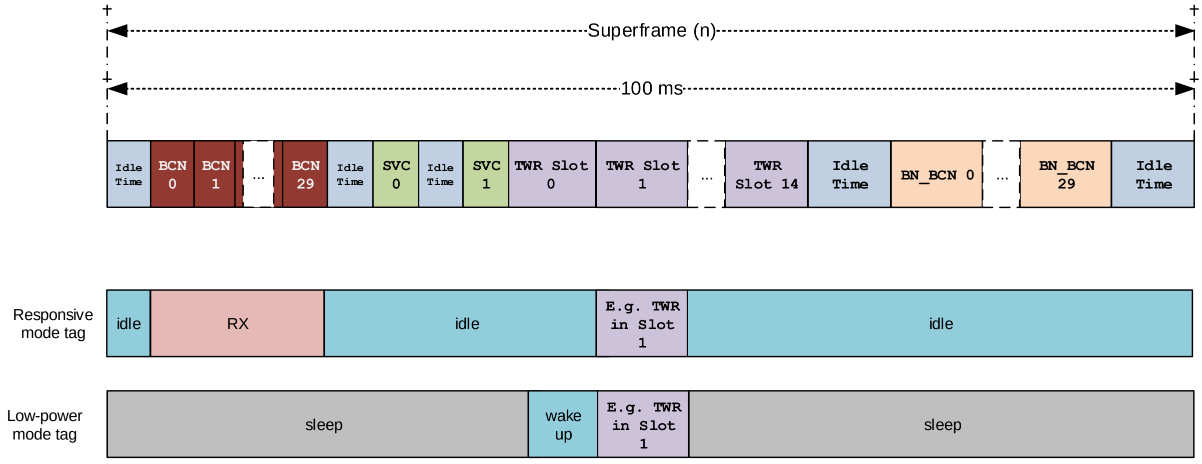


Figure 7: Tag operation in Low-power mode vs Responsive mode

A tag has two modes of operation:

- A *Responsive* mode – following the TWR exchange it will schedule the next listening period in which to listen for the *Beacons* of the superframe in which it has reserved the TWR exchange slot. The DW1000 will not be put into its low power state, but will remain in idle state (clock running). The nRF52832 MCU will be in sleep mode if no other tasks are running on the module. This mode is typically used if Bluetooth is required.
- A *Low Power* mode – following the TWR exchange the DW1000 will be put into DEEPSLEEP and will be woken up prior to the next TWR exchange. The MCU will also be put into sleep with other components of the module except the RTC and accelerometer (if accelerometer is enabled). The module is in the lowest power consumption mode. It will not listen to Beacons unless it moves out of the area in which the anchors it is currently ranging with are located. Once it leaves the area, the tag will proceed with TWR slot reservation as described below in 4.7. With BLE

enabled, the device will perform BLE scan after wake-up every set period (maximum 60 sec). The advantage of BLE scan is to be more power efficient. The device will join the UWB network only if it can detect surrounding anchors over BLE. Note that BLE advertising must be enabled on demand with user button SW2 (if using DWM1001-Dev). The BLE advertising will be automatically disable if no connection is established within 20s (and as a consequence the DWM1001 will not be able to establish a connection with another BLE device such as an android device running the DRTLS Manager).

4.7 TWR protocol and TWR slot reservation

The tag collects the ranging and data slot maps (which show the slot utilisation) from the *Beacon* messages of all anchors in range, and combines them to select a free ranging slot in the superframe in which to range. If there are no free ranging slots in the superframe it will try every 60 s to obtain new data and reserve a TWR slot. Each 100 ms superframe contains 15 ranging slots, each of which is dimensioned to allow the tag sufficient time to perform two-way ranging with 4 anchors, giving a maximum location rate capacity of 150 Hz. If all of the ranging slots are fully occupied, the system capacity is full and new tags will not be able to start ranging until the existing tags move out of the area or give up their slots.

The tag will use the reception time of the received *Beacons* to estimate its clock drift in comparison to the network time. On the clock drift is estimated, the tag can start looking for a free TWR slot. When a free data slot is available it will initiate a location attempt by sending a *Group Poll* message.
A tag in Low Power mode will use time of reception of *Response* message from the anchor to adjust its clock synchronisation.

The tag sends a *Group Poll* message containing: its location (ranging) period and a list of 4 addresses of the anchors it wishes to range to (and a bitmap with flags to indicate the anchors' seat numbers). The *Group Poll* message is a broadcast message so all anchors in range should receive it.

Each anchor listed in the group poll will (assuming it received the group poll) respond by sending a response message in turn, with the transmit time being determined by the position (index, 0 to 3) of its address in the list.

The anchors will notify the tag if downlink IOT data is available from bridge nodes. In this case, the tag will listen for downlink IOT data from bridge node in the sub-frame following the last anchor response message. (see Figure 8). If no downlink IOT data is available, then the tag can potentially transmit IOT data towards the bridge node if any data is ready to be transmitted.

Once the tag receives answers from each anchor and following the IOT uplink/downlink sub-frame, it will proceed to calculate the ranges to it from each of the anchors. If the tag gets 3 or more valid ranges it will use its internal *Location Engine* to work out its location (relative to the anchors' coordinates). Figure 8 shows the individual TWR message slots inside the superframe structure.

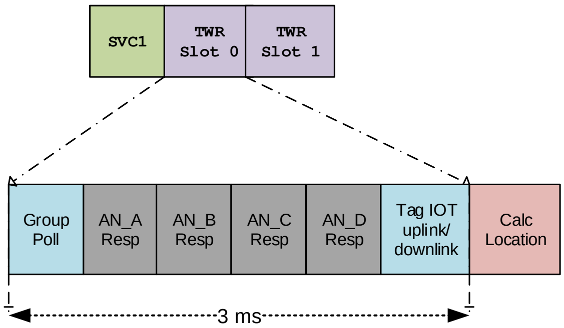


Figure 8: Superframe showing TWR frames

The tag sends its update rate in the *Group Poll* message, each of the 4 addressed anchors will send in their *Response* messages confirming that they have a free slot in the future superframe corresponding to the update rate of the tag. Using the information from the anchors (e.g. all the anchors report the future slot as available) the tag will assume that this future slot will be reserved for it. The anchors will also mark the slot as reserved thus it will not be available for other tags.

The anchor stores slot occupancy in a TDMA bitmap. The length of the TDMA map is 60 s i.e. the maximum period which tag can reserve in advance (minimum location update rate).

4.8 TWR collision detection and resolution

DRTLS employs TDMA, which means that tags perform their two-way ranging to anchors in reserved slots, thus there should be no interference. However, as a tag moves (roaming) from one area to another or during initial picking of "free" slots, two tags might transmit at the same time. To help mitigate this, tags and anchors have a collision detection and resolution algorithm. If the number of the TWR measurements is lower than expected the tag could be experiencing following scenarios:

- The selected anchors might be out of range
- There could have been a collision with another tag
- There could have been collision detection at the anchor side

An anchor sends in one response slot and receives in three other response slots so it may see a message from an anchor that is not in the list of group poll and this means collision. If during the *Response* phase the anchor received a *Response* frame with different source and/or destination address than expected, or if it received a different type of message than expected, it will signal a collision. It will report this in the *Response* message.

The tag will signal a collision if a frame is received with different source and/or destination address than expected, or an unexpected message is received, or no *Response* received or less than three *Responses* received. It will use this information to decide if it should keep using the ranging slot of it should give it up and try to look for a new data slot. The decision is based on following criteria:

- At the end of the *Response* phase if the tag received less *Responses* than expected and collision was detected, then it will give up the slot if the amount of received *Responses* was less than three (minimum amount of distances to calculate a position) or the amount of next slot reservation confirmation is less than three.
- If all the anchors confirmed the next slot then it will continue using it.
- If some anchors did not confirm the next slot then a collision is assumed and it will give up the slot. And then in the next superframe check again the availability of a slot

from the anchor's slot usage bitmap, and if any free it will try and reserve it for its use.

4.9 *Tag's TWR Strategy*

A tag collects information about anchors by listening to the *Beacon* messages. It will create a list of anchors from which it already received the positions. Then it will calculate distances to each of the anchors on the list, based on its current position (if it does not know its position it will use 0,0,0) it can then decide which anchors to choose for the next measurement using the following criteria:

- If possible choose an anchor from each quadrant, i.e. the tag will be surrounded by the anchors with whom it will range with. The tag is inside the polygon created by the selected anchors.
- Select the anchors which are nearest to the tag. It will keep using the selected anchors until it leaves the polygon or measurement with the selected anchors is no longer possible (TWR failed or collision detected).

Consider Figure 9, here T0 selects A1, A2, A9 and A7 as 4 anchors to range with. Each anchor is in a separate quadrant (Q1, Q2, Q3 and Q4).

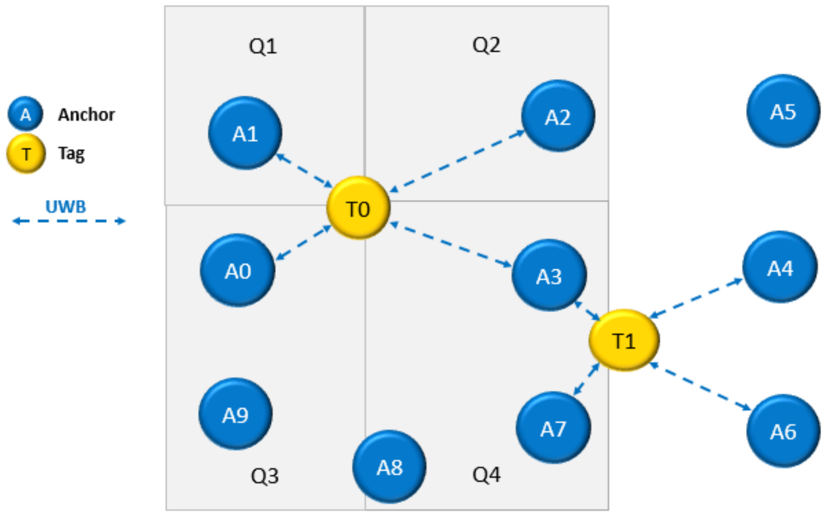


Figure 9: Tags choice of anchors for TWR

5 SCALABILITY

5.1 System Expansion - Anchors

The network infrastructure consists of anchor-initiators and anchors. The entire network uses a TDMA scheme, and thus all anchor nodes need to keep synchronised with the superframe timing of the initiator so they can send in their designated slots.

The network uses collision avoidance, collision detection and collision resolution techniques. Each anchor obtains a BCN slot/seat (4.2.2) and then can participate in the network and transmit *Beacon* and *Almanac* messages.

The system can be scaled to large network sizes but there are a number of scaling rules that must be followed to allow this.

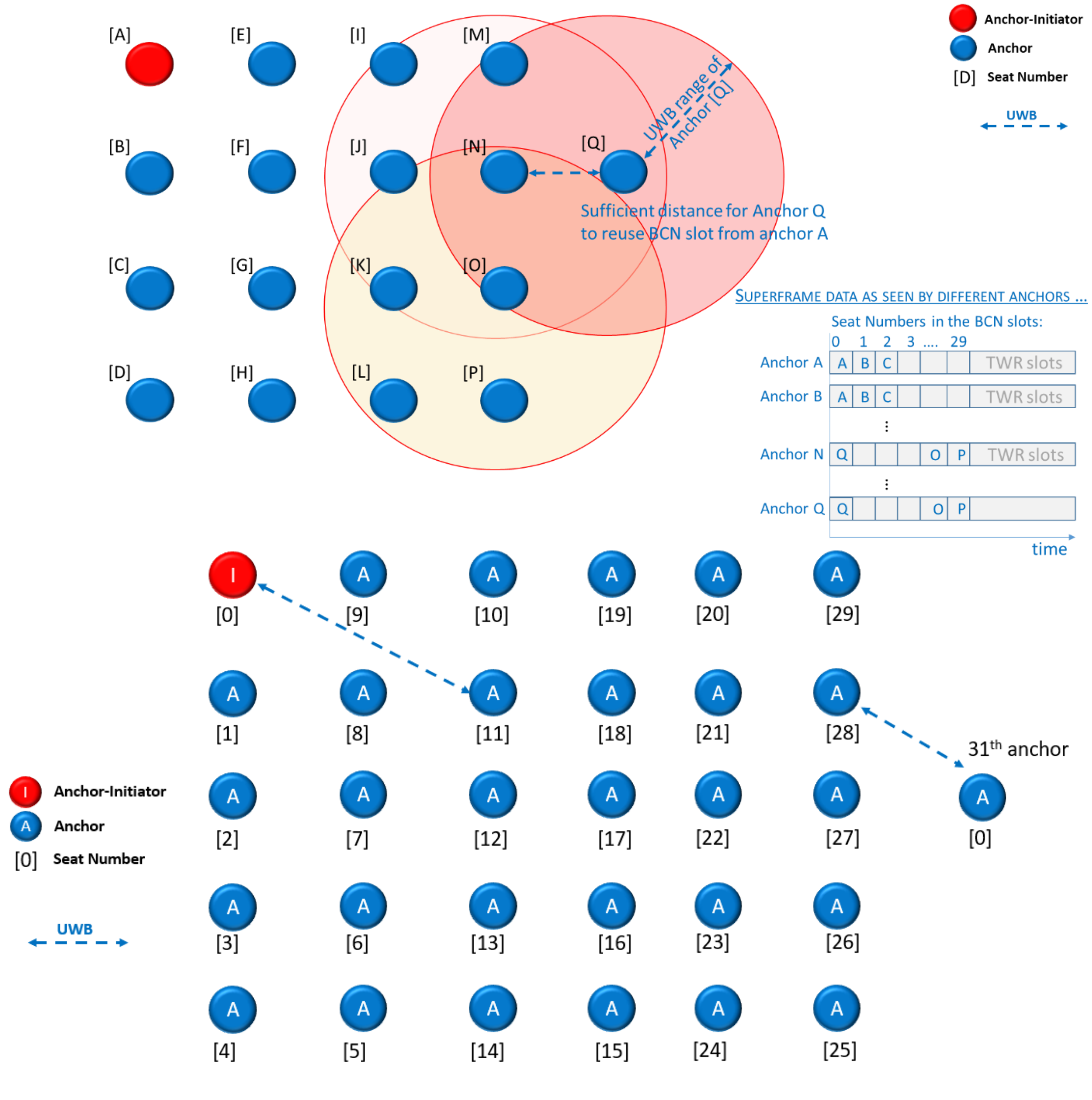


Figure 10:Scaling the DRTLS showing anchor's seat numbers

5.1.1 Scaling Rules

- Each anchor needs to be assigned a seat number between 0 and 29
- No anchor is allowed to hear 2 anchors with the same seat number
- All nodes must be synchronised with the superframe timing of the initiator
- Maximum clock level is 127

5.2 Installation Limitations

5.2.1 Limitation 1: Maximum number of anchor seats is 30

- If the system has 30 anchors or less there are no restrictions on anchor positioning (they can all be in range of each other)
- If a 31st (or more) anchor is required then it needs to re-use a seat number. This can only be achieved if the other anchors are spaced sufficiently far apart, such that no anchor hears two anchors with the same seat.
- Before allowing the 31st anchor to join, all other anchors that are within range of the 31st must confirm to it that a seat number is available (as per 4.2.1)
- For example, in the diagram above, if the anchor 11 is within range of both anchor 0 and the 31st anchor, then the 31st anchor <u>may not</u> use seat number 0 (since that would mean that anchor 11 would be able to see two anchors with the same seat number, i.e. 0)
- The implication of this restriction is that it is best to place anchors reasonably far apart to allow re-use of seat numbers i.e. do not over-populate a single space with too many anchors

5.2.2 Limitation 2: Maximum number of *clock level* is 127

- Each connected node will have its clock derived from the network initiator clock or from its neighboring anchor which is closer to the initiator.
- All nodes within range of the initiator will have a clock level of 1
- Nodes that are in range of nodes with clock level 1, but not in range of the initiator, will have a clock level of 2 and so on
- The highest allowed value of clock level is 127. This means that if a new anchor is trying to join, and it can hear *Beacons* from anchors that are already at clock level 127. It will not be able to join the network.

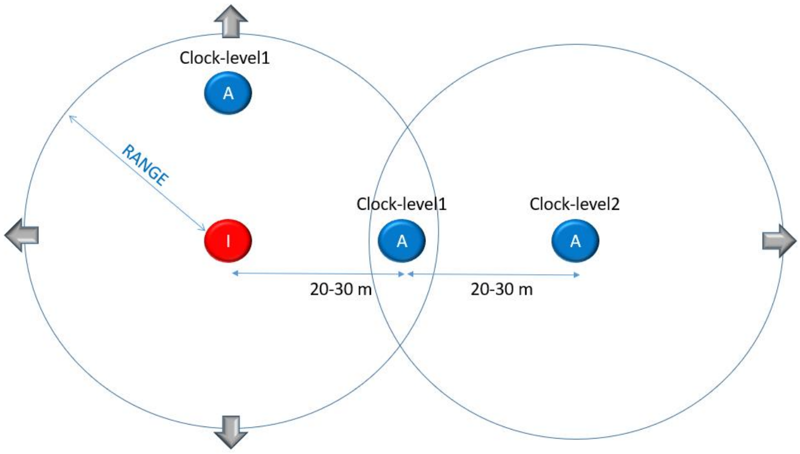


Figure 11:Scaling the DRTLS showing clock levels

5.3 System Expansion – Tags

The system is designed to have a 150 Hz system capacity e.g.

- 15 tags @ 10 Hz (max. location rate)
- 150 tags @ 1 Hz
- 300 tags @ 0.5 Hz
- 9000 tags @ 0.01667 (min. location rate)

Tags are assigned a specific TWR slot and range to up to 4 anchors. If all the slots (there are 15 slots in 1 superframe) have already been allocated to tags, then a new tag will be unable to obtain a slot in which to range. In a similar way that anchors spread over space can reuse seats when out of range of all other users of that seat, over a wider area ranging slots will similarly be able to be reused.

When setting tag update rate, the system installer should consider what level of service will result if it is possible for tags to congregate in one area and should configure the maximum location rate to support that. Otherwise as tags congregate some may become unlocatable as they fail to get access to a ranging slot.

5.4 Coexistence between multiple networks

The PANS FW supports coexistence. This means that multiple networks across the same area share the same air space and do not interfere by overlapping messages. Anchors and tags from a second network will join existing network and share available slots. The two networks must have separate PANIDs (and may use different AES keys), thus the data and location will be kept private. I.e. tags from one network will only be able to range to anchors from the same network (must have same PANID), and IOT data can only be passed between nodes of the same PANID.

5.5 Network Coverage and Expansion

PANS strong scalability allows the user to cover a wide range of areas. Each network will have a different topology based on the geometry in which it will be deployed.

In order to reach the best performance, there are a few points to consider and rules to respects:

- The DWM1001 range is up to 60 meters (point to point) for the best antenna orientation and with ideal line of sight conditions. Within a RTLS system, a tag must range with anchors at various orientation, hence the maximum range is not likely to be achievable. In order to ensure reliable performance across a network, it is recommended to use a reduced range between anchors.
- For improved performance, it is recommended to space anchors 20 to 25 meters apart when placed in a square grid. This is equivalent to a 28 to 35 meters diagonal between opposite anchors.
- For improved performance, it is recommended to consider a maximum range of 30-35 meters for gateways
- In case of non-line of sight, space between node may be reduced to ensure strong signal quality.
- All the metrics above are indicative. It is at the discretion of the user to determine the correct range between nodes to reach a satisfying performance.

The Figure 12 is an example of a grid network deployment for moderate LOS conditions.

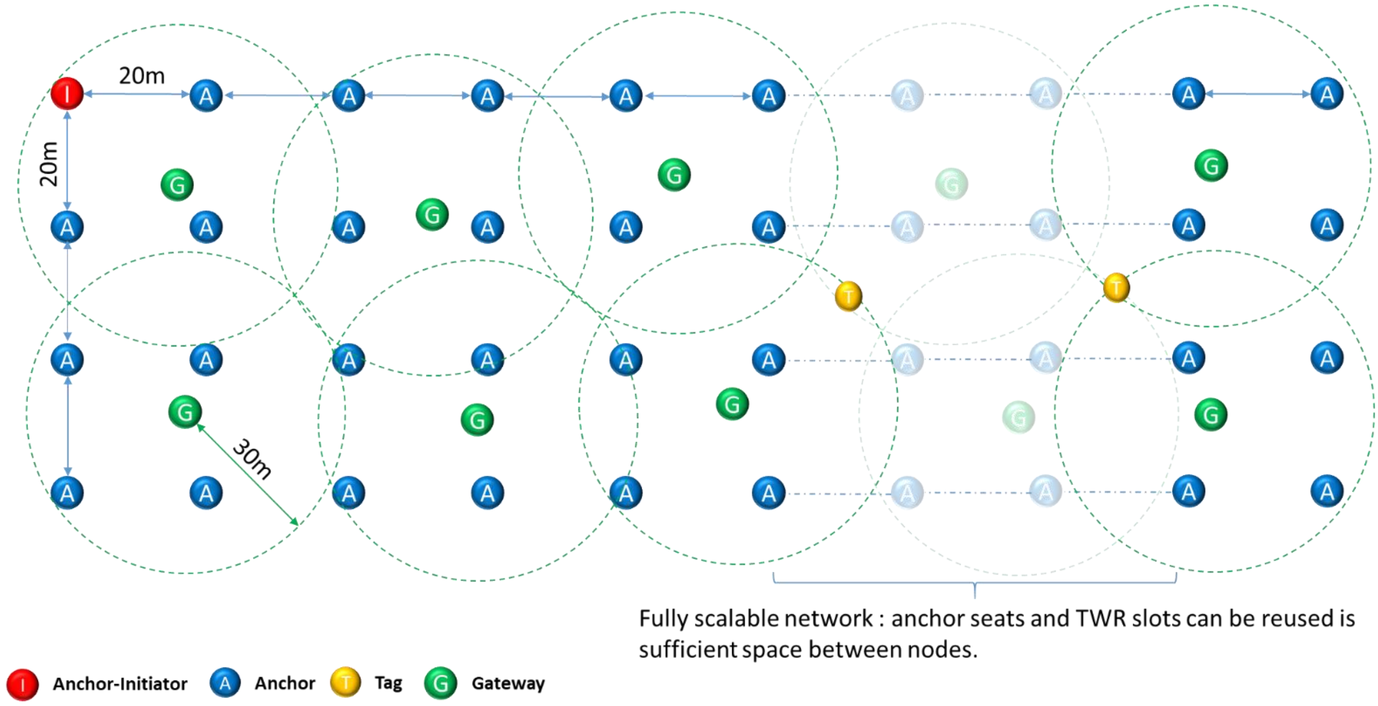


Figure 12: Example of grid network deployment

6 LOCATION ENGINE

The internal TWR location engine is used in tag mode to calculate an estimate of the tag's position using the two-way ranging results and known positions of the anchors selected for the ranging. A location estimate can be calculated with either 3 or 4 range results, i.e. can tolerate missing a response from any one of the four anchors selected for ranging and still calculate a new estimate of the tags location.

The location engine uses maximum likelihood estimation. When it has four ranging results the location engine creates sets of data (4 sets of 3 ranges). The sets are then being used to calculate possible tag positions. The implementation uses cache to speed up the estimation of the positions. If the anchors which are used for calculation has not been changed, the cached value will be used and there is no need to recalculate the initial matrices. The location engine then uses different criteria to choose or to combine them to calculate the estimation of position. The estimated position is verified by using measured distances. The positions which result in shorter distances than the measured are considered less accurate (multipath will result only in longer distances).

The location engine calculates the errors between the estimated positions and the real distance and removes the positions which have high errors. The final position is calculated using the selected estimated positions. The location engine will also report a quality factor (0-100) based on all of these criteria and the calculated errors.

A fixed moving average of the last 3 location results is used for the estimation of the final position.

7 POWER MANAGEMENT STRATEGY

The DWM1001 module FW when operating in default low-power tag mode uses power management strategy to help to keep the system and its components in lowest power mode between the ranging exchanges. The anchors are power efficient but not as power efficient as the tags, as they are continuously on and waiting to be involved in ranging with tags. It is assumed the anchors will be powered via mains power supply.

The DWM1001 onboard operating system (eCos) will register all the tasks with the Power Management function. When all of the OS tasks have completed their operation and are in idle mode, the Power Management will put the MCU into sleep mode without the RTOS tick. While any tasks are running, the RTOS tick is active and the MCU cannot be put into the sleep mode.

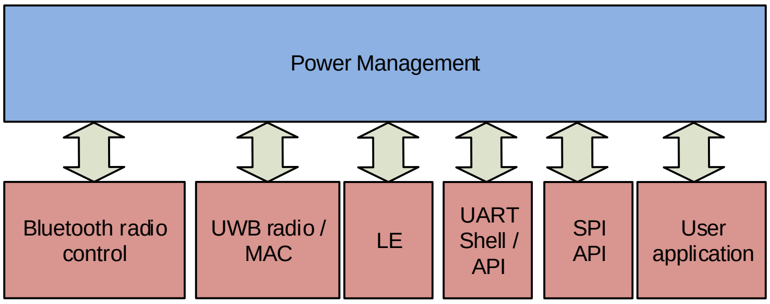


Figure 13: Main DWM1001 system components

7.1 Power Management control of main system components

The Power Management and main system components are shown in Figure 13. The Power Management controls:

- **UWB radio:** The use of DW1000 radio though its driver and the MAC state machine
- **LE:** Location engine operation, it is run as soon as the new range measurements are available, and then goes to idle.
- **Bluetooth radio:** The Bluetooth advertisements are transmitted on DWM1001 power up and when the user presses the Bluetooth wake up button. The advertisements are transmitted for 20 s. When another Bluetooth device (tablet) is connect to the module, the connection is maintained and the module (i.e. tag) will not be able to go to sleep. As soon as the device disconnects, the Bluetooth module will send advertisements for 20 s and if no connection is established, the Bluetooth module will be disabled and its resources released allowing the device to enter sleep state.
- **UART Shell**: Once *shell* is enabled the MCU will not go into sleep mode until the user runs the “quit” command, and disables the *shell*. That means the *shell* and its resources are released from the Power Management.
- **UART API:** Once UART API is enabled (a communication was initialised via UART from a host device – see [5]), The MCU will not go into sleep mode and the host needs to control the MCU sleep via an API call.
- **SPI API:** Once SPI API is enabled (a communication was initialised via UART from a host device – see [5]), The MCU will not go into sleep mode and the host needs to control the MCU sleep via an API call.
- **User Application:** A user can add own application into the DWM1001 FW. This is described in detail in [5]. This application has two options to register with the Power Management and influence when the MCU is sleeping/going into low power mode:

- By registering to receive location data - on reception of location data via the callback, a Power Management automatically waits for User Application to notify it via the API call (when finished) and then the Power Management can put the MCU into sleep mode if no other tasks are pending.
- By registering with the Power Management task and notifying it using API calls

7.2 Wake-up sources

To wake up the tag from low power (sleep) mode a number of signals are used:

- **RTC**: The RTC is used to wake-up the UWB radio and its MAC so that the device can be ready for the next ranging exchange.
- **Accelerometer**: If accelerometer detects movement it will wake up the device and the tag will change to using the non-stationary location rate.
- **UART RX GPIO:** The host can wake up the device with UART RX GPIO so that it can communicate with it via the UART APIs.
- **SPI CS GPIO:** The host can also use SPI CS signal to wake up the device.
- **User button:** The DWM1001 GPIO2 (e.g. if connected to a button e.g. DWM1001 – DEV) can also be used to wake up the device.
- **BLE:** If BLE is enabled with Low Power Tag mode, then the tag's UWB radio will be enabled once the tag is in BLE range of anchor nodes and disabled once the tag leaves UWB network.

7.3 Two location update rates

To help reduce power consumption the tag has an option of two location update rates, the nominal and stationary. The onboard accelerometer is used to detect when the device is stationary and then the stationary rate will be used. If the accelerometer is disabled then only the nominal location rate will be used.

8 Memory Usage and Firmware Update

The DW1001 comes preloaded with DRTLS firmware, which enables the building of a fully functioning RTLS without any further firmware change, however if the need arises it is possible to reprogram the module firmware in its flash memory. This process, called firmware update, is described in detail in DWM1001 Firmware User Guide [5].

The flash memory structure is shown in Figure 14. The area labelled FW2 contains the main functional block and application and area FW1 is just used for the update process.

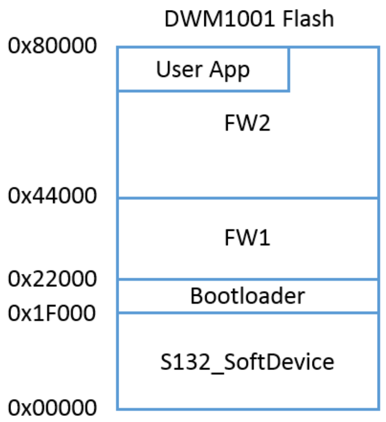


Figure 14: DWM1001 flash address map

8.1 Firmware Update

When the RTLS network is forming, the initiator anchor specifies the firmware version necessary for the network. When automatic FW update is enabled, any devices wishing to participate (join) the network must have the same firmware (version number and the checksum). If a new device does not have the correct firmware it will be updated as per the sub-sections below.

8.2 Firmware update initiated over Bluetooth

If one wants to update the entire network to a new firmware image while the network is operational, it is sufficient to just update the initiator via Bluetooth. The initiator will then propagate the new firmware to all of the other devices over the UWB radio link automatically.

Note, as the initiator is updated first, it will restart the network and as each device re-joins the network its firmware will be updated. Thus, during the FW update the nodes which are performing the updated will be “offline”.

8.3 Firmware update via UWB

When automatic FW update is enabled, every device wanting to join the network needs to have the same firmware version. Each new joining device listens to the *Almanacs* to obtain the system hardware and firmware information. If a firmware update is needed the new device will choose a nearby anchor with free TWR/data slot and will send a *Firmware Update Data Request* message (9.1.6).

The anchor which received the request to supply the firmware, checks the request and if no firmware update process is running will start sending the firmware update data to the requesting anchor over UWB.The firmware data messages are broadcast so any node in firmware update state can receive and process the data.

The receiving node stores the new firmware image data in an intermediate buffer and when the buffer contains the size of internal flash page size, it will be written to the internal flash. Note, during the firmware update, Bluetooth SoftDevice is disabled because writing to internal flash via SoftDevice is very slow (up to 500 ms per page). When the SoftDevice is disabled, a page can be written within ~120 ms.

If the receiving device misses some data (due to failed frame reception), it will request the same data again starting the offset where it stopped in the previous attempt. The cycle will repeat until the requesting node would receive all firmware data. Then it will run a checksum control of the whole firmware and if the checksum values is correct, it will enable the non-volatile register to boot from the other firmware (FW1/FW2) reset. If the checksum failed, it will repeat the whole process again.

Firstly the unit will be running from FW2 and will update FW1, then it will reset and update FW2.

8.4 Manual firmware update

When automatic FW update is disabled, then the user can individually update firmware in a device via Decawave DRTLS Manager or via the SWD connection.

9 Appendix: Frame Formats

9.1 IEEE 802.15.4 frame

The system uses standard frames as defined by IEEE 802.15.4 on the MAC layer. The standard defines multiple frame types, only the data frame is used in this system. Table 2 shows the data frame structure.

Table 2: IEEE 802.15.4 Data Frame Structure

Field	Octets	Description
Frame control	2	As defined in the IEEE 802.15.4
Sequence number	1	Modulo 256 sequence number
PAN ID	2	PAN ID
Destination address	2	Address of destination node, or 0xFFFF if this is a broadcast frame (e.g. *Group Poll* message)
Source address	2	Node's own address is fixed and is the lower 16-bits of the generated 64-bit address (derived as 0xDECA + 28 bits of MCU Unique ID + 20 bit of DW1000 Part ID)
Payload	1-53	Note: the first byte of the payload always denotes the frame type which is followed by the message content as specified in tables below. Message ID codes: 0x10 - UWBMAC_FRM_TYPE_BCN 0x11 - UWBMAC_FRM_TYPE_SVC 0x12 - UWBMAC_FRM_TYPE_CL_JOIN 0x13 - UWBMAC_FRM_TYPE_CL_JOIN_CFM 0x18 - UWBMAC_FRM_TYPE_POS 0x21 - UWBMAC_FRM_TYPE_FWUP_DATA_REQ 0x22 - UWBMAC_FRM_TYPE_FWUP_DATA 0x23 - UWBMAC_FRM_TYPE_ALMA 0x63 - UWBMAC_FRM_TYPE_DL_IOT_DATA 0x65 - UWBMAC_FRM_TYPE_UL_IOT_DATA 0x6A - UWBMAC_FRM_TYPE_BN_BCN 0x30 - UWBMAC_FRM_TYPE_TWR_GRP_POLL 0x31 - UWBMAC_FRM_TYPE_TWR_POLL
Frame check sequence	2	DW1000 automatically generated FCS

The payload of the data frame (see Figure 15) can contain one or more of the messages specified in the sub-section below. Generally, only one message is sent per data frame, the only exception being the Beacon message, which can be followed by an extra message (e.g. Join Confirmation or Position messages).

The IEEE MAC header and payload is shown in Figure 15.

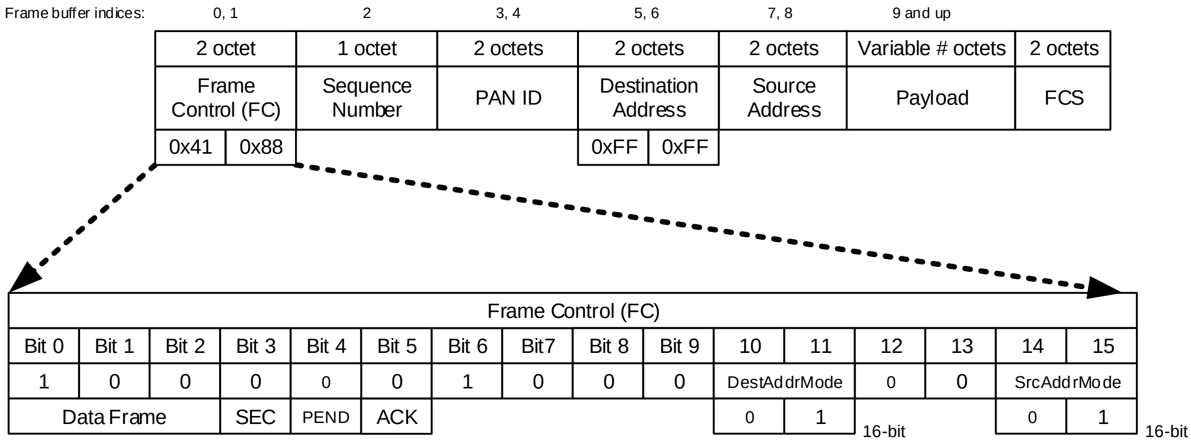


Figure 15: IEEE 802.15.4 MAC frame format, with e.g. broadcast address and

9.1.1 Beacon message

These messages are sent by the anchors in the BCN slots in the superframe. These messages have the IEEE 802.15.4 data frame format with payload specified in the following table. The *Beacon* messages contain information about the current superframe and network slot usage (i.e. which TWR slots are currently used by tags/anchors).

Payload Field	Octets	Description
Message ID	1	0x10 - UWBMAC_FRM_TYPE_BCN
Session ID	1	Random number which the Anchor Initiator generates during network initialisation. All joined anchors must have the same Session ID
Cluster flags	2	Bit field where following flags can be set: • AUTOPOS - auto-positioning is in progress throughout the network, can be used to auto-locate the anchors in range of the initiator • DAT_DL - down-link data will be sent in this superframe • SVC_DL - service slot in this superframe is down-link • SVC_UL - service slot in this superframe is up-link • INIT - node is initiator • BH - node can hear at least one nearby BN • EXT - when set, there is another message following the beacon message, in the payload of the same IEEE 802.15.4 data frame; length of the extra frame is specified in the payload immediately following the beacon message • NBR_SYNC - node is synchronized to the clock from the neighbor network. The highest 7 bits are used for indicating the clock level
SF number	2	Superframe number used for synchronisation.
Cluster slot number	1	Cluster slot number ~ seat number
Cluster map	4	Bitmap indicating occupied seats visible by the sending anchor (a set bit indicates occupied seat, e.g. 0x000081 means seats 7 and 0 are occupied). Mask of the full cluster size is 0x3fffffff, i.e. 30 seats.
Data slot map	2	Indicates which TWR/IOT data slots of the sending anchor are occupied
NONCE	10	Current network NONCE used for encryption/decryption

9.1.2 Join request message

These messages are sent by the anchors, during the Service interval (SVC slots) inside the superframe, as a request to join the network. These are the IEEE 802.15.4 data frames with message payload specified in the following table. These are sent to networked anchors as a request to join the network.

Payload Field	Octets	Description
Message ID	1	0x12 - UWBMAC_FRM_TYPE_CL_JOIN
Hardware version	4	0xDE00002A = DWM1001 hardware version.
Firmware version	4	0x01010001 = 01.01.00.0.1 = Major.Minor.Patch.Reserved.FirmwareVariant
Firmware checksum	4	Firmware checksum - CRC32
Options	4	Bitmap indicating node capabilities
Cluster seat	1	Requesting cluster seat

9.1.3 Join confirmation message

These messages are sent by the anchors after a join request is received. These messages are sent following the *Beacon* message in the same IEEE 802.15.4 data frame (see the EXT flag in the *Beacon* message) and have the format specified in the following table. They are sent in a response to join request to allocate a seat to the anchor trying to join the network.

Payload Field	Octets	Description
Message ID	1	0x13 - UWBMAC_FRM_TYPE_CL_JOIN_CFM
Address	2	Locked address of the joining node
Cluster lock	1	Lock counter (decrementing). 0 means the next *Beacon* will not contain join confirmation unless a new request is received
Cluster seat	1	Confirming allocated seat number or 0xff indicating no seat has been allocated

9.1.4 Almanac message

These are sent by the networked anchors, during the Service interval (SVC slots) inside the superframe. These are IEEE 802.15.4 data frames with payload specified in the following table. Each superframe can only carry a single *Almanac* message, so each of the 30 anchors of the cluster takes its turn to send its almanac once every 120 superframes according to its seat number. They provide the node FW versions and node's capabilities.

Payload Field	Octets	Description
Message ID	1	0x23 - UWBMAC_FRM_TYPE_ALMA
NONCE	10	Network NONCE used for encryption/decryption in the next period of NONCE switch
Flags	1	Special flags, e.g. firmware update force.
Hardware version	4	Hardware version of the sending node
Firmware version	4	Firmware version of the sending node
Firmware 1 size	4	Firmware 1 size of the sending node
Firmware 2 size	4	Firmware 2 size of the sending node
Firmware 1 checksum	4	Firmware 1 checksum of the sending node
Firmware 2 checksum	4	Firmware 2 checksum of the sending node
Node ID	8	Complete 64-bit address of the sending node
Node option	4	Bitmap indicating node capabilities

9.1.5 Service message

The Service messages are used for sending service commands and status between the networked devices. E.g. collision reports. These are sent in the SVC slots. These are IEEE 802.15.4 data frames with payload as specified in the following table.

Payload Field	Octets	Description
Message ID	1	0x11 - UWBMAC_FRM_TYPE_SVC
Code	1	Service code
Argc	1	Number of argument octets
Argv	0-14	Arguments

9.1.6 Firmware Update Data Request message

These messages are sent by devices which need to update their FW. The request is sent to one of the networked anchors. These are IEEE 802.15.4 data frames with payload as specified in the following table.

Payload Field	Octets	Description
Message ID	1	0x21 - UWBMAC_FRM_TYPE_FWUP_DATA_REQ
Flags	1	Flags
Update period	2	Update period in ms
Addr16	8	Max four 2-byte addresses of firmware providers
Offset	4	Requesting data offset
Firmware size	4	Requesting firmware size
Firmware checksum	4	Requesting firmware checksum

9.1.7 Firmware Update Data message

These messages contain the Firmware image data and are sent from the networked anchor to the device that has requested the update. These are IEEE 802.15.4 data frames with payload as specified in the following table.

Payload Field	Octets	Description
Message ID	1	0x22 - UWBMAC_FRM_TYPE_FWUP_DATA
Flags	1	Flags
Slot data map	2	Map indication the next slot occupation
Offset	3	Firmware offset of the sending data
Length	1	Length of the sending data
Buffer	1-44	Content of the sending firmware buffer

9.1.8 Position message

The *Position* message is sent as a part of the extended *Beacon* message, it contains the anchor coordinates. The message payload is as specified in the following table.

Payload Field	Octets	Description
Message ID	1	0x18 - UWBMAC_FRM_TYPE_POS
X	4	X coordinate (of the anchor position), m
Y	4	Y coordinate (of the anchor position), m
Z	4	Z coordinate (of the anchor position), m
padding	4	Padding for encryption

9.1.9 Group Poll message

This message is sent from a device that wants to initiate a TWR exchange. This will typically be a tag but it can also be an anchor (e.g. during auto-positioning process). These are IEEE 802.15.4 data frames with payload as specified in the following table. This is sent as a broadcast so all devices can receive it.

Payload Field	Octets	Description
Message ID	1	0x30 - UWBMAC_FRM_TYPE_TWR_GRP_POLL
Flags	2	Bitmap indicating the anchors to range with. Anchor uses this bit map to figure out if its seat corresponds with the indicated bit. If it does, it will continue to look for its address in the Address field below else the frame will be ignored.
Update period	2	Requesting update period in ms
Address	8	Four 16-bit addresses of the anchors to range with
Sequence number	1	TWR sequence number
Stationary flag + Quality factor	1	Position quality factor (bits 1-7), stationary flag in bit 0 (if 1 then device is stationary).
X	4	X coordinate in meters (of the last calculated position)
Y	4	Y coordinate in meters (of the last calculated position)
Z	4	Z coordinate in meters (of the last calculated position)
Padding	2	Padding for encryption

9.1.10 Response message

An anchor receiving a *Group Poll* nominating it as one of the four addressed anchors will respond with a *Response* message. This is a part of the single-sided TWR protocol. These are IEEE 802.15.4 data frames with payload as specified in the following table. These are addressed to the particular device which sent the *Group Poll*.

Payload Field	Octets	Description
Message ID	1	0x31 - UWBMAC_FRM_TYPE_TWR_POLL
Flags	1	TWR_FL_ECOLL_RSP- Collision detected at TWR Responder TWR_FL_ETYPE_RSP - Incorrect frame type at TWR Responder TWR_FL_IOT_DATA – Downlink IOT data will follow last Response message TWR_FL_ACK – acknowledgement that the ranging AN has a BN in-range and that the location data in the Group Poll is likely to be received by the BN TWR_FL_BH - responder indicates that a BN is in range TWR_FL_SL_CFM - Next slot confirmation
Slot map	2	Map indicating the next occupation of the slot
GP timestamp	4	Receive timestamp of Group Poll
R timestamp	4	This message (Response) TX timestamp
NONCE	10	Network NONCE for the next update cycle

9.1.11 Bridge Node Beacon message

These are sent by the bridge nodes in the Bridge node Beacon message slots. These are IEEE 802.15.4 data frames with payload as specified in the following table.

Payload Field	Octets	Description
Message ID	1	0x6A - UWBMAC_FRM_TYPE_BN_BCN
BN cluster map	4	Bitmap indicating occupied BN cluster seats visible by the sending anchor (the bit set means occupied seat, e.g. 0x000081 means seats 7 and 0 are occupied). Mask of the full cluster size is 0x3fffffff, i.e. 30 seats.
Address	2	Address of the joining bridge node
BN cluster lock	1	Lock counter (decrementing). 0 means the last one and the next message will not contain join confirmation unless a new request is received
BN cluster seat	1	Confirming allocated BN cluster seat number or 0xff indicating no seat has been allocated
Count	1	Number of tag addresses (specified below)
Tag address	2xTID	A list of 2 byte tag's address (max 15)

9.1.12 IOT Data message

A message containing the tag's or anchor's IOT data payload sent to the routing anchor (uplink) or sent to the tag or anchor (downlink) by the routing anchor. These are IEEE 802.15.4 data frames with payload as specified in the following table.

Payload Field	Octets	Description
Message ID	1	0x63/0x65 - UWBMAC_FRM_TYPE_DL/UL_IOT_DATA
ID	2	Node address (for UL can be either TN/AN; for DL it is the address of the receiving node)
Flags	1	Bit 0 - Update rate: 0=Nominal 1=Stationary Bit 1 - Overwrite the previous written data Bit 2 - Set if in Tag mode Bit 3 - Link quality indication: 0=poor 1=good (for downlink purpose, used only between the BN and KM) Bit 4-7 - Data type
Update rate	2	Update rate of the next cycle
Data length	1	IOT data length
IOT payload	1-34	1-34 bytes – IOT data payload

10 REFERENCES

10.1 Listing

Reference is made to the following documents in the course of this document:

Table 3: Table of References

Ref	Author	Version	Title
[1]	Decawave	Current	DWM1001C Product Brief
[2]	Decawave	Current	DWM1001-DEV Product Brief
[3]	Decawave	Current	MDEK1001 Product Brief
[4]	Decawave	Current	MDEK1001 User Manual
[5]	Decawave	Current	DWM1001 Firmware User Guide
[6]	Decawave	Current	DWM1001_Gateway_Quick_Deployment_Guide

11 Document History

11.1 Revision History

Table 4: Document History

Revision	Date	Description
2.1	28 September 2020	Updates per PANS R2.1
2.0	29 March 2019	Updates per PANS R2 Software
1.3	7th July, 2018	Logo Change
1.2	8th May, 2018	Update for Release 2 (alpha)
1.1	7th February, 2018	Minor updates
1.0	18th December 2017	Release for publication

12 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.