**Development in Software Aspect of the Network Communication System for Object Localization**

1. Abstract

The aim of the project is to design a wireless system for object localization based on the UWB scheme. The previous strategy is to set the positions in the host computer program, then place the base stations to the set positions in the field. After that, we introduce tags into the system and calculate the positions of tags by the distance measurements between base stations and tags. The main problem is that there will be an inevitable mismatch between the actual positions of the base stations and the set positions due to manual distance measurements. Our research proposed a new improved method. We removed the distinction between base stations and tags, which means, arbitrarily placing the UWB modules in the field first, then initializing their positions in a computer program by letting the modules themselves measure distances between one another. This method avoids the error of placing base stations so that it can increase the accuracy of the localization system. However, the new scheme requires a different logical implementation of the functioning of the UWB modules and the communication between different modules. So we do our improvement and modification in the software aspect to make the UWB modules’ functioning meets our requirements.

1. Introduction of UWB Scheme

UWB positioning technology is a kind of indoor positioning technology, which has the advantages of strong penetration, low power consumption, good anti-multipath effect, high security, low system complexity, and high positioning accuracy. Thus, UWB positioning technology can be applied to indoor positioning tracking and navigation of stationary or moving objects and people and can provide very accurate positioning information.

UWB positioning technology uses TOF (time-of-flight) ranging, which is a two-way ranging technique that uses the time of flight between two transceivers to measure the distance between nodes. The module generates an independent timestamp from the start. The transmitter of module A transmits a pulse signal of request nature at its timestamp a1, and module B transmits a signal of response nature at moment b2, which is received by module A at its timestamp a2. The equation then calculates the time of flight of the pulse signal between the two modules and thus determines the flight distance. Because the TOF-based range measurement method is linear with distance in a line-of-sight environment, the measurement results will be more accurate.

To execute the above process, we need to do clock synchronization and execute TDOA ranging (see figure 1).

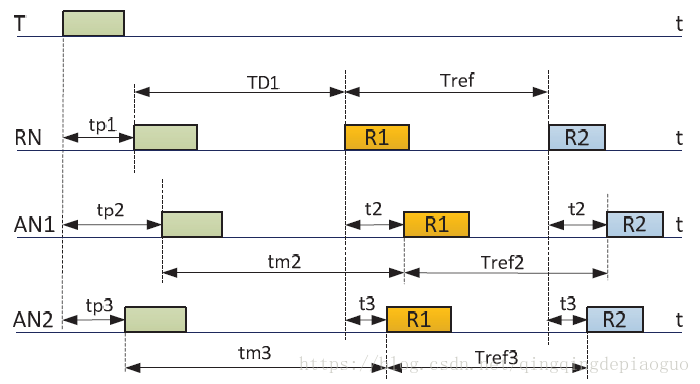


Figure 1.

Combined with the figure above, the steps are as follows:

1. The tag T is broadcast once in the positioning space (any data can be sent);
2. The base station and the reference base station in the positioning space are in the receiving state and can receive the broadcast data in step (1), so each device will trigger to generate a receiving timestamp TP1, TP2 and TP3;
3. After the reference base station receives the broadcast, it converts the working mode to the transmitting mode, and broadcasts R1 content once after a fixed delay (TD1, set by itself) (the broadcast content can be distinguished from the label T, and any data can be sent); After the base station receives the broadcast of (2), the working mode remains unchanged, or the base station is in the receiving mode;
4. In step (3), after the base station broadcast data is referenced, all base stations will receive the content of R1, thus triggering the generation of the receiving time stamp. T2 and T3 are the arrival time. After the base station and the reference base station have been deployed in the positioning space (the relative distance remains unchanged), T2 and T3 are known;
5. After the reference base station broadcasts R1 content, it broadcasts R2 content again after a fixed delay (Tref);
6. All the stations will pick up the R2 broadcast again; End of positioning.

For the base station, a timestamp can be generated when receiving the broadcast, so Tm2, Tref2, TM3 and Tref3 are known measurements. T2 and T3 are also known, because after the base station and the reference base station are deployed, the relative distance and arrival time remain unchanged.

The TDOA calculation process is as follows:

TDOA21 = tp3 - tp2 = tm2 - tm3 + t3 -t2

For tm2 and tm3, the measured values cannot be directly used, and need to be calibrated:



Where tm2m is the measured value;

Trefn also needs to be filtered:



At this point, the algorithm ends.

The list shows the overview of the UWB module used in this project.

|  |  |
| --- | --- |
| YCHIOT UWB Mini 4 module series | |
| Module picture |  |
| Main control chip | STM32F103RCT6 microcontroller |
| Peripheral circuits | DWM1000 module, power module, LED indicator module, reset circuit, etc. |

The following list shows the hardware parameters of the module.

|  |  |  |  |
| --- | --- | --- | --- |
| Basic Parameter | | Wireless Parameter | |
| PCB Craft | 4 layers - epoxy resin | Communication Rate | 110 kbit/s, 850 kbit/s, 6.8 Mbit/s |
| VAB-IO | Micro-USB(5.0V)/connection  column | Operating Frequency | 3.5 GHz ~ 6.5 GHz |
| Communication Interface | Micro - USB (5.0 V)/serial port  (3.3 V TTL) | Work Channel | 6 |
| Download Port | SWD (VCC SDIO SCK  GND) | Transmitting Power | -35dbm/MHZ ~ -62dbm/MHZ |
| Host Controller Driver | STM32F103RCT6(64pin) | Max Packing Length | 1023 bytes |
| External Crystal | 8Mhz | Communication Distance | 30m |
| PCB Size | 35mm \* 24mm | Data-dependent Jitter | Typical ±10cm, general occlusion ±30cm |

The indoor positioning function of UWB is very similar to the principle of satellite, that is, by arranging several positioning base stations with known coordinates indoors, the person who needs to locate carries a positioning tag, the tag emits pulses at a certain frequency, continuously ranges with several base stations, and determines the location of the tag through a certain precise algorithm.

1. Introduction of the Program Embedded in the UWB Modules

In this chapter, we will briefly discuss the source code of the program embedded in the UWB modules, the DecaRange RTLS ARM application, covering the structure of

the software and the operation of the ranging RTLS demo application particularly the way the range is calculated.

Below shows the layered structure of the DecaRange RTLS application, giving the names of the main files associated with each layer and a brief description of the functionality provided at that layer (See Figure 2).

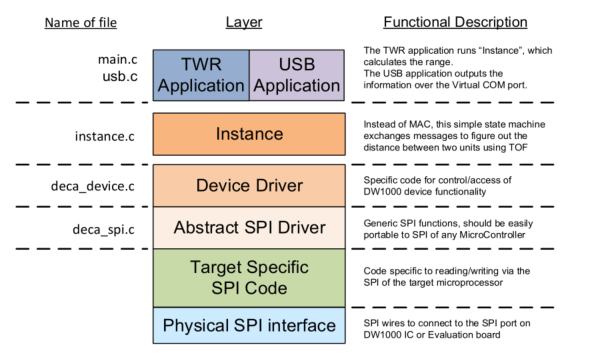


Figure 2: Software layers in DecaRange RTLS ARM application

Regarding Figure 1, the identified layers are described in more detail below.

* 1. Target Specific Code

The low-level ARM specific code can be found in \src\platform\ – the two files port.c and port.h define target peripherals and GPIOs which are enabled and in use i.e. SPI1 for SPI communications with DW1000, SPI2 for SPI communications with LCD, other GPIO lines for application configuration and control.

* 1. Abstract SPI Driver – SPI Level code

The file deca\_spi.c provides abstract SPI driver functions openspi(), closespi(), writetospi() and readfromspi(). These are mapped onto the ARM microcontroller SPI interface driver.

* 1. Device Driver – DW1000 Device Level Code

The file deca\_device\_api.h provides the interface to a library of API functions to control and configure the DW1000 registers and implement functions for device level control. The API functions are described in the “DW1000 Device Driver Application Programming Interface (API) Guide” document.

* 1. Instance Code

The instance code (in instance.c) provides a two-way ranging RTLS demonstration application. The two-way ranging RTLS demo application is implemented by the state machine in function testapprun(), called from function instance\_run(), which is the main entry point for running the instance code. The instance runs in different modes (Tag or Anchor) depending on the role configuration set at the application layer. The Tag and Anchor modes operate as a pair to provide the two-way ranging demo functionality between two units. In TREK single tag ranges up to 4 anchors and then a separate DecaRangeRTLS PC application can then use the ranges to calculate tag’s location relative to anchors’ and display on the GUI.

3.4.1. Operating mode – Tag

Once powered on the tag unit will try and range to 4 anchors and then go to sleep. After a period (superframe/scheduling period, T sf ) it will wake up and range to 4 anchors again. The tag ranges to all 4 anchors simultaneously. It sends a Poll as a broadcast message (destination address is set to 0xFFFF), receives any responses, and then sends a Final. If no response from anchor #0 is received the Final message will not be sent.

Figure 2 gives an overview of the superframe structure.

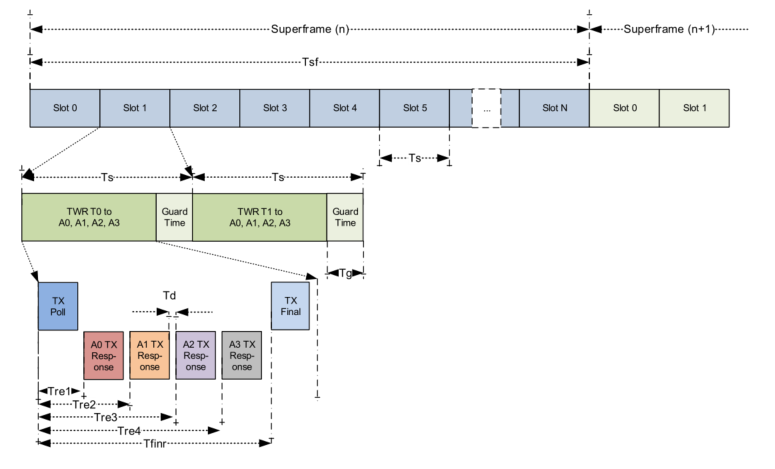


Figure 3: Tag TWR RTLS time profile

To support multiple tags (without interfering), we are using a TDMA approach. In our TREK example modes there are 10 slots in the superframe. Each slot is 28 ms in 110 kbps modes and 10 ms in 6.81 Mbps modes, and, as we assign one slot to each tag a maximum of 8 tags can be supported. Two slots are reserved for anchor-to-anchor ranging.

The anchor #0, is responsible for assigning and maintaining tags into their own slots.

3.4.2. Operating mode – Anchor #0

Firstly, Anchor #0 operates to assign activity slots to each of the tags so that their ranging exchanges do not mutually interfere. Anchor #0 does this by including a sleep time adjustment value every time it responds to a tag’s ranging attempt. This sleep time adjustment is calculated to position each of the tags to wake-up in a separate slot, based on their address (#0 to #7).

Secondly, Anchor #0 reports the results of all ranging exchanges via its USB port. It does this by listening for and receiving time-of-flight results that the other anchors embed in their response messages, and gathering these along with its own calculated TOF results, before sending the a full set of ranging results for each tag to and attached PC via its USB port.

Anchor #0 also starts anchor to anchor ranging, which is used for auto-positioning feature. Last two slots of the superframe are used for this. Anchor #0 initially ranges to anchors #1 and #2 and then anchor #1 ranges to anchor #2. The resultant six ranges are outputted over the USB so that the anchor positions relative to each other can be calculated.

3.4.3. Operating mode – Anchors #1 and #2

Anchor #1 and #2 are involved in tags to anchor ranging and also anchor to anchor ranging.

3.4.4 Operating mode – Anchor #3

Anchor #3 is only involved in the tag to anchor ranging. It ignores any anchor to anchor ranging messages.

3.4.5 Range Result

The ranging results are output via the USB port. Tags will report any ranges results it receives from the anchors it ranges with, as returned to it by the anchor’s Response message.

All the anchor will report ranges they calculate for the tags that range with them, but also any range reports they receive from other anchors (all will receive TOFs inside the response messages). No frame filtering is used so any messages on the air can be received.

3.4.6 Ranging Method

The ranging method uses a set of three messages to complete two-round trip measurements from which the range is calculated. As messages are sent and received the DecaRangeRTLS ARM application retrieves the message send and receive times from the DW1000. These transmit and receive timestamps are used to work out a round trip delay and calculate the range. Figure 3 shows the arrangement and general operation of the two-way ranging as implemented by the DecaRangeRTLS ARM application.

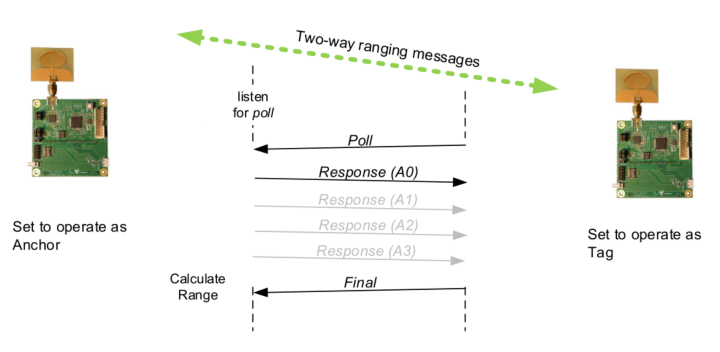


Figure 4: Two way ranging in DecaRangeRTLS ARM

* 1. Top level Application code

The top-level application (main.c) contains the main entry point for the DecaRanging ranging demo application and also all the user interface code.

1. Problems Encountered and Solutions in the New Software Program Implementation

Our main goal is to remove the distinction between base stations and tags, so in our program, the problem is that there is no need to divide the modules into “base stations” and “tags”. Notice that in the software layers in DecaRange RTLS ARM application, the instance layer provides a two-way ranging RTLS demonstration application implemented by the state machine in function testapprun(). The instance runs in different modes (Tag or Anchor) depending on the role configuration set at the application layer.

To solve the problem, we first set all the modules to be the anchors at the application layer, then change the TOF ranging process. Originally, the system do the poll-response process by letting a tag to pole and 3 or 4 anchors respond in each time slot in the superframe. Now in our new scheme, denote modules A0, A1, A2, and so on. In our new superframe, we invoke the modules by the given order. For slot 0, A0 is the poller and gets the response from A1 so we have the distance A0A1. For slot 1, A1 is the poller and gets the response from A2 and A3 so we have the distances A1A2 and A2A3. Similarly, at the next time slot, we let the newly invoked module be the poller, the other modules invoke before the current slot to be the responder, and get the distance between the poller and responder. After the superframe ends, we can calculate the distance by these distances.

1. Discussion

Overall, we have designed a new program to do the positioning using only one kind of module. Thus, we simplify the system and improve the accuracy. Notice that because in our new scheme, there is no need to have fixed base stations, which means this is a decentralization system and each module can be integrated into the moving robots and do the real-time positioning while moving, which improves the utilization of the modules.

However, considering the situation of the increasing of responses in multiple modules, we may prolong the period of each slot for poll-response. So there is a trade-off between the refresh rate and accuracy of the positioning system in the implementation process.

Here, we propose the timeline for summer term:

June: Field test. During tests, improve the program according to the test results therefore the performance of the system is better.

July: Summarize the progress and achievement.