Start-Up Sequence in the PTT project

Ida Elders

Norwegian University of Life Sciences
Department of Mathematical Sciences and Technology
Master Thesis 30 credits 2011

Table of Contents

Table off Figures

1 Abstract

GPS positioning is a well-known positioning system and used in several applications. However, these signals are sent with low power and do not penetrate walls and have problems in urban areas with tall buildings. On the other hand, GPS shows advantages compared to other technologies when used outside. Since timing is one of the most important issues when it comes to positioning, some of the timing tools from the GPS can be helpful in other positioning systems.

Using terrestrial communication such as television signal, which is designed to penetrate into people's houses, makes it possible to positioning indoor and in urban areas as well. Still, this is not designed for positioning use, and there are more possible positioning errors. Depending on what the positioning is used for, the accuracy with TV signals may be enough, but when used in violence alarms and for firefighters there is a need for much higher accuracy. It is also important that the altitude can be decided.

The Pseudo Television Transmitters (PTT) system has been designed for high accuracy positioning indoor, where it is crucial to reduce errors. Especially the timing and start up order is important and one of the technical things we can control. The rest of the error is often caused by multipath, and bad geometry on the transmitters. The PPT system consist of GPS receivers who give the absolute position for the transmitter, baseband transmitters with special designed antennas which broadcast the TV signals, Software signal generator, filters, and amplifiers. There is an interface controller which connects and controls all the parts and which the communications goes through.

In the system discussed and the applications developed in this paper, GPS has been used as a helpful part of the system. The system needs to use GPS for absolute positioning of the baseband receivers, and provides an in common clock for the total system. It also provides a 1 Pulse per Second (1PPS) signal which is used as a reference to make the system work together in the best way.

This paper has focused on the interface controller which is the communication between the parts in the system. One of the most important parts is to optimize the timing and have a start-up sequence where the accuracy errors are reduced as much as possible. This paper is written at and for Rosum Corp, and development done is used in a system which will run live, already sold to a customer.
To be able to make an optimized interface controller, a hardware unit is chosen regarding to the specification, and the start-up sequence developed, implemented and written in the programming language C.

2 Acknowledgement

This assignment is done on behalf of Rosum cooperation and together with the PTT team led by PhD Guttorm Opshaug at Rosum. I would like to thank the full team for the help and support and for the big effort done in the total project. I also would like to thank them for welcoming me to their company and including me in all the tasks done.
I will especially like to thank Guttorm Opshaug for the idea on PTT-project and the overall design and for being a great leader in a project driven by a customer and changing needs. Guttorm also was also a big part of the practical tests.

I would also like to thank MSc Roshan Baliga for all the help on the practical test, the development of the applications for the Software signal generator (USRP) and keeping me updated on the hardware parts chosen.
I would like to take the customer for its involvement and for letting us lend a developer for the GPS positioning part, PhD student Gabriel Wong. Gabriel did a great effort and we had a good cooperation to make our program play along, since they both had to be implemented on Interface controller. Gabriel's work is including in my work, but will only be referred to as Gabriel's work.

I would also like to thank Pål Johan From for being my adviser along the way, helping me find the right approach and where to put focus and make more depth in the paper. I will also Including Jan Kåre Bø for the great effort and always believing in me. He was also a great help, making it possible to stay with Rosum in the USA while writing the master thesis.

3 Introduction

This thesis marks the end of my master studies at the Norwegian University of Life Science, UMB. It describes my work on developing and implementing the start-up sequence for the PTT project at Rosum Corp in Mountain View, CA.

3.1 GPS and TV positioning

The GPS system, initially devised in the early 1970s, is widely used for position location, navigation, survey, and time transfer. The GPS system is based on a constellation of 24 on-orbit satellites in a sun-synchronous 12 hour orbits. Each satellite carries a set of precision atomic clocks and transmits pseudo-noise signals, which can be precisely tracked to determine pseudo-range. By tracking 4 or more satellites, one can determine precise position in three dimensions in real time, world-wide.

GPS has revolutionized the technology of navigation and position location. However in some situations, GPS is less effective. Since the GPS signals are transmitted at relatively low power levels (~ 500W EIRP) from satellites that are in orbits around 20200 km above the surface of the Earth, the received signal strength is relatively weak; in the order of –130 dBm. Thus, the signal is marginally useful or not useful at all inside buildings.

While GPS is easily obstructed indoors and in urban areas, it has excellent coverage elsewhere. TV, on the other hand, gives great coverage in urban areas, but may lack coverage elsewhere. Therefore, the Rosum HPM contains both a GPS receiver and a TV-signals receiver. With this, Rosum are to provide "location, inside and out™."

The assignment is based on the principles behind GPS and TV positioning, using both these technologies to make a more complete positioning system, with high accuracy in rural, urban areas together with inside buildings. The focus in this assignment is the interface controller between the GPS and the Baseband generator, which generates the TV signals, and how to make the timing correct to reduce the error in accuracy. The general approach is to reduce errors, to make the accuracy of the system higher.

3.2 Motivation

In the recent years, we have become more and more aware of the limitations of the GPS technology and the need for developing a system that work indoor. We

have the seen a need for more accurate violence alarms for protection and system which can follow the rescue workers when entering tall buildings in case of fire or nature catastrophes. I was introduced to Rosum's indoor-positioning technology through an internship and was soon interested in being a part of the further development of the exciting prototype for the PTT system. This project demands an interface controller which handles a special start-up sequence to make the positioning as accurate as possible.

3.3 Definition of the problem

The overall system accuracy is affected by how well the PTT locations are known, as well as the knowledge of clock differences among the PTTs. The transmitter part of a PTT will be tied to the same clock as the co-located GPS receiver. In this way, change in time offsets in the transmitted signal can be observed though GPS measurements. However, it is of vital importance to identify the initial time reference upon start-up of the system. The system consists of several modules that need to communicate with each other through an interface controller. In order to sync up the system before transmission it has to be turned on in a specific sequence. It needs to detect a 1-Pulse-Per-Second (1PPS) signal that is used to make a rough time location to the interface controller. The approach to the problem is to find out how the sequence must be set up, and which devices that has to be turned on in witch sequence. Another part of it is to design and implement this program on the interface controller. It is necessary to find where the errors might occur by practical tests, to see where the system can be improved.

3.4 Objectives

This assignment demands great understanding of GPS and its functionality, it also demands great understanding of TV-signals and how they are used for positioning. Since the program is to be implemented on the Zeus board (see section 7.3.2), which is a PC104 embedded computer, the use and functionality of an embedded computer with a lot of accessories has to be used. Another purpose is to learn how to do a project, meet with customers and the following up process. In this project we also have to select the different hardware components. Even if this assignment was concentrating on the hardware used in for the interface controller, the embedded computer, it was also important to be involved with the other hardware choices for the complete system. To be able to have a complete overview, it was important to attend all meetings regarding the system, this was important since the interface controller had to control the communication between all the parts in the system.

3.5 Restrictions

This project includes:

i) Find the best start up-sequence
ii) The development of the process
iii) The implementation of the program
iv) Make the softeware and the start-up sequence play along with other parts of the software developed for this project.

The project is limited to:

i) The development of the start-up sequence which will be run on the Zeus board.
ii) This project is written for a company, and we have chosen to keep the document as an open document. Because of this, there will not be represented detailed description about the hardware modules and software solution not directly related to my part of the assignment.
iii) The actual position fixes are not published due to restrictions of the paper. To show the results a visualized description and interpretation of results are shown.

3.6 About Rosum Corporation

Rosum was founded in 2001 by Dr. James Spilker, the co-architect of GPS, and Dr. Matthew Rabinowitz, an expert in high-precision navigation systems. The two founders wanted to address some fundamental limitations of GPS, and they found TV signals to be excellent sources for positioning. Rosum is located in Mountain View, California in the middle of Silicon Valley and are funded with external capital from different investors like Charles River Ventures, allegis Capital, In Q tel, Motorola ventures, Steamboat ventures and KTB ventures. Rosum employs approximately 20 people, most of which work in research and development. Rosum's technology addresses the limitations of GPS technology indoor and in urban areas, where the GPS signals are often too weak for detection. The markets they are addressing are E911, first-responders and asset tracking. Rosum doesn't want to make devices for the end-user, but rather lease out the technology and make a system that fits the customers need. Rosum's current product is a receiver called the HPM – Hybrid Positioning Module that contains both a GPS and a TV positioning receiver.

4 Terminology

Here is a presentation of words and abbreviation is used in the paper. The chapter includes both definition of technical terms and a list of symbols.

4.1 Definition of technical terms

Terminology	**Description**
GPS	Global Positioning System
HPM	Hybrid Positioning module
PTT	Pseudo Television Transmitter
ATSC	Advanced Television Systems Committee is a group that developed the ATSC digital television standard for the United States. ATSC standard is replacing the analog NTSC television system.
Duty Factor	Pulse width/ pulse period
DGPS	Differential Global Positioning System is an enhancement to Global Positioning System that uses two or more reference stations to broadcast the difference between the positions indicated by the satellite systems and the known fixed positions.
Baseband Generator	The baseband generator is the part that actually generators the signals who are transmitted. This hardware was designed for the PTT-system and has special designed antennas.
Interface controller	A controller which handles and controls the communication with the hardware parts included in the system. In the PTT system an embedded PC-104 board that has different features from supplier and applications made by Rosum is used.
Embedded computer	An embedded computer is a computer designed to perform one or a few dedicated functions, often with real time computing. They are dedicated to handle a specific task, which may require very powerful processors. It can be optimized to reduce size and cost with increased reliability and performance.

Multipath	In wireless telecommunication, multipath is the propagation phenomena that results in radio signals reaching the receiving antenna by two or more paths. Causes of multipath include atmospheric ducting, ionosphere reflection and refraction, and reflection from terrestrial objects such as mountains and buildings. The effects of multipath include constructive and destructive interference, and phase shifting of the signal.
PN511	The PN511 is a highly integrated transmission module for contact less communication at 13.56 MHz. This transmission module utilizes an outstanding modulation and demodulation concept completely integrated for a variety of passive contact less communication methods and protocols at 13.56 MHz
M sequence	A pseudorandom binary sequence of *m* bits that (a) is the output of a linear shift register and (b) has the property that, if the shift register is set to any nonzero state and then cycled, a pseudorandom binary sequence of a maximum of $n = 2^m - 1$ bits will be generated, where m is the number of stages, *i.e.*, the number of bit positions in the register, before the shift register returns to its original state and the *n*-bit output sequence repeats. *Note:* The register may be used to control the sequence of frequencies for a frequency-hopping spread spectrum transmission system
GCR	Ghost Cancelling Reference, is a special sub-signal on a television channel that receivers can use to attenuate the ghosting effect of a television signal split into multiple paths between transmitter and receiver.
EIRP	Effective isotropic radiated power is the amount of power that would have to be emitted by an isotropic antenna (that evenly distributes power in all directions and is a theoretical construct) to produce the peak power density observed in the direction of maximum antenna gain. EIRP(dBm) = (Power of Transmitter (dBm)) – (Losses in transmission line (dB)) + (Antenna Gain(dBi))
DOP	Dilution of precision
GDOP	Geometric Dilution of Precision

I²C BUS	A bus designed by Philips to allow easy communication between components. The original communication speed was defined with a maximum of 100 Kbit per second and many applications don't require faster transmissions. For those that do there is a 400 Kbit fast mode and - since 1998 - a high speed 3.4 Mbit option available. Most significant features include: • Only two bus lines are required • No strict baud rate requirements, the master generates a bus clock • Simple master/slave relationships between all components. Each device connected to the bus is software-addressable by a unique address • I2C is a true multi-master bus providing arbitration and collision detection
GPIO	General Purpose Input/output is a generic pin on a chip whose behavior can be controlled/programmed through software.
USRP	The Universal Software Radio Peripheral™ (USRP™) products are a family of computer-hosted hardware offered by Ettus Research LLC and its parent company, National Instruments, for making software radios. The USRP™ product is a hardware device facilitating the building of a software radio. The USRP™ hardware connects to a host computer through a high-speed USB or Gigabit Ethernet. In PTT system, the USB connection is used.
RAP	A program that gets the location data and sends it to Rosum location server where the position is calculated and returned to the user device.

Table 4-1: Definition of technical terms

List of symbols

Symbol	Description	Units
SVx_i	The satellite i's coordinate in the x direction, from the earth center, se figure 5.1	m
SVy_i	The satellite i's coordinate in the y direction, from the earth center, se figure 5.1	m
SVz_i	The satellite i's coordinate in the z direction, from the earth center, se figure 5.1	m
P_i^S	The satellite i's pseudo range. Pseudo range is (time difference * speed of light). Time difference is the time measured at the satellite and the time measured at the receiver)	m
R_x	Receivers estimated coordinate in x direction	m
R_y	Receivers estimated coordinate in y direction	m
R_z	Receivers estimated coordinate in z direction	m
R_i	The range from receiver position estimate to Satellite.	m
$\rho^S(t,t^S)$	The range from receiver (at receiver time) to the satellite (at transmission time)	m
Dx_i	Directional derivate for coordinates *xyz..i*	s
T	The known reading of the receiver clock when signal is received	s
T^S	The reading of the satellite clock when the signal was transmitted	s
t	True time received	s
C	Speed of light in vacuum = 299792459m/s.	m/s
τ	Clock bias for both receiver and satellite clock	s
$P^S(t)$	The pseudo range as a function of the true time the signal was received	m

Table 4-2: List of Symbols

5 Theory

5.1 GPS positioning

The Global Positioning System (GPS) is a space-based global navigation satellite system that provides reliable location and time information in all at all times on or near the Earth when and where there is an unobstructed line of sight to four or more GPS satellites. It is maintained by the United States government and is freely accessible by anyone with a GPS receiver. The GPS system is based on a constellation of 24 on-orbit satellites in a sun-synchronous 12 hour orbits. [2]

To use these satellites for GPS positioning, the GPS receiver detects the signal transmitted from the satellite in the direction of the Earth. This signal is encoded with Navigation Information which includes orbit parameters. The orbit parameters make the GPS receiver able to compute satellites coordinates (*SVx, SVy, SVz).* These are Cartesian coordinates in a geocentric system, known as WGS-84, which has its origin at the Earth center of mass, Z axis pointing towards the North Pole, the X pointing towards the Prime Meridian, and the Y at right angles to the X and the Z to form a right-handed orthogonal coordinate system. Figure 5-1 shows how the axis is. The algorithm which transforms the orbit parameters into WGS-84 satellite coordinates at any specified time is called the "Ephemeris Algorithm". [2]

5.1.1 Signal Description

The signals from the GPS are driven by atomic clocks. The fundamental frequency is 10.23 MHz. Two carrier signals are created from this signal by multiplying the frequency by 154 for the L1 channel and 120 for the L2 channel. The reason for the second signal is for the self-calibration of the delay of the signal in the Earth's ionosphere. The information is encoded in the form of binary bits on the carrier signal by phase modulation.

There are three types of codes on the carrier signals:

- The C/A code
- The P code
- The Navigation message

The C/A ("course acquisition") code can be found on the L1 Channel. This is a code sequence which repeats every second. It is a pseudo random code which is generated by a known algorithm. Each satellite has a different C/A code, so that they can be uniquely identified.

The P (“precise) code is identical on both the L1 and L2 channel. Whereas C/A is a courser code appropriate for initially locking onto the signal, the P code is better for more precise positioning.

The navigation Message can be found on the L1 channel, being transmitted at a very slow rate of 50pbs. The navigation Message includes information on the Broadcast Ephemeris (satellite orbital parameters), satellite clock corrections, almanac data (a crude ephemeris for all satellites), ionosphere information, and satellite health status.

How the signal is generated are not applicable for this paper, but can be studied in some of the references used.

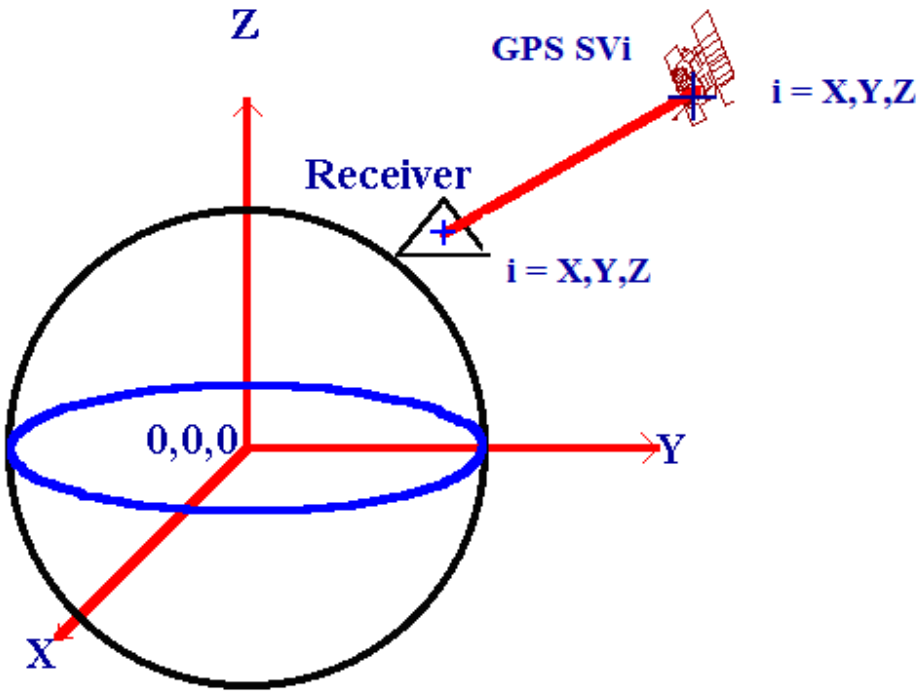

Figure 5-1, shows the earth center with the axis and the satellite and receiver. Where SVx is the satellite’s coordinate in the x direction, SVy is the satellite’s coordinate in the y direction and SVz is the satellite’s coordinate in the z direction. ([1] Dana, Peter H.1994)

5.1.2 Navigation equation – pseudo range observation equation

[2] The receivers record data at regular, specified intervals. It is the reading of the receiver clock time T which is used to say exactly when the measurements are

sampled. Therefore, the value of T at a measurement epoch is known exactly, and is written to the data file log along with the observations. (What is not known is the true time of measurement) The actual observation to satellite SV can be described: Pseudorange = (time difference * speed of light) = P^S

(A.1) $$P^S = (T - T^S)c$$

Where T is the known reading of the receiver clock when signal is received, T^S is the reading of the satellite clock when the signal was transmitted, and c is the speed of light in vacuum = 299792459m/s.

Pseudorange = (time difference * speed of light) = P^S

The modeled observation can be developed by setting the clock time T equal to the true receive time t plus a clock bias τ, for both receiver and satellite clocks: [3]

(A.2) $$\begin{aligned} T &= t + \tau \\ T^S &= t^S + \tau^S \end{aligned}$$

Substitution gives the pseudo range as a function of the true time the signal was received:

(A.3) $$\begin{aligned} P^S(t) &= ((t+\tau) - (t^S + \tau^S))c \\ &= (t - t^S)c + c\tau - c\tau^S \\ &= \rho^S(t, t^S) + c\tau - c\tau^S \end{aligned}$$

Where $\rho^S(t, t^S)$ is the range from receiver (at receiver time) to the satellite (at transmission time.)

From Pytagoras theorem, it gives:

(A.4) $$\rho^S(t, t^S) = \sqrt{\left(SVx(t^S) - Rx(t)\right)^2 + \left(SVy(t^S) - Ry(t)\right)^2 + \left(SVz(t^S) - Rz(t)\right)^2}$$

The navigation message allows us to compute the satellite position (SVx, SVy, SVz) and the satellite clock bias τ^S. We are now left with 4 unknowns, the receiver position R_i and the receiver clock bias τ.

To do GPS pseudo range Navigation the satellites (SV) coordinate in ECEF XYZ from Ephemeris Parameters and SV Time is used. Shown below as following:

Satellites Pseudo range in meters (from C/A code epochs in milliseconds

(A.5) [2]

$$\begin{matrix} SVx_0 & SVy_0 & SVz_0 \\ SVx_1 & SVy_1 & SVz_1 \\ SVx_2 & SVy_2 & SVy_2 \\ SVx_3 & SVy_3 & SVz_3 \end{matrix}$$

5.1.3 Positioning using pseudo range – Least squares estimations

Linearized model

[1] [3]. The positioning problem can be solved by first linearizing the pseudo range observation equation, and then using the familiar methods of least squares analysis. For completeness, we summarize the linearization procedure and the development of the least squares method specifically for the GPS point positioning problem. First we assume we can write the actual observation to be the sum of a modeled observation, plus an error term

(B.1) $$\begin{aligned} P_{observed} &= P_{\mathrm{mod}el} + nois \\ &= P(x, y, z, t) + ClockBias \end{aligned}$$

Next, Taylor's theorem is applied, where the equation is expanded about the model computed using provisional parameters values (Rx, Ry, Rz, Rt).[3]

(B.2) For each of 4 SV's i = 0...3

$$P(SVx, SVy, SVz, SVt) = P(Rx, Ry, Rz, Rt) + (SVx_i - Rx)\frac{\partial P}{\partial SVx} + (SVy_i - Ry)\frac{\partial P}{\partial SVy} +$$

$$(SVz_i - Rx)\frac{\partial P}{\partial SVz} + (SVt_i - Rt)\frac{\partial P}{\partial SVt}$$

$$= P_{computed} + \frac{\partial P}{\partial SVx}\Delta x + \frac{\partial P}{\partial SVy}\Delta y + \frac{\partial P}{\partial SVz}\Delta z + \frac{\partial P}{\partial SVt}\Delta t$$

We use the following substitution, where Range from Receiver Position Estimate to SVs (R) and Array of Observeb, subtracted with Predicted or computed Ranges [3]

(B.3) $$\Delta P = P_{observed} - P_{computed}$$

$$R_i = \sqrt{(SVx_i} - Rx)^2 + (SVy_i - Ry)^2 + (SVz_1 - R_z)^2$$

The result can be written:

(B.4) $$\Delta P = \frac{\partial P}{\partial SVx}\Delta x + \frac{\partial P}{\partial SVy}\Delta y + \frac{\partial P}{\partial SVz}\Delta z + \frac{\partial P}{\partial SVt}\Delta t + v$$

To simplify, the Directional Derivates for XYZ and Time are computed.

(B.5) $$Dx_i = \frac{SVx_i - R_x}{R_i}, \qquad Dy_i = \frac{SVy_i - R_y}{R_i}, \qquad Dy_i = \frac{SVz_i - R_z}{R_i} \quad Dt_i = -1$$

Solve the equation for correction to receiver Position Estimate, with the directional derivate, using least squares, and shown below. The coefficients Δx, Δy, Δz, Δt directions cosines.

(B.5) $$\begin{bmatrix} \Delta P_0 \\ \Delta P_1 \\ \Delta P_2 \\ \Delta P_3 \end{bmatrix} = \begin{bmatrix} Dx_0 & Dy_0 & Dz_0 & Dt_0 \\ Dx_1 & Dy_1 & Dz_1 & Dt_1 \\ Dx_2 & Dy_2 & Dz_2 & Dt_2 \\ Dx_3 & Dy_3 & Dz_3 & Dt_3 \end{bmatrix} * \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta t \end{bmatrix}$$

Equation B.5 can be expressed as following:

$$\partial L = A\partial R$$

Using least squares to solve the equation:

$$dR = (A^T \cdot A)^{-1} \cdot A^T \cdot L$$

Calculate satellites positions and apply satellite clock corresponding to the pseudo range measurements. Form the initial position estimate iterate until convergence. Calculate approximate pseudo range based on position estimate and SV positions. Form the geometry matrix A. Subtract measured pseudo ranges from estimated pseudo ranges. Update the user state by solving equation B5. Apply Corrections to Receiver XYZ and Compute receiver Clock Bias Estimate.[2] B.6)

(B.6) $$Rx = Rx + dR_0,\ Ry = Ry + dR_1,\ Rz = Rz + dR_2,\ Time = dR_3$$

5.1.4 Position fix

A position fix is the GPS receiver's computed position coordinates. There is several possibilities of calculate the position fix. As an example it can be based on the last position the receiver had when receiving signals from the satellite. Another way is to using at least one GPS signal and determining when a pseudo range can be measured there from. See equations in 5.1.3

5.2 Error contributions and Dilution-of-Precision (DOP)

5.2.1 GPS Errors

GPS errors can be divided into two categories: Gaussian noise and bias. The different problems will affect the fixes differently. Noise errors are the combined effect of PRN code noise, that can be give an error around 1 meter and noise within the receiver (around 1 meter). The noise can be caused just by the environment we are surrounded by in normal life.
Bias errors can be a result of Selected Availability (SA), which was the intentional degradation of the Standard Positioning Service (SPS - a.k.a. the civilian signal) signals with a time varying bias. SA was controlled by the Department of Defense (DoD) to limit accuracy for non-US military and government users. SA was turned off permanently on May 1 1999. Other bias errors can be Space Vehicle (SV), clock errors uncorrected by Control Segment, ephemeris data errors, troposphere delays, un-modeled ionosphere delays and multipath.

Multipath is caused by reflected signals from surfaces near a receiver that can either interfere with or be mistaken for the signal that follows the straight line path from the satellite. Multipath is difficult to detect and can be hard to avoid.

All the errors mentioned above is typically for regular GPS systems, and some of these errors would be reduced with use of differential GPS (DGPS). In comparison, regular GPS accuracy is approximately 10 m (2D), 16 m (3D), while Differential GPS accuracy - 0.5m to 5m (2D), 0.8m to 8m (3D).

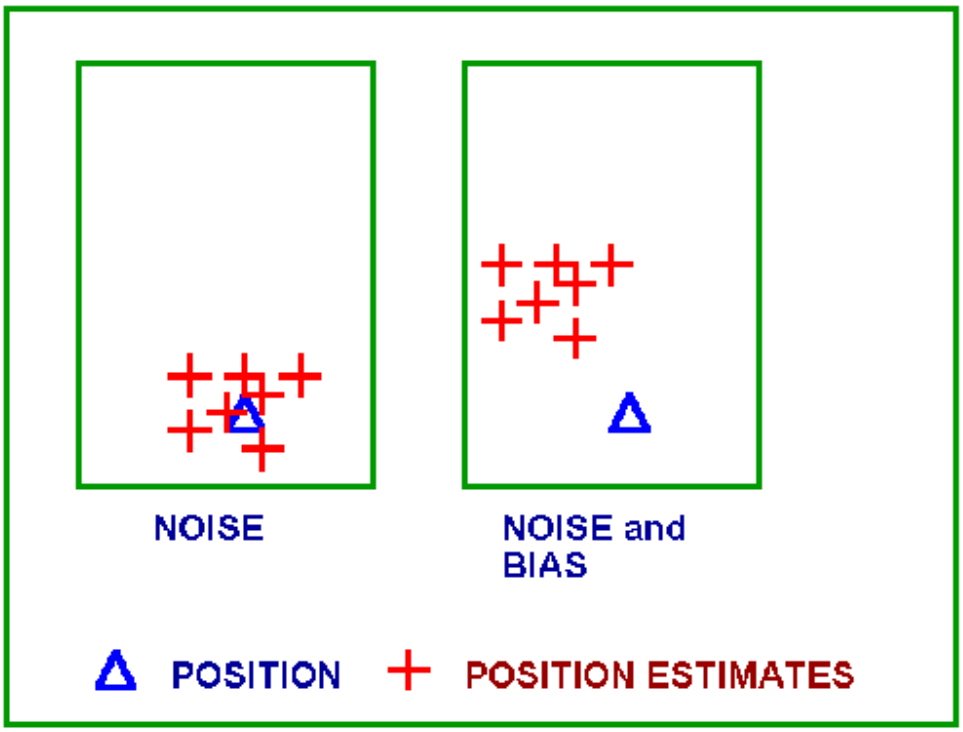

Figure 5-2, shows the position fixes affected by one or more types of errors. [1] Dana, Peter H.(1994)

5.2.2 Dilution of precision

Dilution of precision (DOP) tells us the geometric strength of the satellites configuration on the fix accuracy. The GPS need 4 visible satellites to position; an important factor is also how they are moving in relation to each other. If they are close to each other, there will be fields that are not covered by a satellite, and the geometry is said to be weak and the DOP value is high: when far apart, the geometry is strong and the DOP value is low.
GPS ranging errors are magnified by the range vector difference, between the receiver and the satellites. The volume of the shape described by the unit-vectors from the receiver to the satellites used in a position fix is inversely proportional to GDOP. Figure 5-3 and 5-4 shows a drawing of how the satellites are placed to each other in good and bad DOP.
The factor that affect the DOP besides the satellite orbits are, the presence of obstructions which make it impossible to use satellites in certain sectors of the local sky. Especially in urban measurements, this may be limiting.

GDOP is based on 4 components; PDOP (Position Dilution of Precision(3-D)), sometimes the Spherical DOP, HDOP(Horizontal Dilution of Precision(Latitude, longitude)), VDOP (Vertical Dilution of Precision(Height)), TDOP (Time dilution of Precision (Time). Even each of these GDOP terms can be individually computed, they are formed from covariance and are not independent of each other. A high

TDOP(time dilution of precision), will for an example cause receiver clock which will eventually result in increased position errors. The DOP value will tell you how reliable your fixes are.[1]

DOP Value	**Rating**	**Description**
1	Ideal	This is the highest possible confidence level to be used for applications demanding the highest possible precision at all times
2-3	Excellent	At this confidence level, positional measurements are considered accurate enough to meet all but the most sensitive applications
4-6	Good	Represents a level that marks the minimum appropriate for making business decisions. Positional measurements could be used to make reliable in-route navigations suggestions to the user.
7-8	Moderate	Positional measurements could be used for calculations, but the fix quality could still be improved. A more open view of the sky is recommended
9-20	Fair	Represents a low confidence level. Positional measurements should be discarded or used only to indicate a very rough estimate of the current location.
21-50	Poor	At this level, measurements are inaccurate by as half a football field and should be discarded.

Table 5-1; shows a description of the different DOP values. [1] Dana, Peter H.(1994)

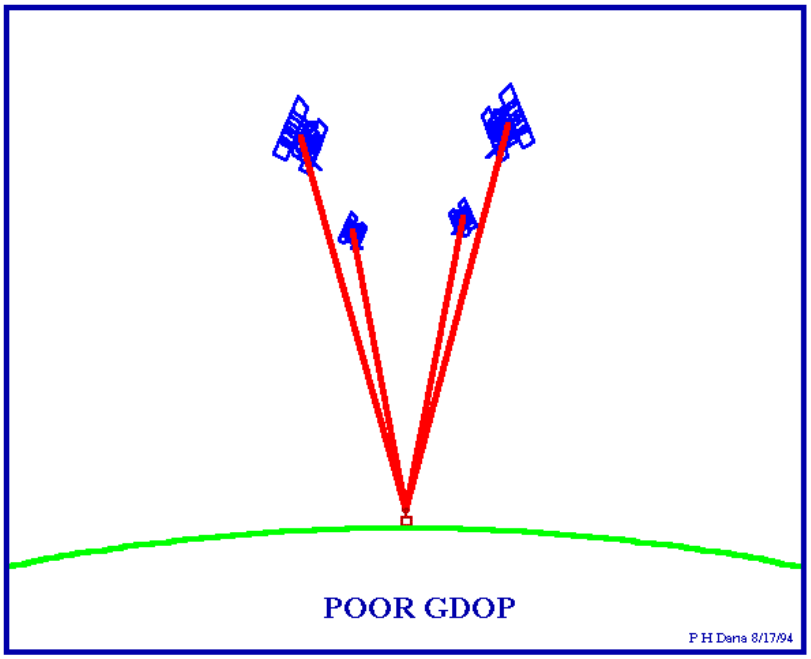

Figure 5-3, The satellites in a geometry that gives poor DOP.
[1]Dana, Peter H.(1994)

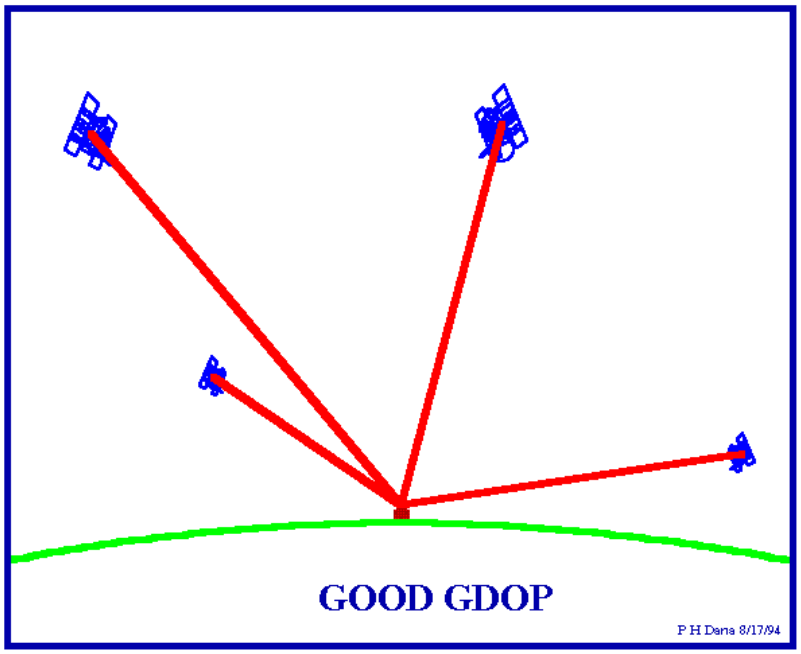

Figure 5-4, the satellites in a geometry that gives good DOP.
[1] Dana, Peter H.(1994)

5.2.3 DGPS Errors

Typical DGPS requires one reference receiver that compute measurement corrections, and one or more “roving” receivers that receive the correction data. If the distance between rover and reference is relatively small (~200 km), many of the before mentioned errors will be common mode and drop out from the navigation equations. Differential GPS became popular as a result of “Selective Availability” (SA), but will also largely remove errors from ephemeris, ionosphere, troposphere and SV clock. However, multipath errors are unlikely to be common mode, and they could hurt accuracy.

5.3 TV-signal positioning

Rosum uses TV-signals to positioning in places and areas where GPS doesn’t work. Since TV signals are transmitted at tremendously higher power than GPS, they can more easily penetrate building walls.
The signals used for this purpose in the US are ATSC (American TV Standards Committee) and NTSC (National Television Systems Committee). The former is the US digital TV standard, and the latter is the US analog TV standard. Both standards are transmitted in channels that are 6 MHz wide, in 3 bands that span from 54 MHz to 800 MHz
An ATSC half-frame consists of 313 segments, where each segment contains 832 symbols that are 8-VSB modulated. Every segment starts with the same 4-symbol sequence, and the first segment in a half-frame contains a 511 symbol long pseudo-noise (PN) M-sequence. Both the 4-symbol segment sync and the PN511 can be used to measure pseudo ranges between a transmitter and a user.
An NTSC field consists of 525 lines, with the option of inserting an alternating Ghost Cancelling Reference (GCR) every 262-263 lines. The first 10 use of each line contains line sync, and the GCR is in essence a chirp signal (linear frequency sweep). Both signal features can be used for pseudo ranging.
TV transmitters are generally close to population centers and the transmitters broadcast signals at levels of 30 kW to hundreds of kilowatts. With transmitter antenna gain, many stations transmit at more than 1 MW of Equivalent Isotropic Radiated Power (EIRP). TV systems are fundamentally designed so people can watch TV from the comforts of their living room using a set of indoor “rabbit-ear” antennas. (Rosum Corp (2006))
Rosum only needs to track the above synchronization codes and no actual decoding of the data signals are needed. Thus, the ability to track signals indoor at substantial ranges from TV towers can be greatly enhanced through the use of properly designed receivers. Furthermore, the signal processing needed to perform tracking has been implemented in a single Integrated Circuit (IC).

Rosum have two types of systems: the wide area system and the local-area Pseudo Television Transmitter (PTT) system, which is the object of this thesis.

In the wide-area system, the location of the transmitters is found from Federal Communications Commission (FCC) databases. However, a monitor system is required in order to synchronize TV transmitter clocks since they are not generally tied to a common and stable clock standard.
The PTT system is deployable, and the transmitter locations will be surveyed using GPS. GPS will also provide a common and stable clock standard to all transmitters.

5.4 PTT system

The PTT system is another product Rosum is working on, also called TV positioning plus, and is the project this paper is based on. The main purpose is to get better accuracy indoor when emergencies. An example of use of the system will be to track a firefighter that goes into a burning building. If the firefighter doesn't get out again, you can find his which room he is in, even if there is smoke every way. This system is based on an idea from Dr. Guttorm Opshaug and are built on the same principles as the wide area system, but in this case own TV signals are generated own, and the transmitters are much closer to the building.

While the projects move forward the performance of the system is not to be changed. All the specifications has to be fulfilled at each state, there is just how the system is build that is changing.

In the original prototype design all the PTTs have fixed positions, and are a wired system. All the PTT has a known longitude, latitude and altitude and they are power by external power. The signals sent out is generated from a signal generator where the signal was amplified and sent over calibrated coax cables to the PTTs antennas. It has to be designed to meet the accuracy on < 6m, and that it should be able to give position and floor in a 25 store building in an urban environment. Figure 5-5 shows the function of the original prototype design. The system has 4 PTTs (Pseudo Television Transmitters), a location server, PTT signal generator and a user device. The PTTs are placed one each side of the building and as the figure 5-5 shows. They are all sending out TV-like signals that are detected by the receiver, Location device, and send the information it gets to the server who then does the calculations. All the fixes are saved on the server, so you can get all the information later. The information is also sent back to user device and represented on the screen as number.

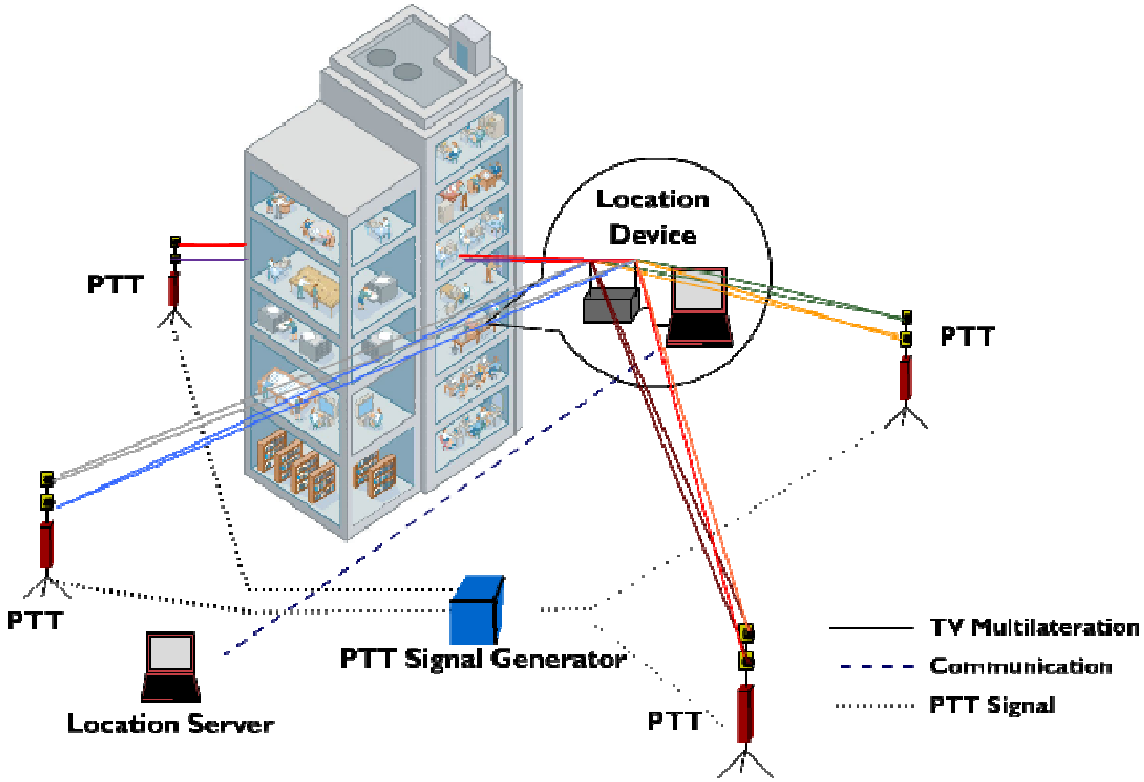

Figure 5-5, the PTT system at the start of the assignment, (Draw courtesy, Todd Young, Rosum (2007))

In the figure 5-6 you can see a presentation of all the communication between the Rosum user device and the Rosum location server. This is set up in the same way as when using the wide area system. The PTT project is not finished even if the interface controller is, so there will be some changes at this point.
The process starts by running the location application, RAP, that is a program that requests aiding data from Rosum Location Server (RLS) once, the communication used is a GPRS modem and over the GPRS network, and that is how the request goes to the server. The RLS provides aiding still over the GPRS to the HPM (once). The User Device gets the TV measurements and provides them to the RLS. The RLS does the calculations and provides the position fix to NMEA port. You can either choose to get the data at once, or analyze the results later. This is most likely different relating to the purpose.

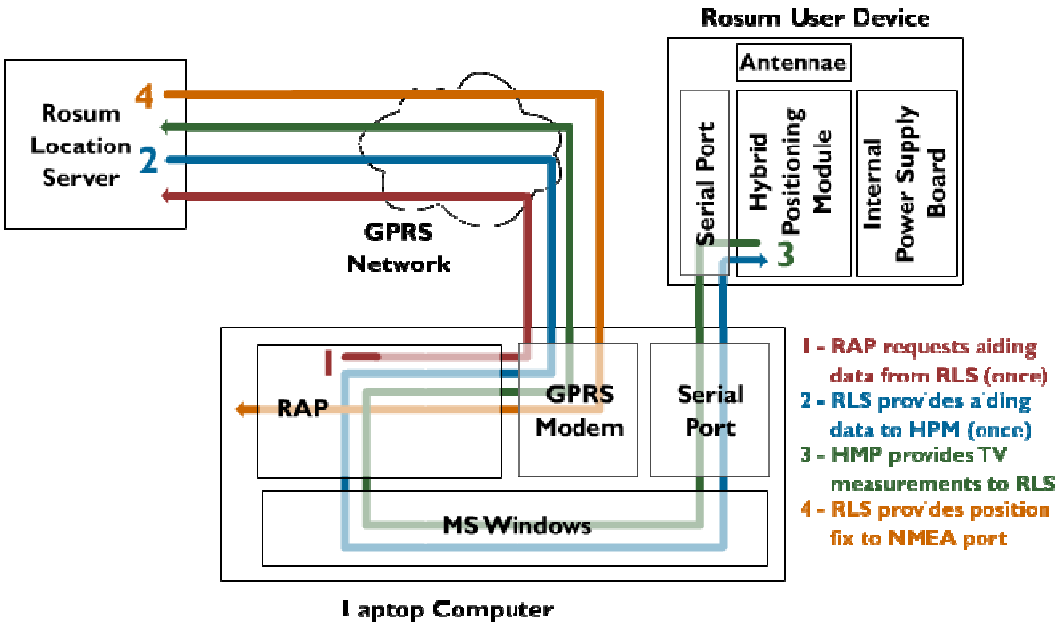

Figure 5-6, Shows the software communication and data transfer between the user device and the server. Drawing courtesy Yound, Todd.(2007). RosumCorp.

The improvements for the system are that the PTTs has no longer a fixed position, they have to be able to locate themselves with use of a DGPS. If the system weren't able to locate itself, the system would not be in useful for buildings where you didn't know the exact how the PTTs are positioned to each other. Another improvement is that the system is fully automatic.

5.4.1 PTT (Pseudo Television Transmitter)

To be able to position a user device accurately, knowing the position of the PTT is crucial and it has to be able to self-locate. This feature will be provided by GPS. Differential GPS (DGPS) will be used, since the absolute position is not as important as the relative one, and DGPS provides a much better accuracy levels than stand-alone GPS. Additionally, barometric pressure sensors will add independent vertical measurements, which is the direction of highest uncertainty for GPS.

The PTTs transmit a signal that has the same spectral shape as an ATSC signal. However, the PTTs transmit continuous repetitions of only the PN511 sequence to improve overall Signal-to-Noise-Ratio (SNR). In order to provide frequency diversity, we have chosen to broadcast our high-duty-factor location signal on two separate frequencies from each Beacon. It is advantageous to have

frequency diversity in the signals used for ranging in a positioning system, because attenuation and multipath effects are both frequency-dependent. These effects are greatly amplified in urban and indoor environments where signal attenuation and multipath effects are great. Having two or more signals at each transmitter increases the probability that a precise range can be computed between the user devices and the transmitter.

.

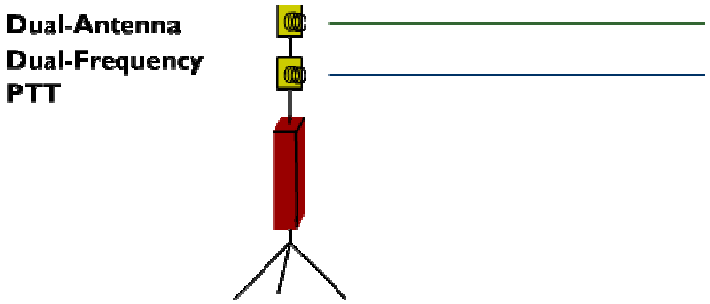

Figure 5-7, the PTT with single antenna dual frequency and dual antenna dual frequency. Drawing courtesy; Opshaug, Guttorm (2007) Rosum Corp.

5.4.2 User Device

The user device currently used is a development kit that is used for system testing purpose. As shown in figure 5-7; there is a laptop with a GPRS communication unit connected to the Rosum user device through the RS232 serial port at the HPM with a USB connection on the laptop. Since the code is set default to execute the communication on a higher baud rate than is supported through the RS232 we have to use an Edge port or make changes in the code. It is connected on the customer port where it reads in the data from. The Rosum User Device is a HPM without the GPS. It's chosen not to include the GPS in this project since, the PTTs use differential GPS with relative positioning and this other GPS would use absolute position. This user device will only be used in indoor environment, and it would most likely not see any satellites and result in no fixes. The HPM gets it power either from a wall plug, who is more than possible for testing purposes inside a building, or a battery to increase the flexibility which is needed in real life. The OS run on the computers are right now windows. Rap, who are the positioning code, exist in both Linux and windows format, so it's up to the user to choose what they prefer.

If a customer wishes to implement the receiver in their police radios or similar existing equipment, that is fully possible, but it is not a part of this project. The user device runs rap, who is the a program that gets the location data and sends it to Rosum location server where the position is calculated and returned to the user device, or where you want to have this position.

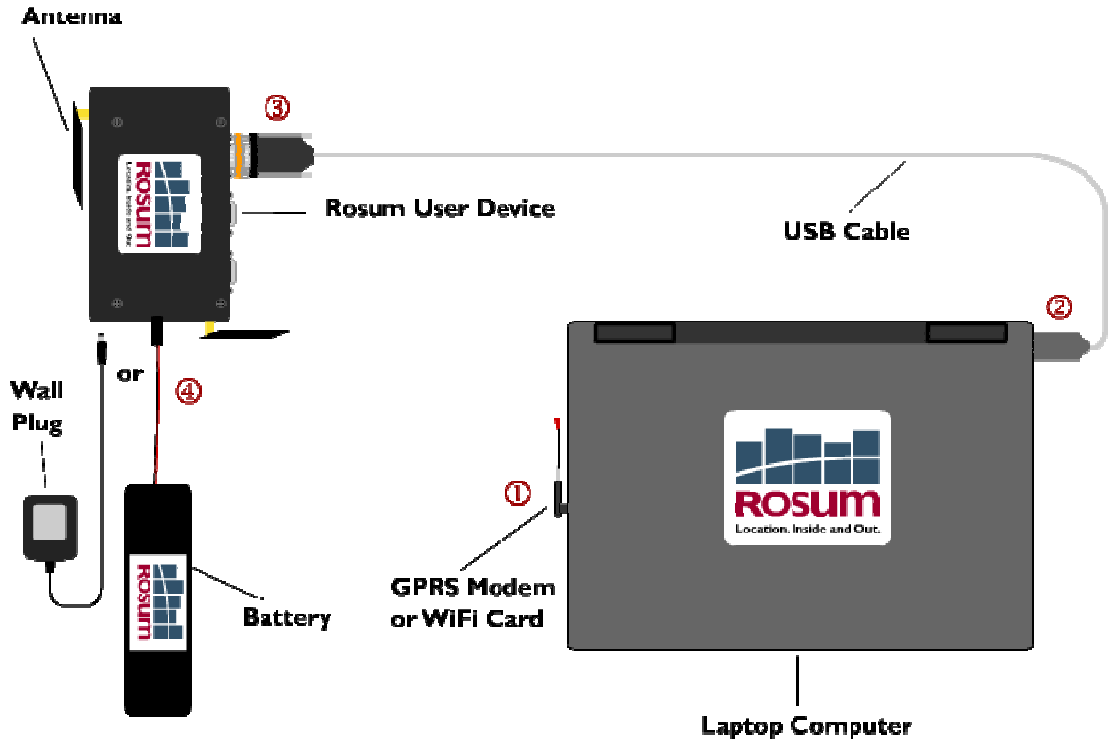

Figure 5-8, the used device with laptop. The same user device is used in both the wide area system and the PTT system. Drawing courtesy, Young, Todd(2007). Rosum Corp.

To make the best out of the user device it is enhanced to include a second antenna. The second antenna provides spatial diversity and doubles the number of independent ranging measurements processed. This will improve the error performance and decrease the probability of outage. The dual antenna device is shown in figure 5-9 below.

Figure 5-9, the Rosum User Device with dual antennas

5.4.3 Server

Rosum location server is where the calculations of your position are done. This gets all the information, calculates the position and sends it back to the user device that shows the position fix. The receiver, The Rosum User Device, communicates with the server on GSM, SMS, GPRS or WiFi, dependent on the hardware chosen. In the PTT system this will be chosen later in the assignment.

5.5 The difference between GPS and TV positioning and the PTT system

The main difference between GPS positioning and TV positioning are that the GPS signals are designed for positioning purposes and TV positioning is designed to broadcast TV in to you living room. That results in GPS being a great positioning tool outside, while TV work better inside. If you compare the two technologies outside in a rural area, GPS will be more accurate than the TV positioning, and are the reason why the User device for the wide area system is a hybrid with GPS also included. The PTT system is designed purely to position with high accuracy indoor and for use in security purposes and it will not make sense to use it for positioning outside or in not “specified areas”. In the three systems is also a difference on who are interested in knowing the position. GPS is used by, drivers and sailors which want to navigate, while when you are inside, there is very likely that you know where

you are, and more for other to know where you are located, for instance violence alarms. If the alarm goes off, the Police needs to know where you are located.

The technical difference between GPS positioning and TV positioning is that the TV signals are transmitted with much higher power than GPS; the TV signals can therefore more easily penetrate building walls. TV signals provide a 50dB power margin over GPS.
The GPS signals are transmitted with a power of 50 watt or less. The signal from the satellite has to pass through space and the atmosphere before reaching the receiver at the earth's surface, the distance is around 20200 km.

The PTT system is transmitted on less power than the TV signals for the wide area system, but it is also placed directly outside the building and maximum 50-60m away from the receiver.

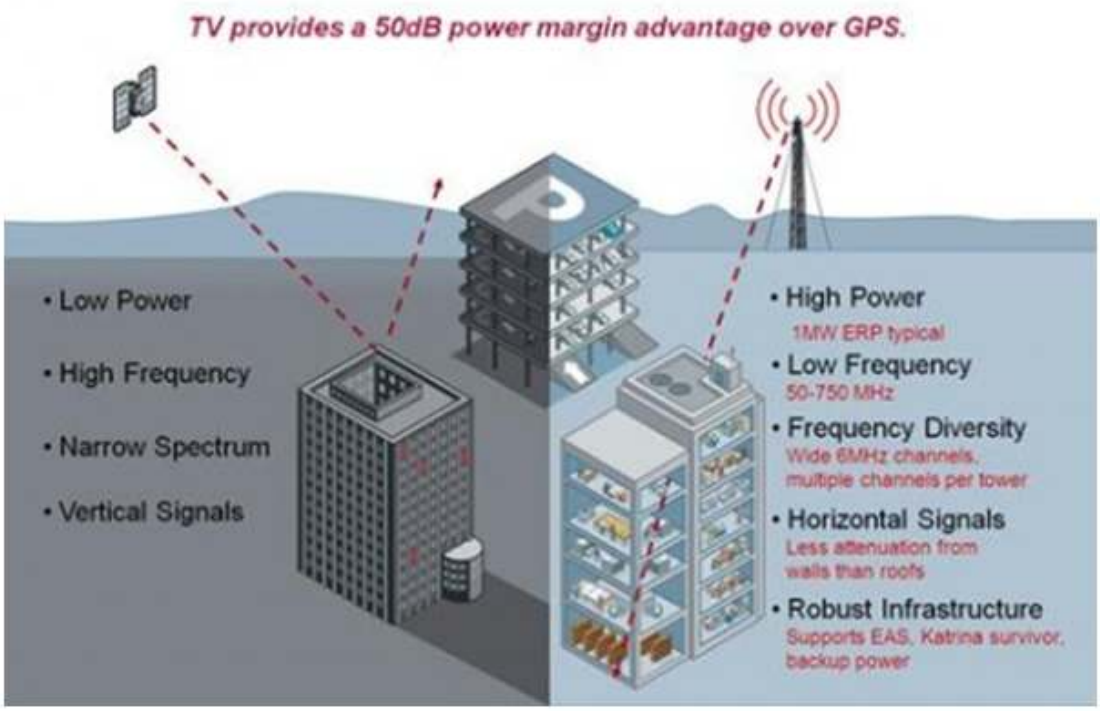

Figure 5-10, Differences between GPS and TV signals, [6] Davies, Chris (2009).

The TV-transmitters are often located close to the population centers, approximately 10-40 km away depending on the topology. The TV signals are transmitted with a power level of 5-10000 watts, and are designed and generated to reach into people's homes.

The GPS signal has a much higher frequency than the TV signals, and that differs from both the Wide area system and the PPT system. The TV signals are normally transmitted on 50-750MHz, while in the GPS system, all

satellites broadcast at the same two frequencies, 1.57542 GHz (L1 signal) and 1.2276 GHz (L2 signal). Lower frequencies can easier penetrate concrete structures, which provide possibilities of indoor coverage.[6]

When it comes to frequency diversity, as mentioned earlier the GPS only broadcast on two different frequencies from all the satellites, with a bandwidth on 24MHz. It has a narrow spectrum compared to TV positioning system where there are used several channels with 6MHz bandwidth, and there are multiple channels on each. The channels/frequencies transmitted from TV towers – might be the same or different. In the PPT system there are 4 different transmitters, which each transmits on two frequencies.[5]

Compared to the GPS signals, you don't need to decode the TV-signals. The position of the TV towers is fixed. Otherwise, TV positioning is based on the same principles as GPS positioning. The PTT system is based on the DGPS technology instead of the absolute GPS.

The local PPT system, which is designed to have a higher accuracy than the TV Wide Area TV system, it is based on the same principles, but in this case the TV signals are generated only for the positioning purpose and has a fixed position outside the building where the system are in use. The TV signals are transmitted from the baseband generator placed outside the building for positioning purposes and cover a rather small area.

A GPS receiver is a quit common devices which is normal in cars, sailing and today on almost all cellphones. To find out your position you only need contact with 4 satellites and your position is calculated in the device. So compared to the User device in TV positioning and the PTT system you don't need a second communication form in the receiver. The User device for TV positioning and PTT system sends the received information to the Location server, where the position is calculated and if wanted returned to the user device. The PTT system is more designed for other people to know where the user device is located, so it is not necessary to send the position back to the user device.

TV system is today more robust, than the GPS positioning. Since the TV signals are commercial, the broadcast company would have several phone call and complaints if it disappears, they also risk losing customer. It is very normal to have invested in redundancy and extra insurance of the TV tower. If something happens to TV GPS system, new satellites have to be shooting up in the air. The PTT system is a much smaller system and spear parts are available and easy to change. The system doesn't have to run 24/7, and therefore not so susceptible to damage.

6 Test results

When designing the start-up sequence there was need for live testing and also some of the points needed real live testing to tune in the exact numbers needed in the system. The test is done before the start-up sequence was final developed to find out the importance of the start-up sequence and other errors. It was necessary to see where how accurate the position fixes where, and where we could make the improvements after we had the facts from real practical testing. There are some limitations in the testing since the Rosum Corp office building only had two floors, and the system will be developed for a 25 story building.

The humidity and temperature is also different from the customer's site, where the test of the final system will be executed and demonstrated.

6.1 Full system test

To eliminate other error then the time there was done a practical test with the equipment where there were used fixed positions on the baseband transmitters. This was later taken into real life with testing with the equipment, but a complete final system test was not executed before after this assignment was done. The reason for using fixed positions was to isolate the single error sources.

In this case there was only need to send aiding request and get answer from Rosum Location Server once, since the positions where fixed. When testing the final system, the will be a higher need to pass aiding info. The setup is as shown in figure 5-5 and an overview drawing in 6-1.

6.1.1 Time and place

The different testing sites was chosen, because it was spread over the whole building and therefore gave the most representative data for a real life situation.

When doing the test, each user device will be left on for approximately 30 min, and this correspond to ca. 100 position fixes. The position fixes will not be dependent from the previous, since at this point there is no averaging function in use. When the averaging function is turned it will continue averaging until the motion sensors detects movement.

6.1.2 Goals for data

For analyzing the data, the 67% CEP will be used for presentation since this is the demand from the U.S government for 911 calls in the USA. The 67% CEP refers to data sorted in a descending way and find which fix encloses 67% of all readings. A reason for choosing 67% CEP is that it differs from 1-sigma, since 1-sigma would consider outliers, while 67% percentile will not.

Raw data is restricted to Rosum Corp and cannot be presented in the paper, but the results, interpretations and a visualization of the CEP will be shown for all 6 sites.

The Raw data is readings obtained without filtering. The currently filter is an averaging filter, and if an outlier comes up it will cause a worse result than only raw readings.

6.1.3 Setting up the system

The 4 Baseband transmitters, also called PTTs where placed on each side of the squared building. They were connect to a baseband generator, since the USRP was not yet finalized. To connect them it was used thick coax cables are rolled out around the building. It was important that the coax cables were put correctly without being in contact with edges of the building and being under pressure. This was to avoid unexpected error in the tests.

In this stage of the project, the tasks of the interface controller, was executed manually.

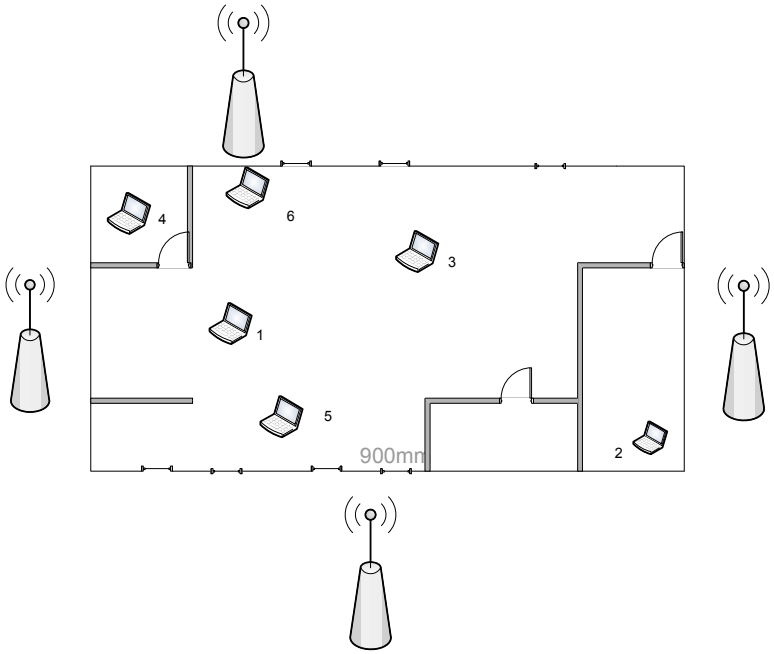

Figure 6-1, the set-up of the system, and used device test sites within the building

6.1.4 Test Results

Test site 1: The fixes are as expected and within 67% CEP, see figure 6-2 for visualization.

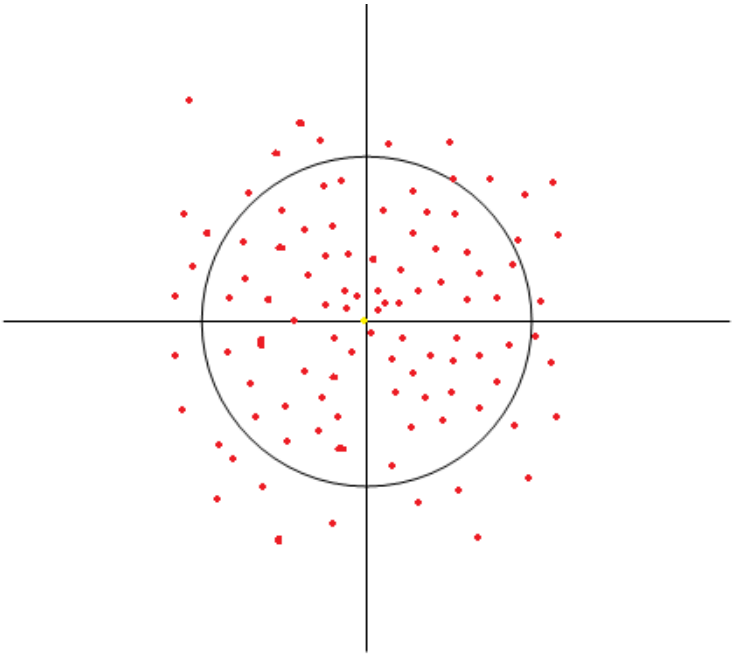

Figure 6-2, A visualized test result, for test site 1,3,5. Data is restricted.

Test site 2: The fixes are elongated in one direction; this is caused by the triangulation between the PTTs. To improve this, there must have been another PTT placed in that direction. See visualization in figure 6-3.

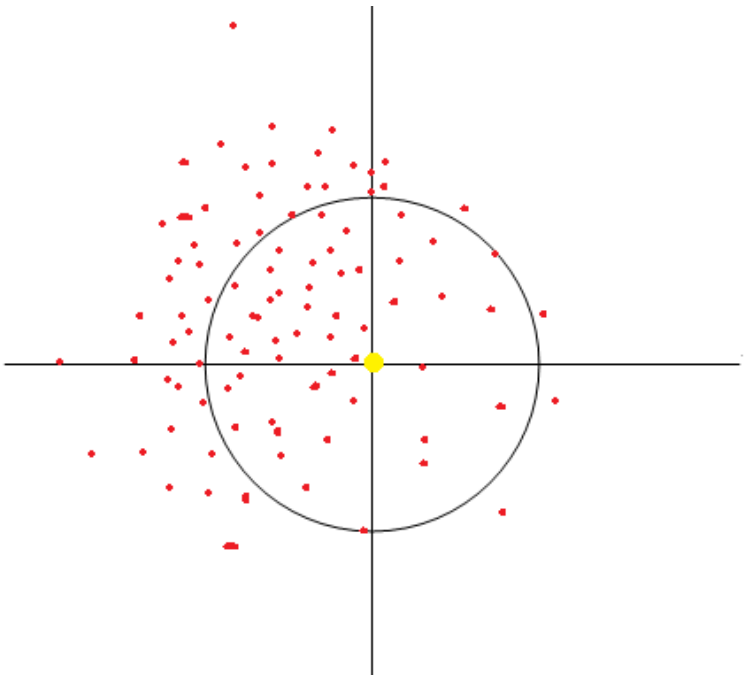

Figure 6-3, a visualized description of the test results for test site 2. In this case the 67% CEP is not met and where is should have been are only 49%.

Test site 3: The fixes are as expected and within 67% CEP

Test site 4: Because the correlation peaks are 80m wide, the MMA will only remove the multipaths that are way off. The closer ones are harder to detect as a multipath. A way to solve this is to increase the bandwidth transmitted, or to use a Box signal to narrow the correlation peak. It is noted that just increasing no of symbols transmitted from 511 to a larger no would not help with this problem. It is noted that because it took 7s to get the fix, one possibility is that clock drift would have contributed about 1 m difference between 1st and last position.

Test site 5: The fixes are as expected and within 67% CEP.

Test site 6: In this case the site is on the edge of the coverage area; the “isorange lines” from the transmitter intersect very shallowly at the site => less control over how the algorithm converges. Since starting iteration from the center of the building which is not equal to the true site, miss convergences occurs

easily. Thirdly, since true site is very near one of the PTTs, linearization of the equation is difficult.

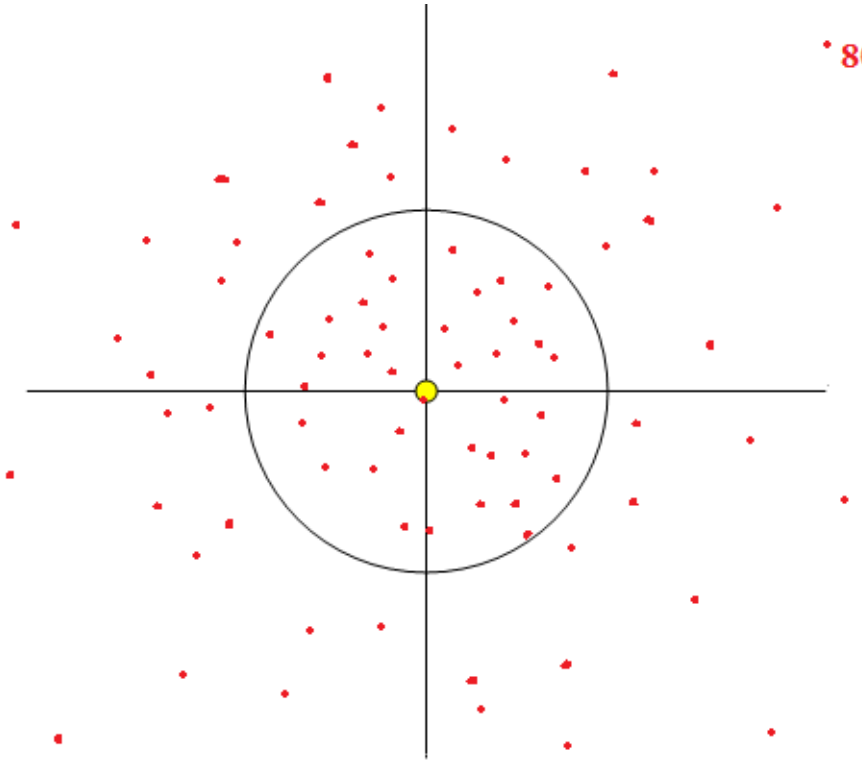

Figure 6-4, visualizes the result in test site 4, with up to 80 m correlation peaks. The circle is 37% CEP.

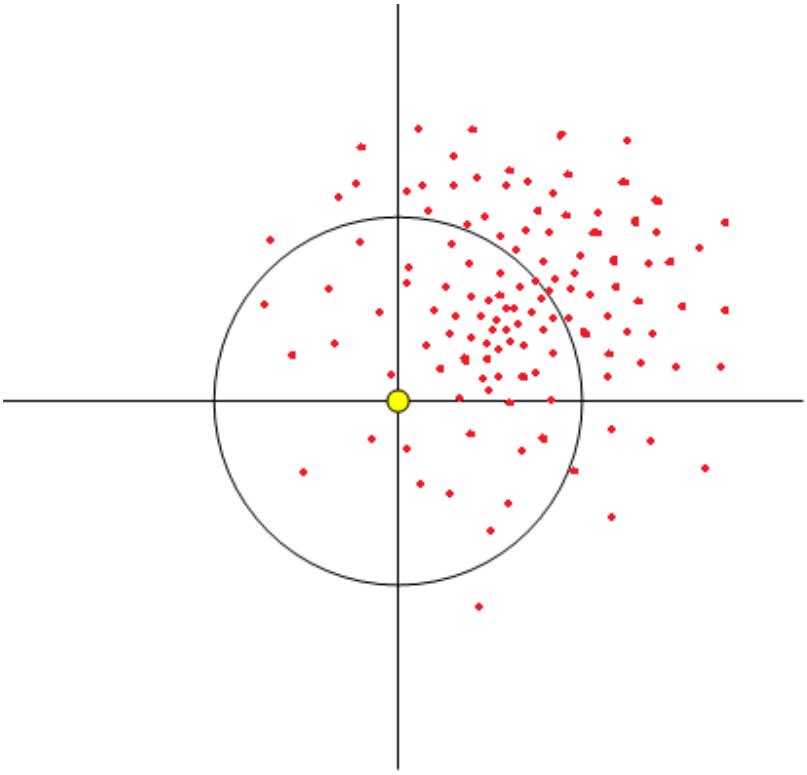

Figure 6-5, visualizes the result in test site 4, the circle is 55% CEP. It shows that the true site is not correct.

6.1.5 Conclusion

From the results, we can see that there are a lot of error sources. Some of them can be solved technically by filtering, but the start-up sequence is crucial so this stays without time drift. In test site a clock error accrued. A positive effect by using the GPS with the start-up sequence compared to the system with fixed position will be the 1-PPS if it is used correctly. This test are in this case not approved, even if the way the system was started was first turning on power and the RF before turning on the monitor mode.

7 Analysis

When looking into finding the best way to start up the PTT system for achieving highest accuracy in positioning, there are several sources of errors that have to be taken in consideration.

7.1 Qualitative description of error contributions

The system has several sources of errors, some that can be corrected for and some that cannot. To make the total system error as small as possible, all the errors that are possible to correct for should be corrected for. The building in the figure below has a receiver inside and a transmitter with a co-located GPS receiver to its left. A perturbation of the transmitter location in the vertical direction would have only a small effect on the range between the transmitter and receiver. However, if the perturbation of the transmitter location was in the horizontal direction, the range to the receiver would change by the entirety of the motion. Unobservable errors in the transmitter clock will always act in the direction of the receiver, regardless of their relative position. Thus, keeping good knowledge of transmitter clock drift is crucial for the overall system accuracy.

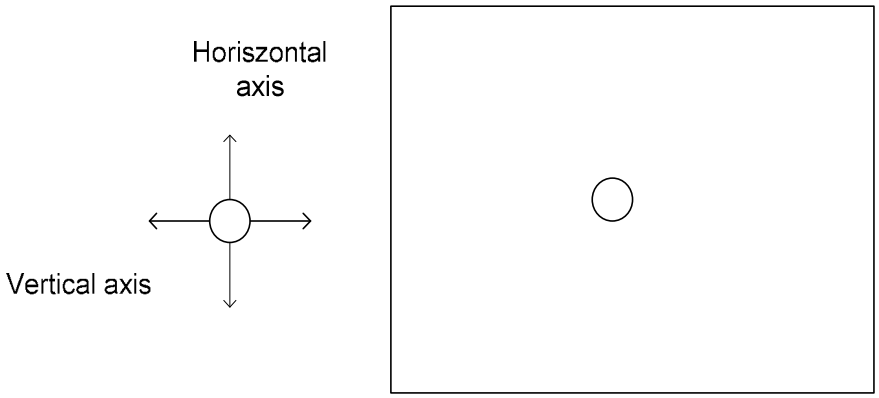

Figure 7-1; *Shows the PTT with the GPS receiver outside a building which will be used to locate the user inside the building. A time drift will make the GPS position move the vertical direction and change the distance to the end user remarkably.*

The thought of using the same clock for the whole system even if the system uses different frequencies that would come from the same clock in proportion to

use different oscillators for each module. With use of different oscillators you would have to measure the difference in time between each module and the measurement will always contain noise and then you have to do 2 measurements that contain Gaussian noise, the differential measurements will have twice the noise variance as one single measurement. Given that you have on single source then all the clocks will be coherently locked to each other.

GPS receiver uses a 10MHz clock signal that it up-converts to a 20 MHz and a 48MHz clock signal, with respective time periods of 50 nsec and 20 1/12 nsec.
During the operation, the GPS receiver will transmit a 1pps signal. This signal will be generated on the closest edge of the 10 MHz clock signal to true GPS time. The closest clock edge can be as far away as 50 ns seconds. This time error will cause an error in distance of 15m, which may far exceed the total system error requirement. To correct for this error we have to know where we are in the period, and calculate how far away from the real time. This time differential can is found by the GPS receiver in solving the set of GPS navigation equations, yielding user position and time offset from GPS time. This delta is how far into the future the 1pps will be sent out. The 1pps is used to synchronize the local receiver time to the true GPS time. The error source in this case is that the 1pps can only synchronize on whole clock cycles, so if the aberration in time is a float number it will be rounded to the closest half-cycle of the 10 MHz clock. The fractional part of the cycle can be found by solving the same navigation equation set and the correction applied to the user pseudo range.

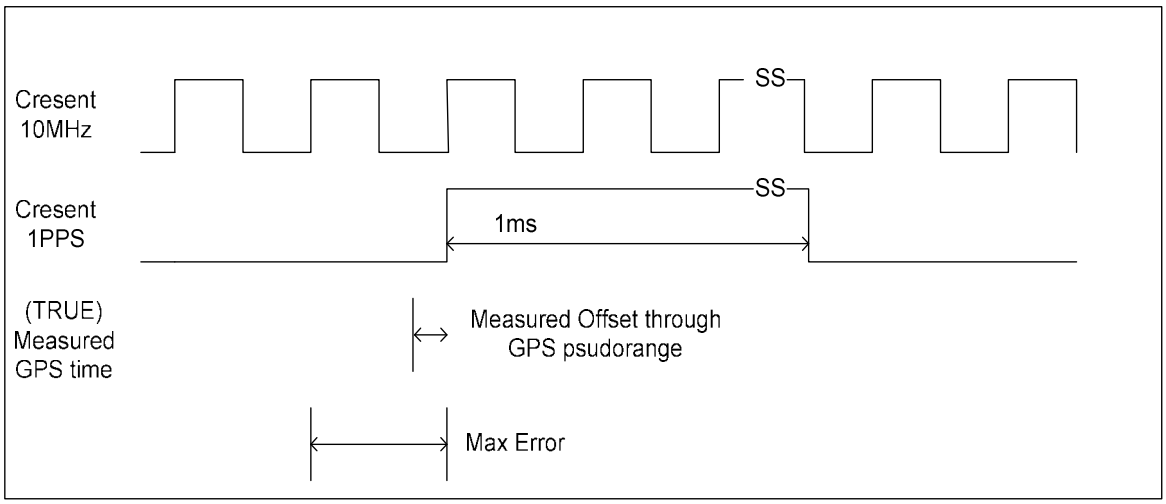

Figure 7-2; shows the Crescent 10MHz, Crescent 1pps, the true measured GPS time and the maximum error in this case.

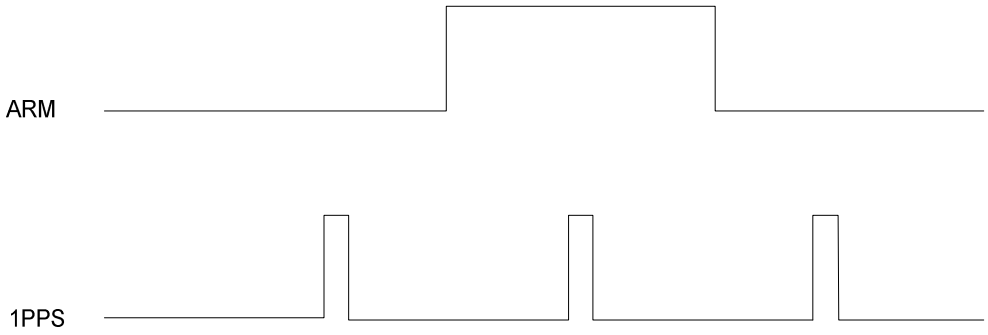

Figure 7-3; *shows the 1pps and the arm signals that starts ½ second before next expected 1pps, and ends a ½ second after it is finished.*

.

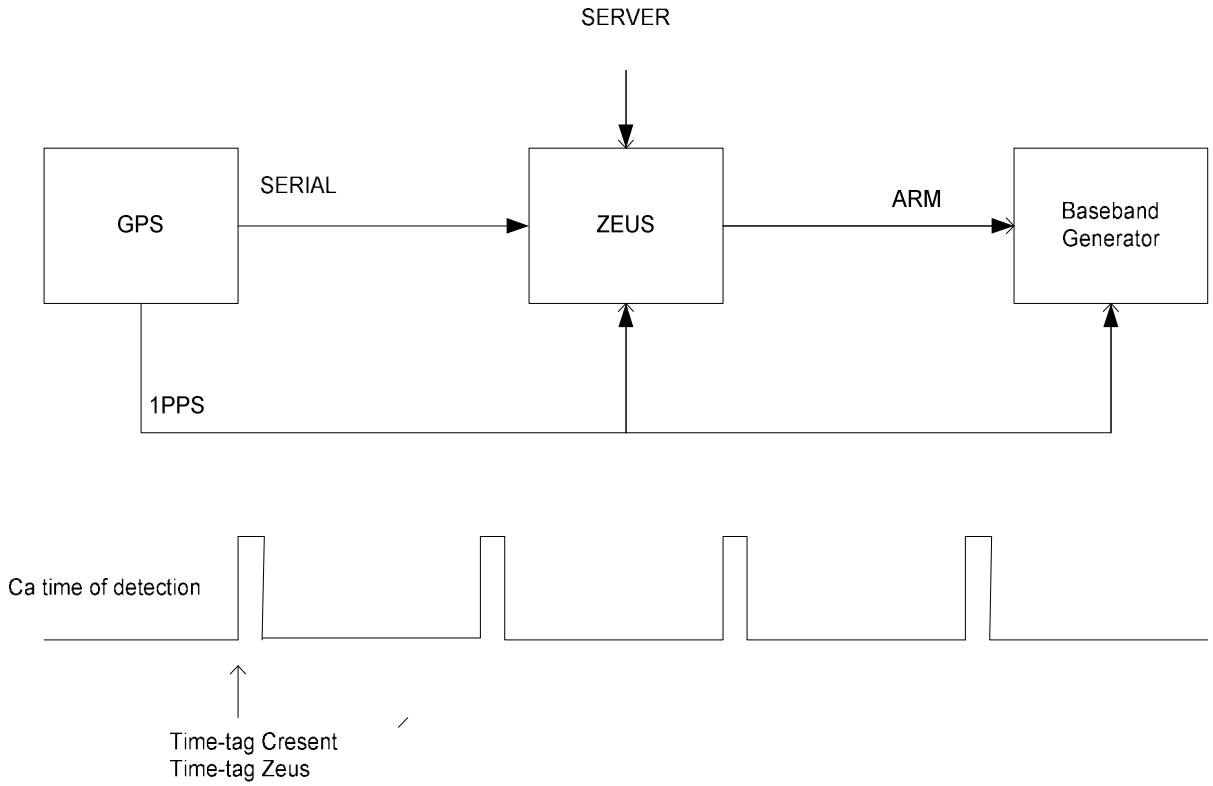

Figure 7-4; the signal float through the different modules and underneath you have the 1pps Signal.

7.2 Quantitative description of error contributions

The total system error for the overall system has to be < 6m

7.3 The Interface Controller

The interface controller is the unit that will make all the modules to communicate with each other. This will have the direct communications with a module and pass the data or commands in the right format to another module in the system. This unit was not chosen in the end of this project, and the specifications had to be set before a fitted unit could be found and bought.

7.3.1 Specifications

Before looking for a module, the specifications had to be decided upon. It was important to go through the whole system and discuss which modules needed and what kind of communication they were using.
After looking into what operations needed to be done and the power requirements for the system, we came to the conclusion to use an embedded computer with a PC-104 interface. Embedded systems are designed to do one or more specific task, rather than a general-purpose computer for multiple task, and since it is small it is easily to build in to a system. Embedded systems are easy to program and the firmware is stored in a read-only memory or flash memory chips instead of a disk drive. The PC-104 standard is intended for specialized embedded computing environments where applications depend on reliable data acquisition, in our case time. PC-104 is a standard form factor, and often used for similar purposes. Rosum is using this form factor in other products, and there is a plethora of companies supporting the standard.

Regarding power and communication specification, the systems requirements are what the decisions are based on. The PPT system is limited in how much power it can use, since it must work in a standard environment without the need for extra power generator when in use. In order to minimize the power and thermal load of the overall system the PC controller was limited to a total power consumption of 10W. To meet this specifications we pick a standard voltage and then the maximum of current that is possible when still following the limit of power (P=UI). It is important that the PC104 board meets the power requirements, other ways it will cause problems for the other parts of the system. The power requirements for the PC-104 board are as followed:

Voltage: 5VDC
Current: Max 2 A
Power: < 10W

The operating system has to be Linux to conform to other SW build environments in use at Rosum. The computer where the development will happen have

RedHat Linux installed, and that system will host a cross-compilation environment to the Arcom embedded Linux environment on the PC interface controller board.

The next step is to decide the different communications units needed. This interface controller is going to communicate with a GPS, pressure sensor and a baseband generator. The GPS communicates through a serial port, the pressure sensor needs I²C and we need a GPRS modem to be able to communicate with the server. The baseband generator can communicate over a USB port. So in total the PC-104 board needs to have the following communications features.

- 3 com ports, RS232,
- GPRS, quad-band, with Linux drivers,
- I²C for the pressure sensor
- GPIO for alternative use
- SPI for Exciter control
- USB for the Base-Band signal generator
- GPIO lines are used to the 1PPS

Initially, there was also a desire to include WiFi capabilities in the system, but it made the selection process harder since WiFi normally requires more power usage. Therefor this feature was dropped after talks with the end-users.

A last thing that should be considered is the price. This may not be the most important factor in a project of this scale, but the price should be <$500 in small quantities. Except from these specifications, there are no other limitations or requirements for the hardware.

7.3.2 The decision making process

There are several different suppliers of PC board and each vendor have several different PC-104 boards. To find an appropriate one there was a lot of products to look through. The internet was used to get an initial overview of available products, and then there was established phone contact with several different vendors. They often know more about their products and the different features than was displayed online. The result of the search was that very few boards had the I²C bus. The I²C is a bus created by Philips and to use the name and implement the technology you have to pay a royalty to Phillips to use it. It is possible to simulate a I²C bus on another port, but this can be an unwanted error source.

It was almost impossible to find something that could fill all the requirements, and we had to prioritize the feature set. We could choose to stack modules with different features, but we also wanted to keep it as simple as possible. We therefore had to look for which of the specifications we could be without. The less

important thing is the price, since all the other features are necessary. There is also a possibility to be without the I²C since it is possible to create that communication in software by doing bit-banging over a parallel port. This makes operations very slow, and we would most likely use a lot of time to troubleshoot the code and the implementation. We would also meet a new specification with the parallel port. We made a decision to make a slack at the price to find a board with all the specifications. It would anyway be cheaper to just by a board with all the features instead of using time to develop and add the features we need.

The board chosen was the Zeus board from Arcom; It has all the features needed. Their development kit has other features such as an LCD Touch screen so the firmware can be developed on board the Zeus if needed. As an added bonus, Rosum had used other products from Arcom, and there was already established a customer relationship including a dedicated point_of_contact. This made the ordering and the shipping processes a lot simpler.

7.3.3 The development tools

Besides figuring out what the start-up sequence look like and pick the right hardware, the software for the start-up sequence has to be developed. The software should be developed in C, since much of the other code in the project is written in that language. Emacs will be used as the editor, just because that is what I am most familiar with. We also realized that the Zeus board does not have emacs editor, but has both the editors' nano and vi.

To test the program and to verify some of the function on the Zeus board, there is a need to use testing equipment as an oscilloscope, function generator, multimeter and the other modules the interface controller are communicating with.

7.4 The software

All the software is developed on machines running the OS RedHat 8 or 9. Even if this is old versions, it is better to keep continue using this since all the other software Rosum is using is developed on that platform. In the PTT project the software developed will be ran from the Zeus board with a different Linux environment. Zeus uses another processor, ARM, and to make the software run there, all the code has to be cross compiled. Instead of using the regular gcc, we have to use arm-linux-gcc. This makes us change all the makefiles and include this compilator. This also makes the testing process more difficult. The code has to be developed on the desktop, since the Zeus has too little memory for the developing, then the code has to cross compiled and transferred over to the Zeus board for testing. Typical problems we meet if the software is tested out on the desktop environment, it might not necessary work on the Zeus board. The main

cause of this is different kernel tree on the Zeus and on RedHat 8 or 9. Then the same libraries may not be included or have a different path.

When developing the software, there were some issues that had to be figured out before it could be implemented in the program.
To detect the 1PPS from the GPS receiver the best thing to be is to use a for-loop. Since the 1pps pulse only lasts for 1ms we have make sure that the loops runs enough times to detect the signal. That is impossible to do the mathematics on, since we don't know the speed the program will be executed in. We therefore had to make a test in the final environment. We got the system time when the for-loop started and again when the for-loop ended. We sampled 10000 times just to be sure and saved all the data from the pins. Then we ran through all the data with some simple selection criteria's and found that the program used 4 sec to run the loop 10000 times, and it detected the 1pps 8-12 times. This told us that the sampling frequency was approximately 2,5kHz and to run the for-loop 2500 will sure to detect the pulse that happen once within a second.

8 Implementation

The start-up program is made in C, and is run from the Zeus board. The Zeus board id the interface controller and all the communications between the different modules goes through it. This chapter will discuss how the software and hardware works and how they are they are connected to each other. It is presented in details related to how the configure the ports. And which pins are related to what function.
The software consists of the header files, functions and the main() program. The header files include the libraries and definitions, the functions, are used for the I2C communication, the GPS tag and the serial communication.

Figure 8-1 shows the different interface in the system, and which part speaks togheter.

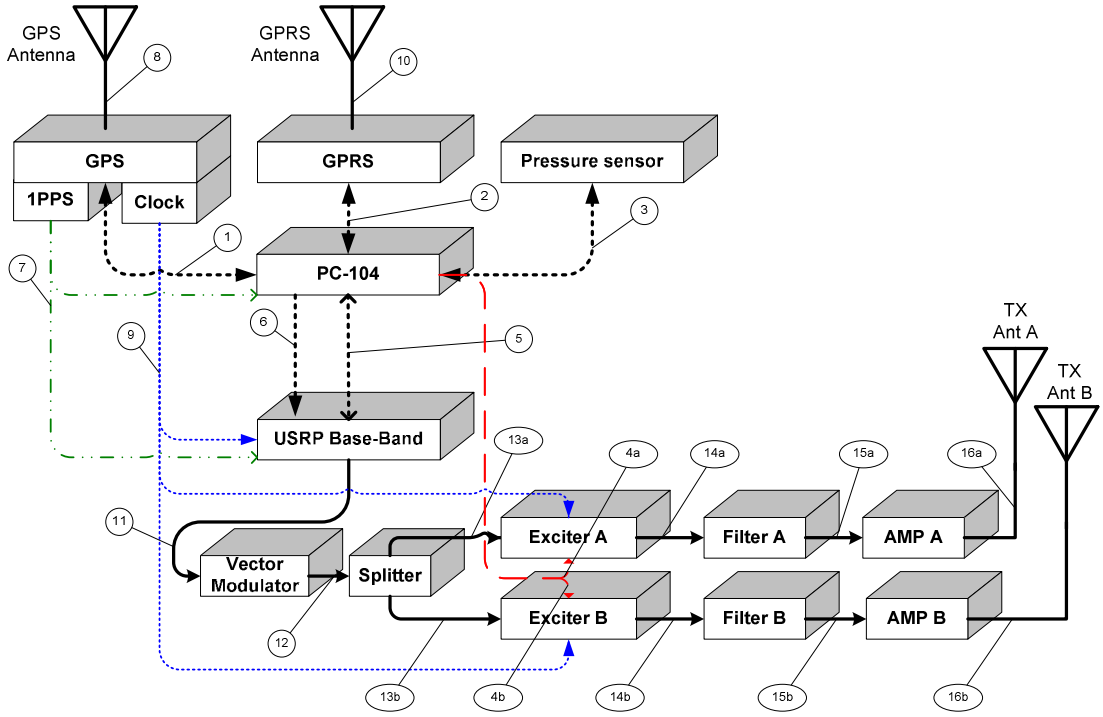

Figure 8-1, PTT HW Modules and Interfaces.

Interfaces is specified in Appendix 1. Only the one related to the part of the implementation done in this specific paper are specified. That includes all the interfaces from the PC-104, ZEUS to GPS, GPRS, Pressure sensor, USRP Base-Band and the 1PPS.

8.1 The start-up sequence in software

8.1.1 Included header files

To get access to the all the functions needed, some libraries and headers has to be included. This libraries and headers are also where all the functions are included.

A table over all the headers included in the program	
#include	**Description**
stdlid.h	The C Standard General Utilities Library defines several general purpose functions, including dynamic memory management, random number generation, communication with the environment, integer arithmetic's, searching, sorting and converting.
string.h	This header file defines several functions to manipulate *C strings* and arrays.
stdio.h	This header includes the standards in and out functions, like printf.
fcntl.h	File control options
unistd.h	This header defines miscellaneous symbolic constants and types, and declares miscellaneous functions. This include the function write() and read() that we're using to write/read to the registers.
sys/time.h	This header includes timeval functions that make us able to get the system time.
sys/ioctl.h	This header includes the ioctl functions we use in select_I²C_bus
error.h	This header includes the error messages used as perror()

Table 8-1 explains the header files used in the program.

8.1.2 Functions

int **open_i2c_bus(**const char *devnode**)**

This function has as intentions to open the I²C bus for communication. To accesses the bus use /dev/i2c-0, that is a path which is included in all newer Linux kernels. The open() functions is used, and it creates a file descriptor that will be used whenever we're accessing the I²C bus for communication. If something goes wrong, the function will print an error message to the screen.
The function return the integer fd. I²C

void **select_i2c_slave**(int fd, uint16_t slave)

This function is used to select which salve device we want to communicate with. Since the I²C bus is a true multi master and has the possibility to have several slaves, PC/equipment to control. In our system it has 5 different slaves and it is therefore need for an option on how to choose address the slave who are communicating with. In this specific program, there is only communication with one slave, the GPIO ports with the address 0x21. To select slave, the system call *ioctl(),* who allows an application to control or communicate with a device driver outside the usual read/write of data is used. The call are using the file descriptor returned in the open_i2c_bus() function, the defined slave address(a request code number), and the address for the selected slave. This is a void function and doesn't have a return value. When it comes to error messages, the same here as in the other function. If ioctl doesn't return the right value, the program displays an error message on the user monitor.

uint8_t **read_i2c_bus**(int fd, int r)

This function is used to read from the registers. But to be able to read, you first have to tell the function what you want to read, then you can read the data in the registers. The function starts by using the write() function with use of the file descriptor, the data register you want to write to and the value that are written to the registers. The return value from the write() function is usually equal to the *nbytes* argument, otherwise an error has occurred, and an error message will be printed on the screen.
With first using the write function, the MAX7313 was told what we wanted to read and uses the read() function to read out the value saved in the register, or the data from the 1pps input. The return value from the read_i2c_bus is the data read.

void **write_i2c_bus**(int fd, uint8_t val)

The write function calls the write() function with the file descriptor, the registers address and the value of the operation. The write function returns a value, there is a check to see and if an error is detected an error message is printed. The write_i2c_bus is avoid function and has therefore no return value.

GPS TIME TAG FUNCTION

This function is developed by another project member. This has to be included and fitted into my program. In the attached version of the program, this part is only referred to.

Serial communication

Establishes the path and enables the serial communication with the GPS from the Zeus board through the path /dev/ttyS2 as default on the Zeus board, but /dev/ttyS0 is default for the desktop. Most computers do not have 3 COM ports, and there has to be an opportunity to run the code there too.

main()

The main program starts by initializing the variables used, and sets some of them to a default value. So as long as they don't get any other message, or something goes wrong they will have this value. The first default is the communication part; this will not be changed as long as the program is run on Linux computers. Both the slave and reg variables are sat default to -1, so if they don't get a value and write() or read() function is executed there will come an error message.

The next operation is to configure all the GPIO ports by writing the wanted value, as shown in the chapter about configuration of the GPIO. Since the MAX7313 chip let one operation happen at the time, the write function has to be called every time you want to write a new value to a register. The only operation that can be done one is to open the i2c bus and setting the slave. After the integer fd is set, the file descriptor, this is used for all communication one over the i2c bus. The main program only talks to one slave, so this slave is given by the program. If communication with a second slave is needed, some changes in the code have to be implemented.
The main program calls the write functions 5 times just to set all the registers.

The next step is to detect the 1pps. This is done by a for-loop who is run 2500 and every time checks if the 1pps input signal is high. If the 1pps is detected, the system time is read and saved in a variable. After the for-loop is finished, the get-gps-timetag function is called. The function returns the gps-timetag who is related to the system time. This is done to find out the difference in the clock on the Zeus board and the timetag on the GPS.

The next operation is to request/receive RF on

The next step in the program is a countdown timer with time to start from server. Before RF on enables, there has to be a delay for 10 seconds. To make this happen there has been created count-down timer. This is a for-loop with a 1000msec delay in. This I s a countdown to send the ARM signal high. This should be set high ½ second before the 1pps edge and end a ½ second after the 1pps edge. Therefore after counting down the first 8 whole seconds, there has to be a delay on 500 ms. now the ARM signal can be set high. The count-down for-loop includes an if-test which makes the program go and decode a GPS packet and find the time tag every time, the time to count down is not equal to 0.

Start Monitor Mode

8.1.3 Configuration of the GPIO

The program uses to different ports from the Zeus for communication in this program, the serial port for the GPS and the GPIO lines for the GPS, Baseband generator and exciter. To be able to control the different pins, there is a MAX7313 connected that we have to talk to. The chip has 17 registers, where changing the register value affects the behavior of the pin. The registers used are showed in table 5.2

Registers	**Address**
Input registers P7-P0, P15-P8, It is used to read the inputs on the wanted pins	0x00, 0x01
Outputs, P15-P8, P7-P0, used when something is written to the pins	0x02, 0x03
Ports configuration registers, set the port to inputs or outputs	0x06, 0x07
Blink phase 1 registers	0x0B, 0x0C
Master, O16 intensity register	0x0E
Configuration register	0x0F
Outputs intensity registers.	0x10-0x17

Table 8-2, description and address for the registers used to control the GPIO ports.

The main program starts with initialize the pins in the GPIO connector. An overview of how it will be used in table 8-3.

An overview over J7 connector							
PIN	**SN**	**I/O**	**Function**	**PIN**	**SN**	**I/O**	**Function**
1	+5V			2	+5V		
3	P0	O	Clock(SPI) exciter	4	P1	O	Data, MISO, Exciter
5	P2	O	Chip Select, Exciter	6	P3	O	Chip Select, Exciter
7	P4	O	Chip Select	8	P5	O	Chip Select
9	P6	O	Chip Select	10	P7		
11	GND		Exciter	12	GND		USRP,GPS
13	P8	O	ARM_PPS,	14	P9	I	1PPS from GPS
15	P10			16	P11		
17	P12			18	P13		
19	P14			20	P15		

Table 8-3, an overview over GPIO connector, and what the different pins are connected to.

We will start with configure, if the pins are going to be output or inputs. When writing to the ports configuration registers, 0 = output and 1 = input. That gives us the values

Details	Register	Bin value	Hex value
Configures P7-P0	0x06	0000 0000	0x00
Configures P15-P8	0x07	0000 0100	0x04

Table 8-4, shows the value of the configuration we need. Set all unused pins to output.

Since we want to make some of the GPIO lines work as a SPI bus, we have to configure the pins used to act like a SPI bus. The SPI bus has a clock, MOSI, MISO and chip select(s). The MAX7313 lets us make a clock from the PWM opportunity it has. It becomes a very slow clock, but since we don't need a high speed opportunity here, that is enough. The clock will be on P0, pin 3. To make this happen we has to set up the registers as followed. To make the clock, there has to be set up a combination with the master intensity duty cycle set to 16/16, who gives us the frequency. The individual intensity gives the pulse width, and has to be set to 7/16. The values used to create the clock signal are given in table 8-5.

Details	Register	Bin Value	Hex value
Enables PWM oscillator	0x0E	1111 1111	0xFF
Enables interrupt and blink	0x0F	0000 1001	0x09
PWM duty cycle set to 8/16 on pin 3	0x10	0000 0111	0x07

Table 8-5, shows the registers and the value they need to have to make a clock signal for the SPI bus.

8.2 Zeus Module

The software discusses in the earlier part of this chapter has now to be implemented on the Zeus board, with the connected modules. The software has to be downloaded to the Zeus, since there is from there the program will be ran. There is some problems here, since the Zeus board is using a different kind of processor that the computer where the code is built. Regular desktop computers are using x86 processors, while the Zeus is using an ARM processor. To make all programs work on Zeus we has to cross compile all the code to make it play along with the other processor type. The program is transferred over to the Zeus and executed from there.

8.2.1 GPIO

The GPIO ports are the J7 connector on the Zeus board, with 16 general input/ports. The GPIO is configured as showed in table 5.2.

8.2.2 Serial Port

The Zeus board has seven serial communication ports available, three of them area already used by already onboard features. But there is the four additional high-speed, 16550 compatible serial UARTs implemented by an EXAR ST 16C554Q Quad UART device. Two of these channels (COM1, COM2) can be used as standard RS232 serial interface and the COM3 can be configured as a RS232 or RS485/422. The GPS is connected to the COM3 and the port is therefore configured as a RS232. The port address is 0x11000000-0x110000F, the IRQ is the GPIO11, the FIFO depth RX / TX is 16/16 and the signals RS232: Rx, Tx, CTS, RTS, RI, DSR, DCD, DTR

The COM3 is buffered to RS232 levels with +-15kV ESD protection. RS232 interface supports full handshaking and modem control signals. The maximum baud rate on this channel is 921.6 Kbaud. COM3 RS232 interface is connected to the connector J19.

It is on the serial port the Zeus board receives the data from the GPS. The serial port at the Zeus board's side is DTE, data terminal Equipment.

8.3 Crescent Module

8.3.1 GPIO

GPS side 3.3V TTL
1PPS, P900: 15
GND, P900: 8z

8.3.2 Serial Port

The Crescent Serial port interface level is 3.3 V CMOS.
The crescent has four communication ports referred to as A, B, C and D. Three first of them are fully independent and you can choose among different message output and rate. PORT A is used to communicate with the Zeus board. The communication sent is the GPS data in binary format. The data is decoded on the Zeus by the GPS decode program. The different Pins used in the serial communication are:

Signal	Pin	Description
Port A Tx, P900	9	The transmitted data
Port A Rx, P900	10	The received data
Port A Gnd, P900	8	Ground

Table 8-6, shows the Crescent pin out configuration. When looking at the board and 'P900' is facing up, pin two is beside (to the right) of the first pin. Pin three is below the first pin, and so on.

8.4 Start-Up Sequence

The start-up sequence showed in the figure 8.2 is what the program is based developed out from.

The first that happens is that the PC104 gets turned on and the program is run. It starts by requesting/receive in aiding information and starts the GPS receiver. As soon as the GPS gets the first fix it starts sending the out a 1pps signal. The next step for the program is to detect this signal and it runs a for-loop whit a samplings rate on 2500KHz, then you're sure to detect the 1pps, with a pulse with on 1pps. If the signal is not detected, it keeps looking for it.
When the 1pps are detected, the time of detection on the Zeus board is picked out by help of a time function. This is done to relate the to time tags roughly to each other.
The next step is to decode the GPS packet and find the GPS time tag. To do this, the program calls a function made by Ph.D. student Gabriel Wong. Then the time tag from the GPS is related to the time tag on the Zeus board. The next is to request/receive RF from server.
The next step is to start the countdown timer to start from server. If the countdown doesn't end, it returns to decode the GPS packet and finding time-tag.
When the countdown is finished, it counts own 8.5 seconds

1. Detect 1pps on Zeus
2. Time tag detection of 1pps, relative to Zeus module
3. Decode GPS message
4. relate GPS time tag to Zeus system time tag
5. receive time-of-transmission command from server
6. count down from current time to tot
7. assert arm signal ~1/2 second before expected arrival of 1pps
8. De-assert ARM signal ~ ½ second after 1pps.

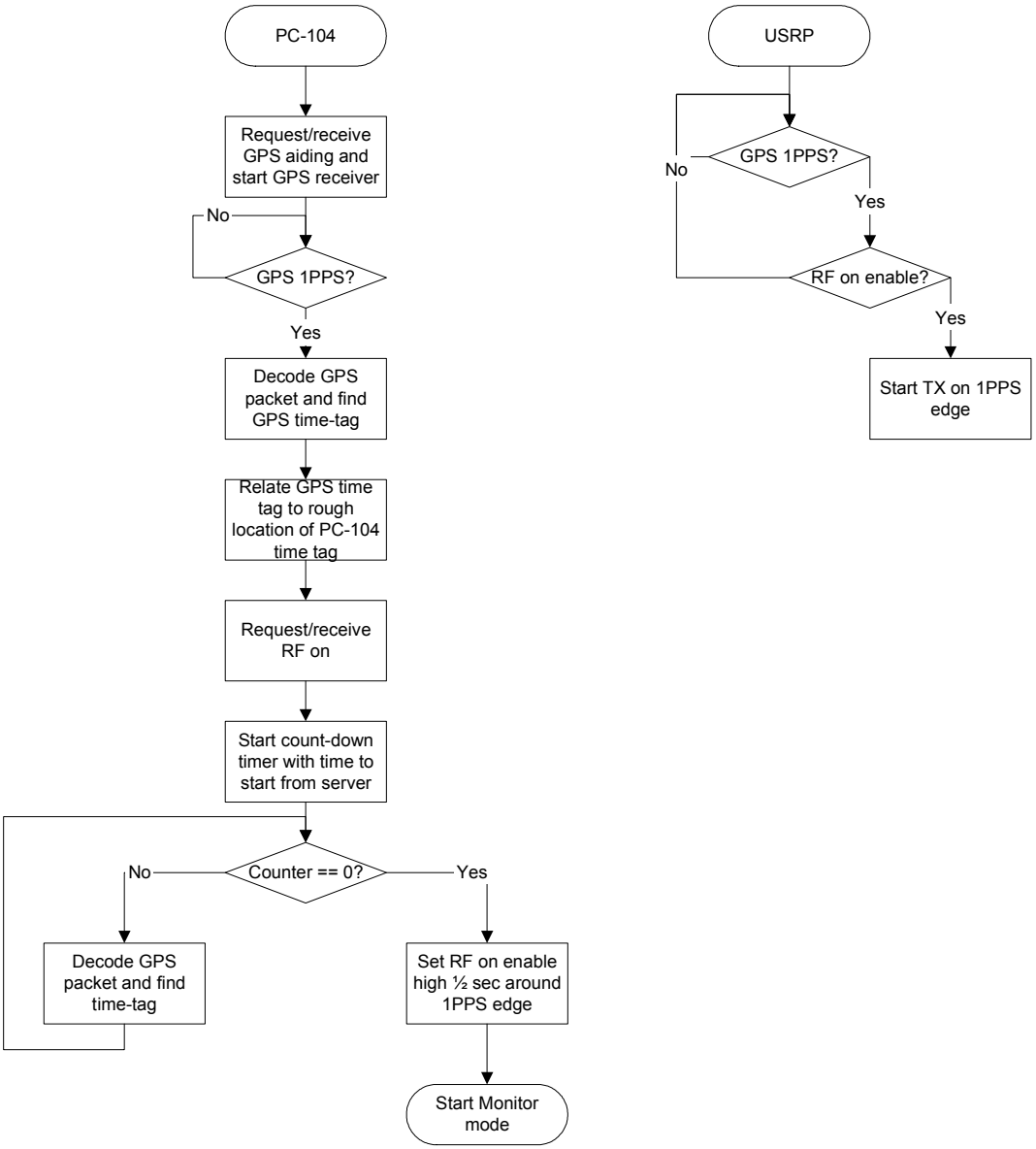

Figure 8-2, the start-up sequence for the PC104 and the USRP.

9 Overall Discussion

To reduce timing error it was concluded to use the GPS time to synchronize the PTT transmitter. After the tests and use of theory it the result of the best start-up sequence was to start with requesting information from the GPS, and when receive the reference signal 1PPS, decode the packet and compare the GPS time with the system time in the interface controller. When the aiding info of the position of the baseband transmitter was received, the RF transmission could start. There was a set countdown before entering the monitor mode. This was done, so the absolute position of the transmitters was as correct as possible before we started to make the positions of the receiver. Each separated baseband transmitters were using absolute positions, while the PTT receiver was based on the principles of Differential GPS which also cause fewer errors.

If the monitor mode was turned on before we had the absolute position of the transmitter in relation to each other, there will be a position error at the receiver. To find the position of the receiver it is based on the distance to the baseband transmitter, and if this position is wrong, the first hits can be far away. This will affect the averaging function implemented as a help function in the receiver.

Without the 1 PPS and the GPS time, it could be a time difference between the baseband transmitters and Interface controller and a time drift will as experienced in the practical test cause a positioning error.

Since this is a new system, this is not any similar cases where we can compare the results, and have to base the findings on theory and actual observations. The conclusion is that the system works as good enough with the chosen start-up sequence, and if the accuracy errors are too high in the real life system, this has to be corrected in other parts of the system and not in the start-up sequence.

The hardware chosen was based on specification related to the rest of the system, even if it was eliminated by the end customer; it could have been beneficial with a Wifi connection. WiFi would have a higher bitrate than GPRS and could communicate with the server faster. The power priority resulted that this functionality was left out in the final system. A thing which could have been beneficial in the development process was to have a PC104 with a different processor. There was an ARM processor, but all the development tools were using a different kind of processor. It could have reduced the development time, and it would have been easier to simulate a full test of the program. Some of the errors was also a result of difficulties in the cross compilation. To find this, it would have taken more time on finding the correct hardware.

The program was developed in C, which are a preferred and the most optimized language when programming embedded systems. This is the same language used in the prior development of the system, so there was no better choice of

development tools. Emacs was used as an editor when developing on the Linux machines, it was chosen because the existing knowledge of this editor. On the PC104 board, Emacs was not available, and VI had to be used. This made the process take longer time, since the commands were different, and to have to relate two editors was more time consuming.

When comes to optimization and flexibility of the software, some things migh have been improved. But because of the time limitations, and the fact that full system is not tested, it is hard to see all the areas of improvement. The program is not very flexible regarding to changing hardware. If the hardware chosen goes out of production, the code has to be re written, since it is addressing the ports directly, and this might be changed on another hardware device. But the architecture and the sub functions are present so it will still be easier than the first time. It will also be necessary to find a new sampling rate, since this is based on the speed on the specific hardware, so it might be different on the next.

There could have been possible to use different interfaces when communicating to the other devices in the system. But since some parts were already chosen when the assignment was started, an important part was actually to find the correct hardware related to the already specified hardware. The GPIO – which is a general port could be used to communicate with different devices, all dependent of the use of it. It would have been possible to simulate the I²C bus, but since hardware had it included, there was no need to simulate. The simulated version could have been slower, since it was most easy to simulate the slow version of the bus. When more communication forms have to be developed, it results in higher probability of errors.

9.1 Further work

Since this paper is based on only one part of the project, the project will continue and further work will be done.

When all the parts is done there will be possible to do a more realistic testing. The program is today only tested towards the specific interfaces.

The project has to be tested and demonstrated at customer's site, with the belonging humidity, weather and area. Then the system will be tested on the 25 story building.

10 Refrences

[1] Dana, Peter H.(1994) Department of Geography, University of Texas at Austin. Localized 20.02.2011 on the World Wide Web: http://www.colorado.edu/geography/gcraft/notes/gps/gps_f.html

[2] Parkinson W Bradford & James J Spilker. (1996).
Global positioning System: Theory and Applications, volume 1.
Aeronautics and Astronautics Inc.

[3] Blewitt, Geoffrey. (1997). Basics of the GPS technique: Observations Equations,
Dept. of Geomatics, University of Newcastle, UK
Located 03.03.2011 on the World Wide Web: http://www.nbmg.unr.edu/staff/pdfs/Blewitt%20Basics%20of%20gps.pdf
Published in the textbook “Geodetic Applications of GPS”, Published by Swedish land survey.

[4] Rizos, Chris.(1999) Principles of satellite positioning. SNAP-UNSW. Localizes 20.02.2011 on the World Wide Web: http://www.gmat.unsw.edu.au/snap/gps/gps_survey/chap1/132.htm

[5] Do, Ju-Yong & Rabinowitz, Mattehew and Enge, Per. (2007) Stanford University.
Multi-Fault Tolerant RAM Algorithm for Hybrid GPS/TV Positioning. Localized 28.03.2011 on World Wide Web:
http://waas.stanford.edu/~wwu/papers/gps/PDF/DoIONNTM07.pdf

[6] Davies, Chris (2009), USA
Rosum TV-GPS triangulates urban and indoor position from TV towers. Localized 20.02.2011 on the World Wide Web: http://www.slashgear.com/rosum-tv-gps-triangulates-urban-indoor-position-from-tv-towers-0933522/

[7] Calais, Eric.(unknown). The Global Positioning System Principles of GPS positioning GPS signal and observables Errors and corrections Processing GPS data GPS measurement strategies Precision and accuracy. Localized 23.02.2011 on the World Wide Web: http://web.ics.purdue.edu/~ecalais/teaching/geodesy/GPS_observables.pdf

Attachment 1: Interfaces

1. GPS to PC-104 Data
    a. Interface: RS-232
        i. PC-104: DTE
        ii. GPS: DCE
    b. Direction: 2-way
    c. Level converter needed
    d. GPS side CMOS 3.3V
        i. Port A Tx, P900: 9
        ii. Port A Rx, P900: 10
        iii. Port A Gnd, P900: 8
    e. PC-104 side, RS-232
        i. COM3 Tx, J19: 15
        ii. COM3 Rx, J19: 13
        iii. COM3 Gnd, J19: 19

2. GPRS to PC-104 Data and control
    a. Interface: PC-104 bus
    b. Direction: 2-way
    c. Built into Zeus board

3. Pressure Sensor
    a. Interface: I2C
    b. Direction: 2-way
    c. Zeus side:
        i. SCL, J19:1
        ii. SDA, J19: 2
        iii. GND, J19: 3
        iv. PWR (3,3VDC), J19:4
    d. Pressure sensor side
        i. SCL, Pin 9
        ii. SDA, Pin 10
        iii. GND, Pin 6
        iv. PWR(3.3VDC), Pin 5

4. USRP Data
    a. Interface: USB 1.1
    b. Direction: 2-way
    c. Zeus side
        i. J8
    d. USRP side

        i. J401

5. USRP Control
    a. Interface: GPIO
    b. Direction: From PC-104 to USRP
    c. Level changer needed
    d. Zeus side 5V TTL
        i. ARM_PPS, J17:13
        ii. GND: J7:12
    e. USRP side 3,3V TTL
        i. ARM_PPS, J667: 35
        ii. GND, J667:37

6. 1PPS
    a. Interface: GPIO
    b. Direction
        i. From GPS to PC-104
        ii. From GPS to USRP
    c. Level converter needed
    d. GPS side 3.3V TTL
        i. 1PPS, P900: 15
        ii. GND, P900: 8
    e. Zeus side 5V TTL
        i. 1PPS, J7: 14
        ii. GND, J7: 12
    f. USRP side 3.3V TTL
        i. 1PPS, J667: 33
        ii. GND, J667: 37

Attachment 2: The source code

```
/*
 * i2c-reg utility.
 *
 * Copyright (C) 2006 Arcom Control Systems Ltd.
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 * (This code for the i2c has been used in the code for the interface controller made by
*Ida Elders, Rosum Corp)
 * v2 13 APR 2007 WJF Arcom
 * - Updated to support register bit masking (read-modify-write)
 */

#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <errno.h>
#include <stdint.h>
#include <time.h>
#include <sys/time.h>
#include <sys/ioctl.h>

#define VERSION      1
#define ISSUE                    2

#define PACKAGE "i2c-regs"

#ifndef I2C_SLAVE
#define I2C_SLAVE     0x0703  /* Change slave address              */
#endif

//defining the registers used
#define PORT_CONFIG_P7_P0  0x06;
#define PORT_CONFIG_P15_P8  0x07;
#define READ_INPUT_P7_P0  0x00;
#define READ_INPUT_P15_P8  0x01
//#define 1PPS(x) ((x << 8)

/*struct timeval {
```

```
  time_t tv_sec;
  suseconds_t tv_usec;
};
*/
int open_i2c_bus(const char *devnode)
{
        int fd;

    if ((fd = open(devnode,O_RDWR)) < 0) {
        perror("open i2c bus");
        exit(1);
    }
       return fd;
}

void select_i2c_slave(int fd, uint16_t slave)
{
    int addr = slave;
    if (ioctl(fd,I2C_SLAVE,addr) < 0) {
        perror("select slave device");
        exit(1);
    }
}

uint8_t read_i2c_register(int fd, int r)
{
    unsigned char buf[3];

    buf[0] = r;

    if ( write(fd,buf,1) != 1) {
        perror("read_i2c_register, write");
        exit(1);
    }
    if (read(fd,buf,1) != 1) {
        perror("read_i2c_register, read");
        exit(1);
    } else {
        return buf[0];
    }
    return -1;
}

void write_i2c_register(int fd, uint8_t r, uint8_t val)
{
    uint8_t buf[2];
```

```
        buf[0] = r;
        buf[1] = val;
        if ( write(fd, buf, 2) != 2 ) {
                perror("write_i2c_register, write");
                exit(1);
        }
}


int main(int argc, const char ** argv)
{

           char *devnode = "/dev/i2c-0";
           uint16_t slave = -1;
           uint8_t reg = -1;
           uint8_t read_value;
           uint8_t (*read_register)(int fd, int r);
           void (*write_register)(int fd, uint8_t r, uint8_t val);
           unsigned int array[10000];
           unsigned int x;
           int numberofpps;
           //        int numberofsec;
           float diffusec;
           float diffsec;
           float timesample;
           unsigned int ppsarray[15];
           //        int timearray[15];

           slave = 0x20;
           reg = 0x06;
           read_register = read_i2c_register;
           write_register = write_i2c_register;
           int fd;
           fd = open_i2c_bus(devnode);
           select_i2c_slave(fd, slave);
           uint8_t value = 0x00;
           printf("Writing 0x%02x\n", value);
           write_register(fd,reg,value);

           reg = 0x0E; // Master, O16 intensity register
           value = 0xff; // Configured as a mix of static and PWM ouputs, with PWM
outputs using different PWM settings, Master intesity duty cycle is 15/15, O16 intensity
duty cycle is 16/16, no PWM
        printf("Writing 0x%02x\n", value);
           write_register(fd,reg,value);
```

```
      reg = 0x0f; // Configuration register
         value = 0x09; // enables data change interupt-int/O16 output is controlled by port
input data change, blink enabled
         printf("Writing 0x%02x\n", value);
         write_register(fd,reg,value);

         reg = 0x10; // Output intensity register
         value = 0x07; // PWM duty cycle  output set to  8/16 on pin 3, SPI-clock
         printf("Writing 0x%02x\n", value);
         write_register(fd, reg, value);

         /*Reading the 1PPS from the system*/
         reg = 0x06 // portconfig p7-p15
           value = 0x00 //sets all the pins to output
         reg = 0x07; // portconfig p8-p15
         value = 0x40; // sets pin 9 to input, sets the rest to output
         write_register (fd, reg, value);
         reg = 0x01; // Input registers, read input ports 8-15
         numberofpps = 0;

         int tid;
         struct timezone timeZoneBefore;
         struct timeval timeBefore;
         struct timezone timeZoneAfter;
         struct timeval timeAfter;

         tid = gettimeofday(&timeBefore, &timeZoneBefore);
         // put in an errorthing


         for (x = 0; x < 10000; x++) //waiting to detect 1pps from the gps
           {
            reg = 0x01; // Read inputs from p8-p15
             read_value =  read_register(fd,reg);
             array[x] = read_value;
             //    printf("%x\n", array[x]);
           }
         tid = gettimeofday(&timeAfter, &timeZoneAfter);
         diffusec = timeAfter.tv_usec - timeBefore.tv_usec;
         diffsec = timeAfter.tv_sec - timeBefore.tv_sec;
         printf("diffsec = %f\n", diffsec);
         timesample = ((diffsec * 1000000) + diffusec) / (10000) ;
         printf("timesample = %f\n", timesample);
         int z;
         int y;
         z = 0;
```

```
        y = 0;

        for(x = 0; x < 10000; x++)
          {
           if (array[x] == 0xff)
             {
                 numberofpps++;
                 ppsarray[y] = x;
                 printf("ppsarray[%d] inneholder sample nr %d\n", y, ppsarray[y]);
                 y++;
              }
          }
        printf("alt i array %d", ppsarray[y]);
        for(y = 0; y < 15; y++)
          {
           ppsarray[y] = (ppsarray[y] * timesample);
           printf("The %d. 1pps occured %d (usec) after program started\n", y ,
ppsarray[y]);
          }

        //        printf("Number of 1pps counted in 10000 times are  %d\n", numberofpps);
        //        printf("The program used %d s to sample 10000 times", numberofsec);
        printf("time before: %ld (sec) after: %ld before: %ld (usec) after: %ld (usec)\n",
timeBefore.tv_sec, timeAfter.tv_sec, timeBefore.tv_usec, timeAfter.tv_usec);




        // call the function who decode GPS packet and find timetag
        //relate GPS timetag to rough location oc PC104 timetag.

        //request/receive RF on
        //start countdown timer with time to start from server


        for(s = 8; s > 0; s--) // countdown to star set RGF enable and start moitor mode
            {
                hent data fra gabriels function
                delay(1000);
            }
                dealy(500);


    return 0;
}
```