# Packet detection and timing estimation on an UWB physical layer

Robin Scheibler

**Semester project**

under the supervision of
Ruben Merz

Prof. Jean-Yves Le Boudec
EPFL-IC-LCA2

February 19, 2007

## Abstract

The original goal of this project was to implement a synchronization algorithm for ultra wide band (UWB) communication and test it on traces from an impulse radio UWB testbed with interference.

However, due to some unexpected problems I had to abandon my experiments with the traces from the testbed. Therefore I first concentrated on writing down properly the effect of the channel on the signal with respect to the timing estimation problem.

Then, the final part of the project was an implementation in Matlab of two variants of the architecture from ST Microelectronics for the 802.15.4a standard and its dedicated synchronization algorithm. This implementation was then used to derive the probability of misdetection by simulation as a function of the signal to noise ratio (SNR).

# Contents

**Report** **3**

**1 Synchronization on the testbed** **3**

1.1 Goal . . . . . . 3

1.2 Solutions . . . . . . 4

1.2.1 Implementation of the verification phase . . . . . . 4

1.2.2 Energy detection . . . . . . 4

1.2.3 Search neighborhood . . . . . . 5

1.3 Achievements and future work . . . . . . 6

**2 Effect of the channel** **6**

2.1 Goal . . . . . . 6

2.2 Transmitter, Receiver and Channel model . . . . . . 6

2.2.1 Transmitter . . . . . . 6

2.2.2 Receiver . . . . . . 7

2.2.3 Channel model . . . . . . 8

2.3 Baseband equivalent model . . . . . . 8

2.3.1 Baseband pulse . . . . . . 8

2.3.2 Baseband channel . . . . . . 9

2.3.3 Received signal . . . . . . 9

2.4 Future work . . . . . . 10

**3 Synchronization on a 802.15.4a architecture** **11**

3.1 Goal . . . . . . 11

3.2 Discretization of the system . . . . . . 11

3.3 Thresholds derivation . . . . . . 12

3.3.1 First scheme . . . . . . 12

3.3.2 Second scheme . . . . . . 12

3.4 Probability of misdetection . . . . . . 16

3.5 Achievements and future work . . . . . . 16

**4 Conclusions** **19**

4.1 Skills acquired . . . . . . 19

4.2 Major events . . . . . . 19

4.3 Self-assessment . . . . . . 19

**Acknowledgements** **21**

**Appendix** **22**

**A Algorithms** **22**

A.1 Power Independent Detection . . . . . . . . . . . . . . . . . . . . 22

A.2 802.15.4a synchronization from ST . . . . . . . . . . . . . . . . . 22

**B 802.15.4a architecture from ST** **24**

B.1 Architecture of the receiver . . . . . . . . . . . . . . . . . . . . . 24

B.2 Preamble construction . . . . . . . . . . . . . . . . . . . . . . . . 24

B.3 Detection threshold for the first scheme . . . . . . . . . . . . . 25

# Report

## 1 Synchronization on the testbed

### 1.1 Goal

The original goal was to evaluate the performance of the algorithm described in [4] in practice. The performance evaluation would have been done with traces acquired from the testbed described in [1] A sketch of how the algorithm works is given in appendix A.1.

This goal should have been achieved in three steps :

1. Efficient implementation of the algorithm.

2. Experimental derivation of the critical parameters needed by the algorithm to achieve good performance in a practical use. A training set of traces from the testbed would have been used to achieve this step.

3. Derivation of the probability of misdetection on another set of traces. This would have been the metric for the performance evaluation.

Due to technical problems with the testbed, only the first step was achieved. The main problem was that the testbed was lacking a mechanism that would allow to automatically check if the algorithm correctly synchronized. Therefore no systematical experiment could be done. The only way to see if synchronization succeeded of failed was to graphically check if the template was coinciding with the received pulses. This method is obviously far from efficient and not precise at all and hence step 2 and 3 couldn't be done.

However, here is a brief overview of the work done during step 1.

1. **Implementation in C**

   I was provided at the beginning of my project with a skeleton of a Matlab implementation of the algorithm. The problem was that the critical part of the algorithm included three imbricated loops which are extremely costly in Matlab. So I rewrote this part in the C language and compiled it as a MEX-file to use it in Matlab.

2. **Implementation of the verification phase**

   In the code I was provided with, the algorithm was only partially implemented. The verification phase was not implemented and then my next task was to implement it. The details of the implementation are described in 1.2.1.

3. **Mitigating bad quality of the traces**

   As an effect of the technical difficulties mentioned earlier, the traces were seriously lacking quality. In order to mitigate this I tried to add some mechanisms that are described in 1.2.2 and 1.2.3.

## 1.2 Solutions

### 1.2.1 Implementation of the verification phase

During the detection phase, the correlation between the signal and a template pulse train (TPT) is done on the length of one acquisition sequence. The $N$ first maximums of the result of the correlation are marked for verification in decreasing order.

The TPT is constructed according to the time-hopping sequence. It is equal to 1 where there is a pulse, 0 everywhere else. Each pulse is actually a burst of several pulses.

The algorithm locks on the first maximum that survive the verification. This verification consists of taking $A$ correlation points equally spaced by the length of the acquisition sequence, starting from the maximum to verify. A threshold check is performed on those points and if at least $B$ of them succeed, synchronization is declared successful at that position.

If none of the $N$ maximums pass the verification, then the synchronization fails.

The mechanism described in 1.2.3 was added to this.

### 1.2.2 Energy detection

Due to the design of the transmitter, the exact shape of the received signal is not known and it is anyway distorted by the multipath effect of the wireless channel. That is why a square signal TPT is used for the correlation. In the algorithm as described, the correlation is run between this template and the received signal.

$$cor(t) = \sum_k TPT(k)\, s(t+k)$$

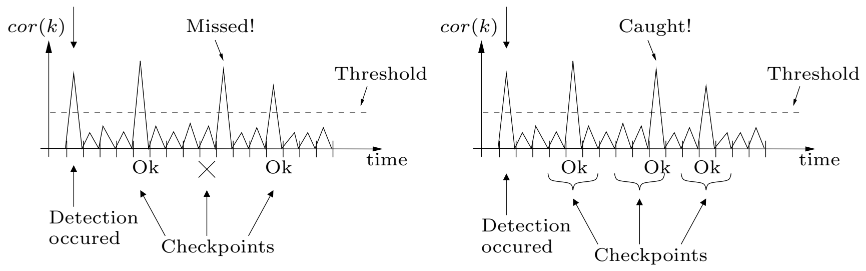
However, since the pulse has some negative parts, I hoped that running the correlation on the energy rather than on the signal itself would give some gain.

$$cor_{energy}(t) = \sum_k TPT(k)\, s^2(t+k)$$

It is not easy to see if the correlation on the energy yields better performance. Since the overall performance of the system depends heavily on many other parameters, such as the thresholds for the elementary correlation and the actual detection, that are far from optimal by now, further studies are necessary to decide which method is appropriate.

### 1.2.3 Search neighborhood

When doing the verification phase, the algorithm check that there are at least B peaks above the verification threshold equally spaced by the length of the sequence and following the first peak detected. The problem was that although the algorithm locked on an actual beginning of the sequence, the following maximum were not found where they were expected. The peak of the following repetition of the sequence was often shifted a bit forward of backward, as illustrated in Fig. 1, and hence was missed during the verification.



(a)

(b)

Figure 1: (a) Illustration of how a peak can be missed during the verification part if it is a bit offset. Here the third peak is missed because it is shifted by one position to the right. (b) In the same situation than before, using neighborhood search the peak is caught.

To mitigate this effect I allowed the algorithm to look for the peak in a defined neighborhood of the expected position. If a peak above the verification threshold is found in this neighborhood, this check is considered as successful. The size of the neighborhood can be given as a parameter.

The offset of the peak to the expected location was actually a consequence that the correct clock signal was not used to generate the pulses at the transmitter. In consequence, the pulses were generated at a lower frequency than expected.

With better quality traces and more exact parameters this mechanism may not be needed as much. However, I think it would be good to let the algorithm search in a small neighborhood around peaks positions. Because, for example as an effect of the multipath, a peak could be offset compared to the others and hence be missed by a too tight verification.

### 1.3 Achievements and future work

- **Achievements** : Two MEX-files written in C (about 300 lines of code each) that efficiently implements the algorithm described in [4] plus the mechanisms described in 1.2. One of them run synchronization on the signal itself, the other one on the energy (as in 1.2.2).

- **Future work** : If a mechanism that allows for systematical verification of the synchronization is implemented in the testbed, then the original goal could be achieved.

  In the implementation of the synchronization on the energy, the signal is squared in the correlation loop. This isn't efficient. The signal should be squared once and for all at the initialization of the algorithm.

## 2 Effect of the channel

### 2.1 Goal

For the timing estimation problem, it is important to get a clear idea of the distortion introduced in the signal by the system and the channel.

This analysis had not been done for the testbed described in [1]. It could for example help to decide if it is worth doing I-Q demodulation or not.

### 2.2 Transmitter, Receiver and Channel model

The transmitter and receiver are approximations of the system described in [3]. The whole chain is illustrated in Fig. 2.

#### 2.2.1 Transmitter

The transmitter is composed of a pulse generator and a passband filter. We assume the pulse generator creates Gaussian pulses of the form :

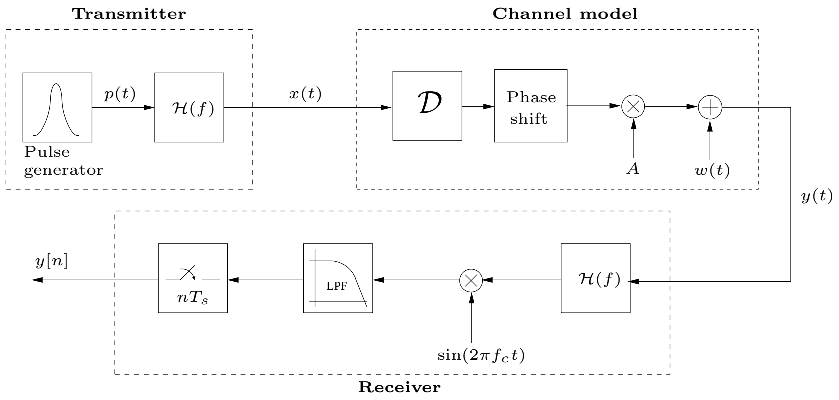


Figure 2: The passband channel block diagram. $A$ models the fading and $w(t)$ is real-valued AWGN with spectral density $\frac{N_0}{2}$. The phase shift is uniformly distributed on $[0;2\pi]$. The delay element introduce a delay $\Delta$.

$$p(t) = e^{-\left(\frac{t}{\tau}\right)}$$

where $\tau$ is the half-pulse duration. In order to up-modulate the pulse to its dedicated frequency band, it is filtered through an ideal passband filter $\mathcal{H}(f)$ with cut-off frequencies at 4 and 4.5 GHz. $\tau$ is chosen small enough so that there is enough energy at 4.25 GHz.

$$\mathcal{H}(f) = \begin{cases} 1 & \text{if } f_c \leq |f| \leq f_c + B, \\ 0 & \text{o.w.} \end{cases}$$

Here, we choose $f_c = 4$ GHz as the carrier frequency and call $B = 500$ MHz the bandwidth of the pulse.

### 2.2.2 Receiver

First, the signal is filtered through the same ideal passband filter than at the transmitter. Then to down-modulate the signal, we multiply it by a pure sinusoid at 4 GHz. Then to remove the double-frequency components introduced by this multiplication, the signal is filtered through an ideal low-pass filter with cut-off frequency at 700 Mhz. Finally, we sample the signal at 2 GHz.

### 2.2.3 Channel model

The channel is approximated by a delay, which is what we want to estimate, a multiplication by a random variable, that characterize the fading, a phase shift and real-valued additive white Gaussian noise (AWGN). The multipath effect is not taken into account. The distribution of the fading is not characterized because we are not interested in studying its effect.

## 2.3 Baseband equivalent model

### 2.3.1 Baseband pulse

The spectrum of the pulse is :

$$P(f) = \sqrt{\pi}\tau e^{-(\pi\tau f)^2}$$

After filtering through the passband filter, we obtain :

$$X(f) = \begin{cases} \sqrt{2\pi}\tau e^{-(\pi\tau f)^2} & \text{if } f_c \leq |f| \leq f_c + B \\ 0 & \text{o.w.} \end{cases}$$

The inverse Fourier transform of this signal can't be done analytically. However, it would be possible to use numerical methods to get an idea of the temporal form of the pulse.

Then, the spectrum of the baseband pulse is :

$$\begin{aligned} X_{bb}(f) &= \begin{cases} \sqrt{2}X(f+f_c) & \text{if } f+f_c > 0 \\ 0 & \text{if } f+f_c \leq 0 \end{cases} \\ &= \begin{cases} \sqrt{2\pi}\tau e^{-(\pi\tau(f+f_c))^2} & \text{if } 0 \leq f \leq B \\ 0 & \text{o.w.} \end{cases} \end{aligned}$$

It is then up-modulated by multiplying by $e^{j2\pi f_c t}$. The relation to the real-valued signal that is sent is :

$$\begin{aligned} x(t) &= \sqrt{2}\Re e\{x_{bb}(t)e^{j2\pi f_c t}\} \\ &= \sqrt{2}\Re e\{x_{bb}(t)\}\cos(2\pi f_c t) - \sqrt{2}\Im m\{x_{bb}(t)\}\sin(2\pi f_c t) \end{aligned}$$

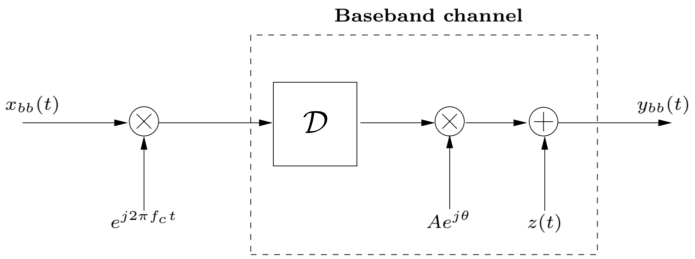


Figure 3: The baseband equivalent channel block diagram. The delay element introduce a delay $\Delta$. $z(t) \sim \mathbb{C}\mathcal{N}(0, N_0)$ is AWGN, $\theta \sim \mathcal{U}[0; 2\pi]$. $A$ is a random variable that models the fading effect of the wireless channel. We are not interested in this effect for our problem.

### 2.3.2 Baseband channel

The baseband equivalent channel is illustrated in Fig. 3. The only difference with the passband channel is that we replace the phase shift element by a multiplication by $e^{j\theta}$ where $\theta$ is uniformly distributed on $[0; 2\pi]$. Moreover the real-valued AWGN is replaced by bandlimited complex AWGN.

### 2.3.3 Received signal

In baseband, the output of the channel is :

$$y_{bb}(t) = x_{bb}(t-\Delta)e^{j2\pi f_c(t-\Delta)}Ae^{j\theta} + z(t)$$

and the real signal received :

$$\begin{aligned}
y(t) &= \sqrt{2}\Re e\{y_{bb}(t)\} \\
&= \sqrt{2}\Re e\left\{x_{bb}(t-\Delta)Ae^{j(2\pi f_c(t-\Delta)+\theta)}\right\} + \sqrt{2}\Re e\{z(t)\} \\
&= \sqrt{2}A\Re e\{x_{bb}(t-\Delta)\}\cos(2\pi f_c(t-\Delta)+\theta) \\
&\quad -\sqrt{2}A\Im m\{x_{bb}(t-\Delta)\}\sin(2\pi f_c(t-\Delta)+\theta) + w(t)
\end{aligned}$$

where $w(t) = \sqrt{2}\Re e\{z(t)\}$ is real-valued bandlimited AWGN with spectral density $\frac{N_0}{2}$. Since we choose $z(t)$ to be bandlimited and that so is the signal itself, the bandpass filter at the receiver has no effect. We are now interested to see the effects of the down-conversion. First, we multiply by the sine :

$$\begin{aligned} y(t)\sin(2\pi f_c t) = & \sqrt{2}A\Re e\{x_{bb}(t-\Delta)\}\cos(2\pi f_c(t-\Delta)+\theta)\sin(2\pi f_c t) \\ & -\sqrt{2}A\Im m\{x_{bb}(t-\Delta)\}\sin(2\pi f_c(t-\Delta)+\theta)\sin(2\pi f_c t) \\ & +w(t)\sin(2\pi f_c t) \end{aligned}$$

Using the trigonometric product to sum formulas, we finally obtain the result of the multiplication by the sinusoid :

$$\begin{aligned} y(t)\sin(2\pi f_c t) = & \tfrac{\sqrt{2}}{2}A\Re e\{x_{bb}(t-\Delta)\}\sin(2\pi f_c\Delta-\theta) \\ & -\tfrac{\sqrt{2}}{2}A\Im m\{x_{bb}(t-\Delta)\}\cos(2\pi f_c\Delta-\theta) \\ & +w(t)\sin(2\pi f_c t) \\ & +\underbrace{\frac{\sqrt{2}}{2}A\Re e\{x_{bb}(t-\Delta)\}\sin(2\pi(2f_c)t-2\pi f_c\Delta+\theta)}_{Double\ frequency\ term} \\ & +\underbrace{\frac{\sqrt{2}}{2}A\Im m\{x_{bb}(t-\Delta)\}\cos(2\pi(2f_c)t-2\pi f_c\Delta+\theta)}_{Double\ frequency\ term} \end{aligned}$$

The double frequency terms disappear after low-pass filtering. The last step is now sampling. The sampling period is $T_s = 0.5$ ns. We'll call $w[n]$ the noise part after low-pass filtering and sampling to simplify the notation, although it is not a sampled version of $w(t)$.

$$\begin{aligned} y[n] &= \tfrac{\sqrt{2}}{2}A\Re e\{x_{bb}(nT_s-\Delta)\}\cos(\tfrac{\pi}{2}-2\pi f_c\Delta+\theta) \\ & \quad -\tfrac{\sqrt{2}}{2}A\Im m\{x_{bb}(nT_s-\Delta)\}\sin(\tfrac{\pi}{2}-2\pi f_c\Delta+\theta)+w[n] \\ &= \tfrac{A}{\sqrt{2}}\Re e\left\{x_{bb}(nT_s-\Delta)e^{j(\frac{\pi}{2}-2\pi f_c\Delta+\theta)}\right\}+w[n] \\ &= \tfrac{A}{\sqrt{2}}\Re e\left\{jx_{bb}(nT_s-\Delta)e^{-j(2\pi f_c\Delta-\theta)}\right\}+w[n] \\ &= \tfrac{-A}{\sqrt{2}}\Im m\left\{x_{bb}(nT_s-\Delta)e^{-j(2\pi f_c\Delta-\theta)}\right\}+w[n] \end{aligned}$$

### 2.4 Future work

- In my model, I do not take into account the multipath effect present in the wireless channel. A more careful study should take this into account.
- I did not investigate the shape of the temporal form of the pulse that is sent (neither passband nor baseband). Although it is probably not

possible to derive it analytically, it could be done numerically. Maybe this could be used to design a better TPT to use for the correlation in the synchronization.

# 3 Synchronization on a 802.15.4a architecture

## 3.1 Goal

The goal of my last task, was to evaluate the performance of the synchronization on two variants of the architecture of the receiver developed by ST Microelectronics for the 802.15.4a standard (see B.1). The algorithm used is described in [7] but a short introduction is given in appendix A.2.

The metric used is the probability of misdetection. The scenarios with multipath and without multipath are investigated.

The evaluation is done through a simulation. The first problem was to find a time-discrete approximation of the receiver in order to be able to simulate it.

Since the performance of the algorithm heavily depends on the choice of the detection threshold for the coarse synchronization, it was necessary to carefully derive it for both schemes. In the second scheme, the quantization threshold is also critical.

## 3.2 Discretization of the system

The system described in B.1 is continuous. Therefore I need to approximate it by a discrete-time system. In order to do so, instead of a continuous signal, I use a signal sampled at very high frequency $f$ (*i.e.* 20 GHz).

Furthermore, I approximate the integration by a sum of rectangle :

$$\int_0^T |s(t)|^2 dt \approx \Delta \sum_{k=1}^{N_T} |s(k\Delta)|^2$$

where $N_T = Tf$ is the number of samples in a period $T$ and $\Delta = \frac{1}{f}$ is the sampling period (see Fig. 4). Since the output of the "integrator" is no more continuous, I replace the sampling of period $T$ by downsampling by a factor $N_T$.

The channel part of the simulation has been written by Manuel Flury. A function that generates discrete bandlimited AWGN according to [2] was provided by Ruben Merz.

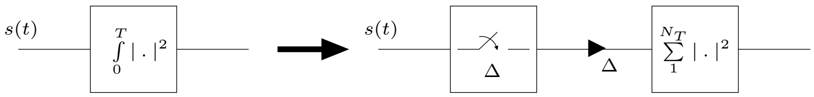


Figure 4: Discrete approximation of the integration by a sum of rectangle. $\Delta$ is the sampling period and $N_T$ is the number of samples in a period $T$.

## 3.3 Thresholds derivation

All the threshold derivations are making the assumption that the noise samples are independent and identically distributed. However, when dealing with sampled bandlimited white Gaussian noise, this assumption is not true unless the signal is sampled at the Nyquist frequency $f_n = 2B$ (with corresponding period $T_n = \frac{1}{f_n}$) where $B$ is the bandwidth of the noise. Therefore, all the derivations are made with the assumption that the signal is sampled at $f_n$.

The threshold we obtain will be anyway independent of the sampling frequency of the input signal. Hence, this is not a restriction.

It should be noted that $\Delta$ is a scaling factor that depends on the sampling frequency $f$ of the simulation. On the other hand $N_T$ changes when we derive the threshold. It is the number of samples in a period $T$ sampled at $f_n$.

### 3.3.1 First scheme

The detection threshold for the coarse synchronization was derived by Manuel Flury. The derivation is given in appendix B.3.

### 3.3.2 Second scheme

**Quantizer threshold** Here we want to derive the optimal threshold in order to quantize to 1 when a pulse goes through the integrator and its amplitude squared is $A^2 = 1$, and to 0 otherwise.

The optimal decision is given by the Maximum A Posteriori (MAP) rule. We assume code symbols to be equiprobable.

$$A = \begin{cases} 1 & \text{w.p. } \frac{1}{3} \\ 0 & \text{w.p. } \frac{1}{3} \\ -1 & \text{w.p. } \frac{1}{3} \end{cases}$$

Those code symbols are used as defined in the 802.15.4a standard to create the acquisition sequence.

We assume also that the sampling point $t_0$ is uniformly distributed on a symbol period $[0; T_s]$.

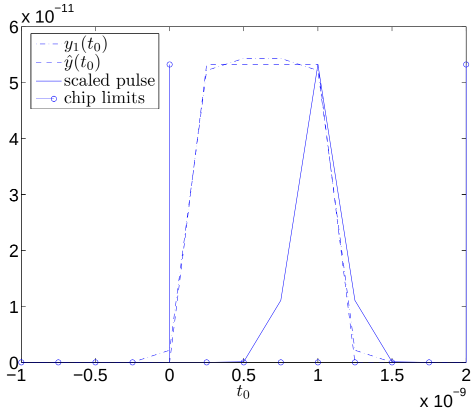


Figure 5: $y_1(t_0)$ (dash-pointed line) is the output of the integrator when there is no noise and a pulse is present in the integration window as a function of the sampling point $t_0$. $\hat{y}(t_0)$ (dashed line) is the approximation of $y_1(t_0)$ by a square function. The plain line is a scaled version of the pulse and the impulses are the chip limits.

We notice that we can write the output of the integrator as a function of the sampling point differently if a pulse is present in the integration window or not :

$$\begin{aligned} y_0(t_0) &= \Delta \sum_{i=0}^{N_T-1} w_i^2 \\ y_1(t_0) &= \Delta \sum_{i=0}^{N_T-1} (x_i(t_0) + w_i)^2 \end{aligned}$$

where :

$$x_i(t_0) = p(t_0 + iT_n)$$

are samples of the pulse and $w_i$ are i.i.d. Gaussian noise samples with variance $\sigma^2$, $y_1(t_0)$ is the output when a pulse is present and $y_0(t_0)$ when there is only noise at the input.

First of all, using Matlab, we look at $y_1(t_0)$ if there is no noise (*i.e.* $w_i = 0\ \forall i$). This is illustrated in Fig. 5. We observe that it is rather flat. Therefore, we approximate it by :

$$y_1(t_0)|_{w_i=0\ \forall i} = \Delta \sum_{i=0}^{N_T-1} x_i^2(t_0) \approx \hat{y}(t_0) = \begin{cases} C & \text{if } t_0 \in [0; 1\,\text{ns}], \\ 0 & \text{o.w.} \end{cases}$$

where $C$ is a constant.

We call $X$ the indicator that $t_0 \in [0; 1\,\text{ns}]$ happens jointly with $A^2 = 1$.

$$p_1 = \Pr(X = 1) = \Pr\{t_0 \in [0 : 1\,\text{ns}]\}\Pr\{A \in \{-1; 1\}\} = \frac{1\,\text{ns}}{T_s} \times \frac{2}{3} = \frac{1}{192}$$

$$p_0 = \Pr(X = 0) = 1 - p_1 = \frac{191}{192}$$

Now we can derive the distribution of $y$ given $X$.

If $X = 0$, the output of the integrator is $y_0(t_0)$. Since $w_i \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma^2)$, then $y_0(t_0)$ follows a $\chi^2$ distribution with $N_T$ degrees of freedom.

Now if $X = 1$, the output of the integrator is $y_1(t_0)$. Since now $w_i \sim \mathcal{N}(x_i(t_0), \sigma^2)$, then $y_1(t_0)$ follows a non-central $\chi^2$ distribution with $N_T$ degrees of freedom and non-centrality parameter $\lambda$ :

$$\lambda = \sum_{i=0}^{N_T-1} \frac{x_i^2(t_0)}{\sigma^2} = \frac{1}{\Delta\sigma^2}\left[\Delta \sum_{i=0}^{N_T-1} x_i^2(t_0)\right] \approx \frac{1}{\Delta\sigma^2}\hat{y}(t_0) = \frac{C}{\Delta\sigma^2}$$

We can now write the MAP rule :

$$H^{MAP}(y) = \arg\max_x\, p_{X|Y}(x|y) = \arg\max_x\, p_{Y|X}(y|x)\Pr(X = x)$$

where $y$ is the output of the integrator. Since we only have a binary hypothesis on $X$, we can rewrite it :

$$\begin{aligned} p_{Y|X}(y|0)p_0 &\underset{1}{\overset{0}{\gtrless}} p_{Y|X}(y|1)p_1 \\ f_{\Delta\sigma^2\chi^2_{N_T}}(y)p_0 &\underset{1}{\overset{0}{\gtrless}} f_{\Delta\sigma^2\chi^2_{N_T,\lambda}}(y)p_1 \\ f_{\chi^2_{N_T}}\left(\frac{y}{\Delta\sigma^2}\right)p_0 &\underset{1}{\overset{0}{\gtrless}} f_{\chi^2_{N_T,\lambda}}\left(\frac{y}{\Delta\sigma^2}\right)p_1 \end{aligned}$$

So, to find the threshold, we need to solve :

$$f_{\chi^2_{N_T,\lambda}}\left(\frac{y}{\Delta\sigma^2}\right) p_1 - f_{\chi^2_{N_T}}\left(\frac{y}{\Delta\sigma^2}\right) p_0 = 0$$

This equation can probably not be solved easily analytically, so we use numerical methods to find the threshold $\varphi$ and the rule is :

$$H^{MAP}(y) = \begin{cases} 1 & \text{if } y \geq \varphi, \\ 0 & \text{o.w.} \end{cases}$$

To simplify the derivation, the effects of the channel (fading, multipath, etc.) were not taken into account. Therefore, its performance will be clearly suboptimal. Especially the fading will have a very big impact on the performance with this threshold.

**Detection threshold** The derivation of the detection threshold for the coarse synchronization with this scheme follows the same line than for the other scheme as presented in B.3.

The correlation function is given by :

$$cor(k) = \sum_{i=0}^{N_{TPT}-1} TPT(i)\, r[k+i] = \sum_{i=0}^{N_{TPT}-1} TPT(i) \sum_{j=k}^{k+N-1} H^{MAP}(y_{i+j})$$

where $N_{TPT}$ is the number of sample in the TPT used for the correlation during the coarse synchronization, $y$ is the output of the integrator and $r$ the output of the receiver. The TPT is a repetition of $G$ times the acquisition sequence.

The number of non-zero symbols in the sequence is $N_{nz}$. The number of samples in a symbol is $T_s/T_{out}$. Then the number of non-zero samples in the TPT is

$$N_1 = N_{nz} \times G \times (T_s/T_{out})$$

Therefore :

$$cor(k) = N \qquad \text{where} \qquad N \sim \text{Bin}(N_1 \times N, p)$$

where :

$$\begin{aligned} p &= \Pr\{H^{MAP}(y) = 1 \mid \text{only noise at the receiver}\} \\ &= \Pr\{y \geq \varphi \mid \text{only noise at the receiver}\} \\ &= 1 - F_{\chi^2_{N_T}}\left(\frac{\varphi}{\Delta\sigma^2}\right) \end{aligned}$$

Then we choose the threshold $\alpha$ such that :

$$\Pr\{cor(k) \geq \alpha \mid \text{only noise at the receiver}\} \leq p_0$$

where we choose $p_0$ as small as we want to.

$$\begin{aligned} \Pr\left\{N \geq \alpha \mid \text{only noise at the receiver}\right\} &\leq p_0 \\ 1 - F_N\left(\alpha\right) &\leq p_0 \end{aligned}$$

where $F_N$ is the cumulant distribution function of $N$. And therefore, the threshold is :

$$\alpha = F_N^{-1}(1 - p_0)$$

## 3.4 Probability of misdetection

The simulation was then run to derive the probability of misdetection as a function of the SNR. 1000 simulations were run for each points.

Four scenarios are investigated : with and without multipath for the two schemes of the receiver.

The probability of misdetection is defined as the probability that the algorithm doesn't lock at all in the presence of the acquisition sequence.

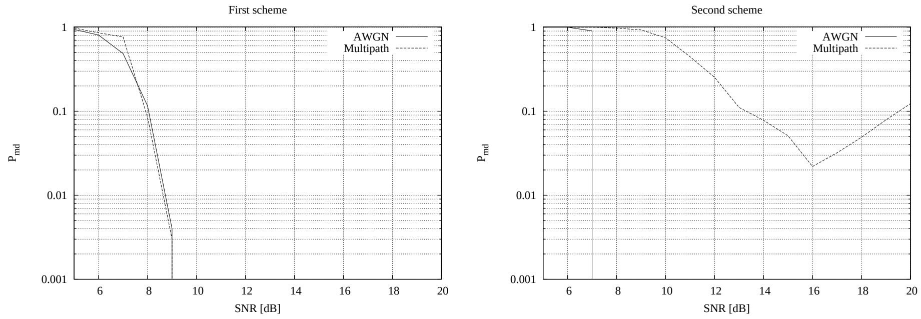
The measure or the quality of the synchronization is the distance $d$ in samples between where the algorithm locked and the closest beginning of an acquisition sequence.

For some transition SNRs, an histogram of the distribution of $d$ is provided in Fig. 7.

## 3.5 Achievements and future work

### Achievements

- An implementation of the simulation in Matlab with some parts written in C for efficiency (about 300 lines of Matlab code and 400 of C).



(a) (b)

Figure 6: The probability of misdetection as a function of the SNR

  - Two variants of the receiver,
  - Coarse and fine synchronization according to [7],
  - A script that computes the threshold of the MAP rule for the quantizer,
  - A script that run the whole simulation.

- Derivation of the critical thresholds for the second scheme.

- Derivation by simulation of the probability of misdetection as a function of the SNR.

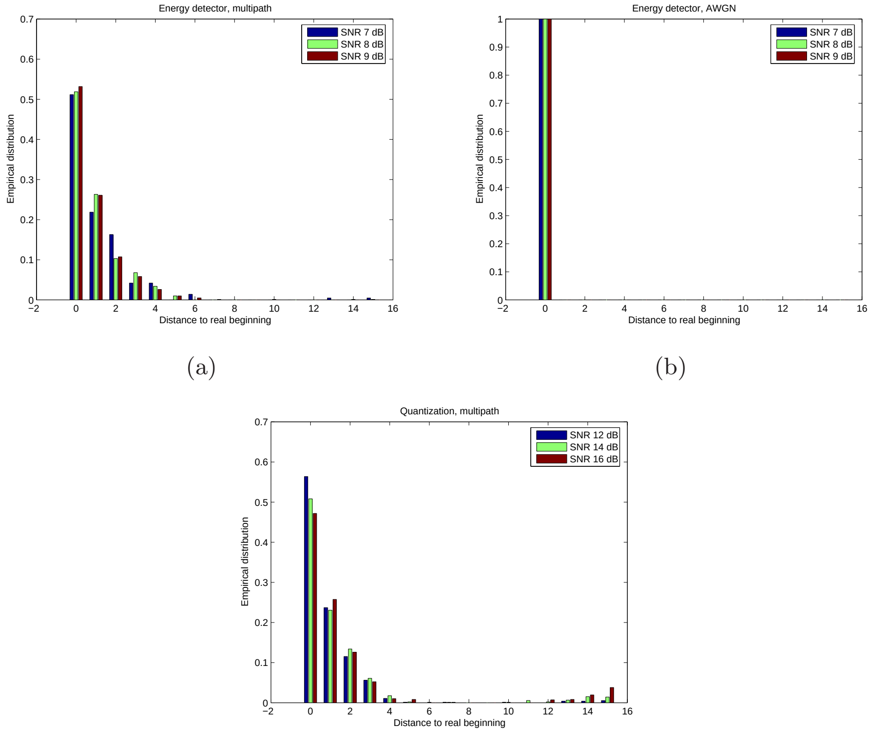
**Future work**

- Derive the quantization threshold without neglecting the multipath and the fading.

- Implement the channel estimation and SFD search parts of the algorithm.

- Derive the probability of misdetection in other scenarios, such as :
  - with interferers
  - with different acquisition sequences
  - etc.

- Derive the probability of false alarm.



(a)

(b)

(c)

Figure 7: The histograms shows the empirical distribution of the distance in samples from the synchronization point to the closest real beginning of a sequence. Distances over 15 samples indicate that the algorithm locked in noise. Those distances are not taken into account in those histograms. (a) First scheme, multipath channel (b) First scheme, AWGN channel (c) Second scheme, multipath channel. For the AWGN channel, the algorithm always lock on the real beginning of a sequence or not at all, thus a plot is not provided.

# 4 Conclusions

## 4.1 Skills acquired

- I learnt how to write C-functions for Matlab (MEX-files).
- I learnt what really means ”experimental”.
- I was able to apply in practice basic principles of digital communications such as :
  - Synchronization,
  - Baseband equivalent,
  - MAP rule,
  - etc.

  And see them work in practice[1].
- I could see the interaction between theory and practice. Especially in the derivation of the thresholds.
- I understand the principles of an UWB physical layer.

## 4.2 Major events

The major event of the project was clearly when we had to abandon the experiments with the traces from the testbed due to their low quality.

Then, although it is a rough approximation, the derivation of the MAP rule for the quantizer wasn't really expected and allowed us to derive a probability of misdetection for the second scheme of the receiver.

## 4.3 Self-assessment

I think I've not been very successful with the first part of the project, the synchronization on the testbed. It has been a bit frustrating to have to leave it behind. However, it wasn't really possible to do more than I did without new traces and a mechanism to automatically check the quality of the synchronization.

For the derivation of the effect of the channel, the model I use is very simple and thus maybe not so useful in itself. But it could be developed further into something that can be used in practice.

[1] Which is quiet amazing by the way ;) ...

On the other hand, although it is a rough approximation, I think my derivation of the MAP rule for the quantizer is a good guideline for a possible derivation of a threshold that takes into account the fading and the multipath.

## Acknowledgements

I would like to thank Ruben Merz for his enthusiastic support and supervision all along the project. Thanks also to Jérôme Vernez for the traces from the testbed and to Manuel Flury for his help during the last part of the project.

# Appendix

## A Algorithms

### A.1 Power Independent Detection

This algorithm aims at providing good performance in the presence of IUI (Inter-User Interference). It is composed of two phases. It is described in details in [4].

**Detection phase** During this phase, a correlation is done between the signal and a TPT. But unlike the classical synchronization scheme, an elementary correlation is run on each individual pulse composing the TPT. Then each elementary correlation goes through an elementary decision block that performs an elementary threshold check $\theta$. Then all the elementary decisions are summed up. This sum goes through a main decision block. If the absolute value of the sum exceeds a main threshold $\varphi$ then a match between the TPT and the signal is declared. This phase is illustrated in Fig. 8.

**Verification phase** When a match is declared, the algorithm goes to the verification phase. During this phase, it runs the detection phase with a second main threshold $\varphi\prime$, larger than $\varphi$, on $A$ points equally spaced by the length of the TPT starting from the first match declared. Then, if at least $B$ matches among $A$ detections are declared, synchronization is declared.

### A.2 802.15.4a synchronization from ST

The algorithm for the synchronization on this architecture is describe in details in [7].

It is composed of four steps :

1. Coarse synchronization,
2. Fine synchronization,
3. Channel estimation,
4. SFD search.

However, I only describe here the first two since I only implemented those ones.

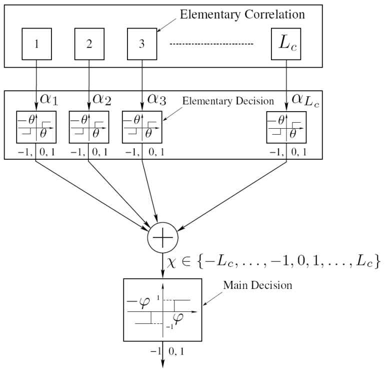


Figure 8: PID method. Illustration of the detection phase. Each pulse is detected based on an elementary decision block. The final decision is based on the number of pulses detected. The sign of the decision depends on the sign of the input of the decision block. Illustration taken from [4].

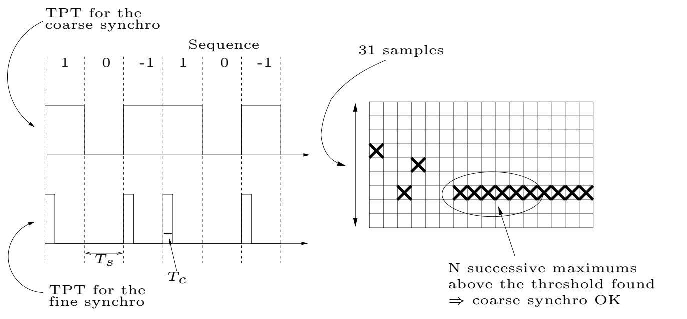
**Coarse synchronization** We construct a coarse TPT. For each symbol of the sequence, we repeat over the whole symbol period the squared value of the ternary code symbol. Then the correlation is run between the signal and this TPT. The correlation is then sampled at the symbol rate. It is then organized into a matrix with 31 samples (corresponding to the 31 symbols in the sequence) per column. The maximum of each column is searched. If it exceeds the detection threshold, it is marked. If at least $N$ successive maximums are marked at the same position, then the coarse synchronization succeeds at points $T_{coarse}$. The matrix is illustrated in Fig. 9.

**Fine synchronization** Now we go back in fine granularity. We have to create a new TPT. This time, the squared value of the symbol is not repeated over the symbol period, but only on the first chip period. Then the correlation is run between this TPT and a small part of the signal :

$$[T_{coarse} - T_s; T_{coarse} + T_s]$$

where $T_s$ is a symbol period. The maximum of the correlation is chosen as the synchronization point.

The difference between the TPT used for coarse and fine synchronization is illustrated in Fig. 9.



(a) (b)

Figure 9: (a) The TPTs used for respectively coarse and fine synchronization. (b) The matrix used for the coarse synchronization.

# B 802.15.4a architecture from ST

## B.1 Architecture of the receiver

The receiver is roughly speaking an energy detector. We investigate two scheme. In the first one, the squared magnitude of the signal is integrated over a period $T = 8$ ns and then sampled at this same period $T$.

In the second scheme, we have the same integration followed by sampling, but with $T = 1$ ns. The integrator is then followed by a 1-bit quantizer with threshold $\varphi$ that is derived in 3.3. This quantizer is followed by a summator on $N = 8$ samples. After summation, the signal is downsampled by a factor $N$. The quantization is hoped to reduce the impact of strong interferers.

In both architecture, the output sampling frequency is $T_{out} = 125$ MHz. Those two architectures are illustrated in Fig. 10.

## B.2 Preamble construction

The preamble is composed of 31 symbols from a ternary code $\{-1; 0; 1\}$. Each symbol has a length of $T_s = 128$ ns. A symbol is split into 64 chips of length $T_c = 2$ ns. A pulse is transmitted in the first chip with its maximum at the center of the chip. Pulses are assumed to be Gaussian of the form :

$$p(t) = Ae^{-2\pi\left(\frac{t}{\tau}\right)^2}$$

where $A$ is the amplitude of the pulse and $\tau$ is the half-pulse duration. We use pulse amplitude modulation, then $A \in \{-1; 0; 1\}$.

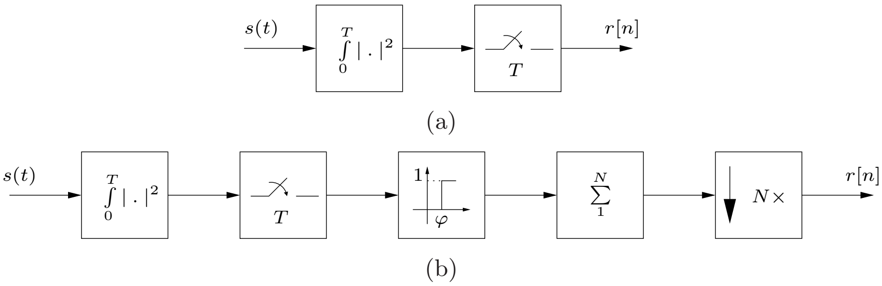


Figure 10: The two investigated variants of the receiver. (a) First scheme : $T = 8$ ns (b) Second scheme : $T = 1$ ns, $N = 8$, $\varphi$ is the quantization threshold.

## B.3 Detection threshold for the first scheme

In this scheme, we need a detection threshold for the coarse synchronization. The correlation is given by :

$$cor(k) = \sum_{i=0}^{N_{TPT}-1} TPT(i)\, r[k+i] = \sum_{i=0}^{N_{TPT}-1} TPT(i)\, \Delta \sum_{j=k}^{k+N_T-1} s(i + j\Delta)$$

where $N_{TPT}$ is the number of samples in the TPT used for the correlation during the coarse synchronization. On top of that, the TPT is a repetition of $G$ times the sequence. Now, we want to find the detection threshold $\beta$ such that :

$$\Pr\{cor(k) \geq \beta \mid \text{only noise at the receiver}\} \leq p_0$$

If there is only noise at the receiver, then $s(k\Delta) = w_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2)^{\,2}$. The number of non-zero symbols in the sequence is $N_{nz}$. The number of samples in a symbol is $T_s/T_{out}$. Then the number of non-zero samples in the TPT is :

$$N_1 = N_{nz} \times G \times (T_s/T_{out})$$

Therefore :

$$cor(k) \sim \Delta\sigma^2 \chi^2_{N_1 \times N_T}$$

[2] For a remark about this assumption, see 3.3

and, if $N \sim \chi^2_{N_1 \times N_T}$, then :

$$\begin{aligned} \Pr\{cor(k) \geq \beta \,|\, \text{only noise at the receiver}\} &\leq p_0 \\ \Pr\left\{N \geq \frac{\beta}{\Delta\sigma^2} \,|\, \text{only noise at the receiver}\right\} &\leq p_0 \\ 1 - F_N(\frac{\beta}{\Delta\sigma^2}) &\leq p_0 \end{aligned}$$

where $F_N$ is the cumulant distribution function of $N$. Then the threshold is :

$$\beta = \Delta\sigma^2 F_N^{-1}(1 - p_0)$$

# References

[1] J. Colli-Vignarelli, J. Vernez, C. Dehollain, R. Merz, G. Quintero, A. Skrivervik and J.-Y. Le Boudec. An Impulse Radio Ultra-Wide Band Testbed with Interference. MICS 2006 5th Scientific Conference and Panel Review in Zurich, October 2006. poster.

[2] N. C. Beaulieu and C. C. Tan. An FFT method for generating bandlimited Gaussian noise variates. In *Global Telecommunications Conference, 1997. GLOBECOM '97., IEEE*, volume 2, pages 684–688, 1997.

[3] J. Colli-Vignarelli. Émetteur et Récepteur Ultra Wide Band. Summer internship report, under supervision of R. Merz (LCA) and C. Dehollain (LEG), EPFL, 2006.

[4] Alaeddine El Fawal and Jean-Yves Le Boudec. A Robust Signal Detection Method for Ultra Wide Band (UWB) Networks with Uncontrolled Interference. *IEEE Transactions on Microwave Theory and Techniques (MTT)*, 54(4, part 2):1769–1781, 2006.

[5] H. Hashemi. The Indoor Radio Propagation Channel. *Proceedings of the IEEE*, 81(7):943–968, July 1993.

[6] J. G. Proakis and M. Salehi. *Communication Systems Engineering*. Prentice Hall, $2^{nd}$ edition, 2002.

[7] M. Sambuq. Synchronisation appliquée à l'architecture XNOR. Internal report from LETI, CEA, France, January 2007.

[8] D. Tse and P. Wiswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.