Resource and Performance Comparisons for Different Acquisition Methods that can be Applied to a VHDL-based GPS Receiver in Standalone and Assisted Cases

Jérôme Leclère, Cyril Botteron, Pierre-André Farine
École Polytechnique Fédérale de Lausanne, Institute of Microengineering (IMT), Electronics and Signal Processing Laboratory, Breguet 2, 2000 Neuchâtel, Switzerland
firstname.lastname@epfl.ch

Abstract — **There are mainly three well-known techniques to acquire a GNSS signal. The first, called Serial Search Acquisition, consist in correlating the Pseudo-Random Noise (PRN) codes for all the codes phases and all the frequency bins. The two others make use of the Fourier Transform in order to parallelize the search, either in the frequency space (Parallel Frequency Space Search Acquisition) or in the code space (Parallel Code Space Search Acquisition). The last two methods make parallelization inherently, but it is also possible to make parallelization with the first method by using multiple correlators testing several code phases at the same time. This paper provides an assessment of the resources required for each method, in terms of circuit, memory and processing power for the case of a hardware based receiver implemented into a FPGA. The associated performances are also provided and compared. The study is based on GPS L1 signal, but the methodology can easily be applied to others GNSS signals.**

Acquisition, Assistance, FFT, FPGA, High Sensitivity, VHDL

I. INTRODUCTION

The first GPS receivers would use an integration time of 1 ms to process the satellite signals, and it would take them quite some time to browse all the search space, roughly 40 seconds (1 ms × 2046 code bins × 20 frequency bins). Today's receivers try to increase the sensitivity to be able to receive very weak signals (due to obstruction or reflection, for indoor application, or to increase the SNR) using longer integration times. In this context, the SIGNATURE project [1] aims to prototype a GNSS solution for Road Applications, using assistance from the EGNOS Data Access Service (EDAS). Within this project, our goal is to develop a hardware receiver with a sensitivity of at least of -145 dBm in acquisition and a rapid time to first fix (TTFF) of 3 to 4 seconds.

To obtain a sensitivity of -145 dBm, the necessary integration times for different sampling frequencies and coherent integration times are given in Table I (the methodology used to obtain theses values is the one presented by Frank van Diggelen in [2]).

The research leading to these results has received funding from the European Community Framework Programme (FP7/2007-2013) under grant agreement n°[228237]

TABLE I. TOTAL INTEGRATION TIME FOR ACQUISITION SENSITIVITY OF -145 DBM (MS)

	Sampling Frequency (MHz)			
Coherent Integration Time (ms)	***2.048***	***4.096***	***8.192***	***16.384***
1	788	419	283	264
2	454	216	158	148
4	220	124	88	84
5	170	100	80	70
10	110	70	50	50

The time to browse all the search space is proportional to these values, which leads to dozens of minutes. Consequently, to be able to process a weak signal, a high parallelization is needed to speed up the acquisition.

The hardware implementation is a first step towards this. An idea is to process the data coming from the front end at the speed of the FPGA, which is far higher than the sampling rate, by buffering the data for the time needed, as shown in Fig. 1. This will speed up the acquisition proportionally to the ratio between the FPGA rate and the sampling rate, which can be from ten to one hundred.

FPGAs contain also different types of elements, logic elements (LE), memory block, DPS block. By balancing the usage between them, we can maximize the parallelization.

For the project and this study, we used the Terasic Altera DE3 System which relies on a high density Stratix III FPGA (EP3S260) and the Rakon GRM8652 high performance front end.

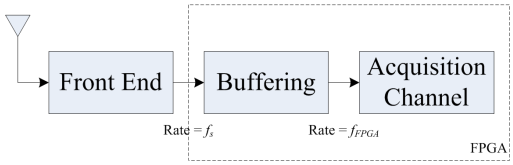

Figure 1. Data Buffering in Hardware Receiver for Acquisition

This paper first investigates the search space considering different cases depending on the availability or not of assistance. The assistance consists in transmitting information, like ephemeris and some corrections, by another transmission channel, like GPRS or RDS. This results in two advantages. The first is that it removes the necessity to decode the data from the signal in space and thus avoid the necessity to wait at least 30 seconds before getting all the data necessary to compute a position. The second concerns the acquisition because we can obtain information about the current satellites that are in view and their Doppler. Another parameter considered is the availability of information about the receiver oscillator, i.e. if we already know the offset leaving just a relative drift unknown, or if we do not know anything. A last parameter considered is the way the replica code is generated.

Different acquisition methods and their implementations into a FPGA will be presented and some subtleties about their implementations discussed.

Finally, a practical case will be presented, with a comparison of the methods for the standalone and assisted case in the context of our project and the choice we made.

II. Doppler Effect on Search Space and Sensitivity

The signals emitted by the satellites are at a specific frequency (1575.42 MHz for L1 carrier and 1.023 MHz for the C/A code). But at the receiver level these frequencies have changed due to three elements:

- Satellite motion, creating a Doppler effect.
- User motion, creating a Doppler effect.
- Receiver oscillator drift, creating a Doppler-like effect.

The effect on the carrier frequency will result in an increase of the search space, whereas the effect on code frequency can limit the maximum integration time, as it will be further detailed.

A. Assumptions

Concerning the assistance, we have to use a reference position to estimate the Doppler of the satellites. Of course, the user is not at the same point creating an error in the estimation, typically 1 Hz/km [2]. Here, the maximum position error considered will be 50 km.

The assumed maximum speed for a user is 130 km/h.

Concerning the oscillator, a deviation from the theoretical value can be influenced by the following three parameters:

- Offset (shift from the nominal value, typ. 2 ppm).
- Drift with temperature (typ. 0.35 ppm/40°C).
- Drift with time (typ. 1 ppm/year).

On the first run of a receiver, these values are not known (we call this case the absolute one), and the total value used is ± 3.35 ppm. But after a first acquisition, the offset of the oscillator can be obtained from the PVT, leaving only a relative drift at the next start up (we call this case the relative one),

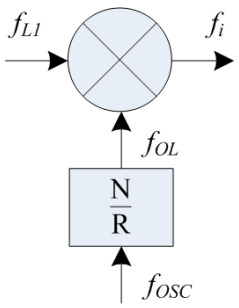

Figure 2. Example of a single stage frequency down-conversion

which is here ± 0.28 ppm (assuming a maximum change in temperature of ± 30°C and maximum one week between two start up).

B. Effect on Carrier Frequency

The Doppler effect results in a shift from the nominal frequency which can be expressed using (1). The Doppler due to the satellite motion and the user motion can be both obtained from this equation.

$$f_d = \frac{f\, v_d}{c} \tag{1}$$

where f is the frequency of the signal, v_d the relative velocity between the satellite and the user, and c the speed of light.

Regarding the receiver oscillator, the problem is different and comes from the frequency down-conversion in the front end. An example with one stage is shown in Fig. 2. The value of the intermediate frequency is obtained using (2). But the oscillator having a deviation (noted α and expressed in ppm), this changes (2) into (3).

$$\begin{aligned} f_I &= f_{L_1} - f_{OL} \\ &= f_{L_1} - \frac{N}{R} f_{OSC} \end{aligned} \tag{2}$$

$$\begin{aligned} f_{I_{real}} &= f_{L_1} - \frac{N}{R} f_{OSC}(1+\alpha) \\ &= f_I - \alpha \frac{N}{R} f_{OSC} \end{aligned} \tag{3}$$

The real intermediate frequency will be shifted from the nominal value, which implies a Doppler-like effect, at the difference that this shift is common to all the channels.

The different causes of the carrier frequency shift are summarized in Table II (see [3] for the details on the satellite motion). If there is assistance, the shift due to the satellite motion will be compensated with still an error due to the reference position [2]. The Doppler due to the user is maximum when the motion is directly towards the satellite and is equal to the user speed.

The maximum carrier frequency shift, which corresponds to the frequency search space, is given in Table III. Four cases

TABLE II. MAXIMUM OF THE COMPONENTS OF THE CARRIER FREQUENCY SHIFT

Component of Carrier Frequency Shift	Value (Hz)
Satellite motion (standalone case)	± 4880
Satellite motion (assisted case)	± 50
User motion (130 km/h)	± 190
Oscillator deviation (absolute case, ± 3.5 ppm)	± 5261
Oscillator deviation (relative case, ± 0.28 ppm)	± 440

TABLE III. MAXIMUM CARRIER FREQUENCY SHIFT

Context	Value (Hz)
No assistance, no oscillator information (absolute case)	± 10331
No assistance, oscillator information (relative case)	± 5510
Assistance, no oscillator information (absolute case)	± 5500
Assistance, oscillator information (relative case)	± 680

have been studied, namely availability of assistance or not and availability of oscillator information (relative case) or not (absolute case). The four cases are given here to be exhaustive, but for the evaluation only the relative case for the oscillator deviation will be considered.

C. *Effect on Code Frequency*

The Doppler due to the satellite motion and the user motion are also obtained from (1), where the frequency is now the one of the code.

The oscillator deviation affects also the sampling frequency and translates this into a Doppler on the code. The real code frequency (after digitalization) can be expressed using (4), and the shift created using (5).

$$f_{C/A_{real}} = f_{C/A}\frac{f_s}{f_{s_{real}}} \tag{4}$$

$$\begin{aligned} f_{dco} &= f_{C/A_{real}} - f_{C/A} \\ &= f_{C/A}\left(\frac{f_s}{f_{s_{real}}} - 1\right) \\ &= -f_{C/A}\frac{\alpha}{1+\alpha} \end{aligned} \tag{5}$$

The maximum of the different components of the code frequency shifts are given in Table IV. The assumptions are the same as used for Table II.

The maximum code frequency shift is given in Table V. Three cases are distinguished, that are different of those for the carrier frequency. The first case is if the code replica generated during acquisition has a fixed frequency (an example is shown in section III.C). In the two others, the frequency of the code replica can be modified, and then it depends on the availability of oscillator information. Note that assistance does not play a role here because the shift due to the satellite motion is compensated, since the frequency of the code replica changes with the frequency bins tested.

TABLE IV. MAXIMUM OF THE COMPONENTS OF THE CODE FREQUENCY SHIFT

Component of Code Frequency Shift	Value (Hz)
Satellite motion	± 3.169
Frequency mismatch	± 0.016
User motion	± 0.123
Oscillator deviation (absolute case)	± 3.427
Oscillator deviation (relative case)	± 0.287

TABLE V. MAXIMUM CODE FREQUENCY SHIFT

Context	Value (Hz)
No frequency compensation	± 6.717
Frequency compensation, No oscillator information (absolute case)	± 3.564
Frequency compensation, Oscillator information (relative case)	± 0.426

If not compensated, the total equivalent Doppler will result in a shift (inversely proportional to the Doppler frequency) between the received signal and the replica that cannot be negligible for high sensitivity as it will be shown in section III.C.

The effect due to the inaccuracy of the sampling frequency exists also for the carrier frequency but it is far less influent, this is why it has not been considered.

III. ACQUISITION METHODS AND IMPLEMENTATION

A. *Conventional Acquisition*

The schematic of the classical process during acquisition, i.e. carrier removal, code removal, coherent integration, modulus operation and non-coherent integration, is given in Fig. 3. The search space is browsed by testing several carrier frequencies and several code phases (2D search).

B. *Correlators Duplication (CD)*

The first idea to test more bins in the same time is simply to duplicate the correlators, as shown in Fig. 4. The gain in speed is proportional to the number of correlators implemented.

The resulting implementation in FPGA is given in Fig. 5. For the coherent accumulator, the hardware cannot be shared. However, after the rate is reduced, it is possible to multiplex the data and share the modulus block and the non coherent accumulator. Moreover, the non coherent accumulator can be implemented with a memory block instead of logic elements.

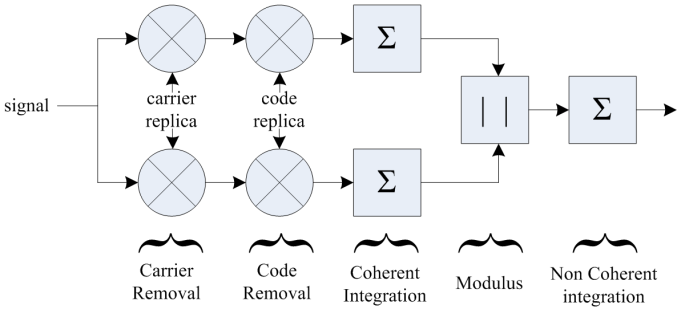

Figure 3. Conventional Acquisition

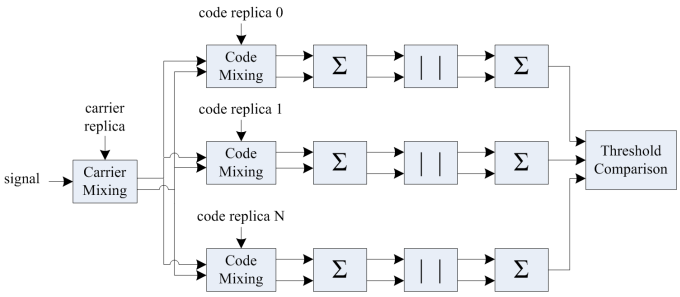

Figure 4. Correlators Duplication Principle

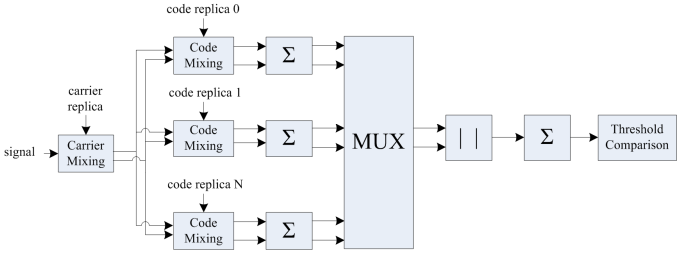

Figure 5. Correlators Duplication Implementation

C. *Parallel Frequency Search (PFS)*

The second idea is to test all the frequency bins at the same time using the FFT as spectrum analyzer, as shown in Fig. 6. More details about this method can be found in [4]. Theoretically, this corresponds to have as much as correlators as there are frequency bins, i.e. roughly from 10 to 200 according to the context.

As this FFT uses a small number of points, it is possible to duplicate it, i.e., it is possible to test several code phases at the same time, but not as many as for the previous method. Such an implementation is presented in Fig. 7. So in fact this implementation mixes the original PFS and CD methods.

Contrary to the CD method, multiplexing is not possible because the output of the FFT is a vector and not a scalar (each point corresponds to a frequency bin).

D. *Parallel Code Search (PCS)*

The third idea is to test all the code bins in the same time using the FFT as correlation accelerator, as shown in Fig. 8. More details about this method can be found in [5] and [6]. Theoretically, this corresponds to have as many correlators as there are samples in 1 ms, i.e. more than 2000. The coherent and non coherent accumulators are implemented with memory blocks. The FFT itself can be implemented in different ways. One is the streaming mode, i.e. that the transform is made on the fly at the clock rate, but it consumes a lot of resources. A second consists in buffering the transformation, which implies a reduced throughput but the resources required are also reduced. Both of them have been evaluated.

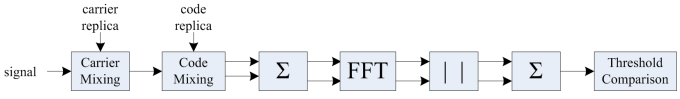

Figure 6. Parallel Frequency Search Principle

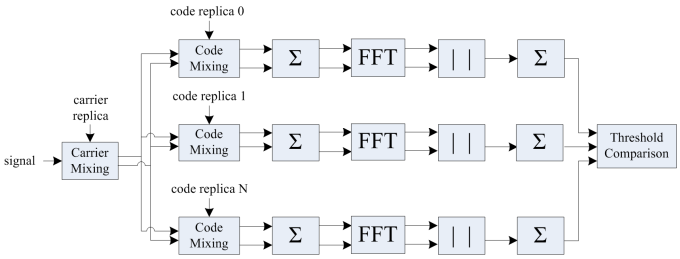

Figure 7. Parallel Frequency Search Implementation

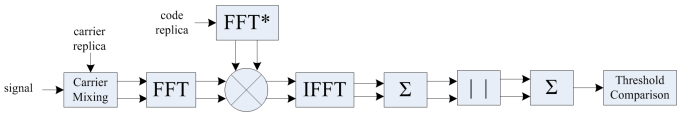

Figure 8. Parallel Code Search Principle & Implementation

IV. SUBTLETIES ON IMPLEMENTATIONS

A. *Data Bit Transition Management*

As we use long integration time, the data bit transition has to be taken into account. There are three known methods to overcome it. The first is to not manage it and take it as a loss, which is not a good idea to achieve high sensitivity. The second is to split the flow of sample into two sets in order to be sure that at least one is free of data transitions, but this results in the processing of twice the length of the original set. And the third is to use a coherent integration time of N ms and test the N possibilities to add them coherently, but this would require N coherent and non coherent accumulators instead of 1, which is not a good idea for hardware implementation. Consequently, only the second method is considered.

B. *Sampling Frequency with Parallel Code Search*

The problem of the FFT is the requirement to have a number of points which is a power of two. Consequently, to respect this rule, the sampling frequency has to be 2.048 MHz, 4.096 MHz, 8.192 MHz, etc.

If this criterion is not satisfied, it is necessary to use zero padding, but that will result in a loss. An example through Tables VI to VIII demonstrates it. Table VI shows the correlation of orthogonal codes, and Tables VII and VIII the correlation with orthogonal codes padded with 0s. In Table VII, by hazard the zero is placed at the good moment on the signal and does not interfere in the correlation, but in Table VIII the zero is placed in the middle of the code which results in half of the maximum correlation. Padding with the code instead of zero would produce the same effect.

As described in [5], to be sure to obtain at least the level of 1 ms correlation, it is needed to use 2 ms of signal, but this results in doubling the memory requirement of the FFT. Consequently, it will be considered for the following that the criterion is satisfied.

TABLE VI. CORRELATION OF ORTHOGONAL CODES

Signal	A	B	C	D	E	Correlation
Replica 0	A	B	C	D	E	5
Replica 1	E	A	B	C	D	0
Replica 2	D	E	A	B	C	0
Replica 3	C	D	E	A	B	0

TABLE VII. CORRELATION OF ORTHOGONAL CODES WELL ZERO PADDED

Signal	A	B	C	D	E	0	Correlation
Replica 0	A	B	C	D	E	0	5
Replica 1	0	A	B	C	D	E	0
Replica 2	E	0	A	B	C	D	0
Replica 3	D	E	0	A	B	C	0
Replica 4	C	D	E	0	A	B	0

TABLE VIII. CORRELATION OF ORTHOGONAL CODES BADLY ZERO PADDED

Signal	D	E	A	B	C	0	Correlation
Replica 0	A	B	C	D	E	0	0
Replica 1	0	A	B	C	D	E	0
Replica 2	E	0	A	B	C	D	3
Replica 3	D	E	0	A	B	C	2
Replica 4	C	D	E	0	A	B	0

C. Code Replica Generation with Parallel Code Search

Usually in the papers presenting this method, the FFT of the code is pre-computed and stored, as shown in Fig. 9. This allows reducing the resources needed but a problem can rise with long integration times. By pre-generating the FFT, it is not possible to modify the frequency of the code replica. According to Table III, the shift on the code frequency can reach 6.717 Hz, which implies a shift of one chip in 148 ms, or half a chip in 74 ms, and this will result in a decrease of the sensitivity. This is why in this paper we present this method by generating the code replica and computing its FFT continuously, to avoid any problem with long integration times.

V. PRACTICAL CASE

A. Methodology

Firstly, according to the context (assistance or not, oscillator characteristics), we compute the size of the frequency search space.

Secondly, according to the sensitivity required, we determine the total integration time. Different coherent integration times are available, for the evaluation given in the paper we used 10 ms.

Thirdly, following the architectures given in Fig. 5, 7 and 8, and the hardware, the resources to implement an acquisition channel are evaluated.

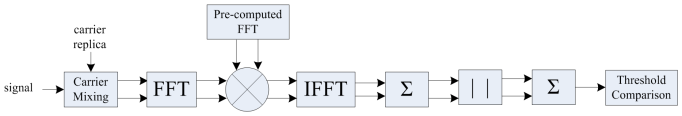

Figure 9. Other Parallel Code Search Implementation

TABLE IX. NUMBER OF CODE CORRELATORS IMPLEMENTABLE PER CHANNEL FOR CD METHOD

	Sampling Frequency (MHz)			
Number of channel implemented	***2.048***	***4.096***	***8.192***	***16.384***
1	1950	1860	1775	1700
2	965	920	880	845
3	640	610	580	555
4	475	455	435	415
5	375	360	345	330
6	310	295	285	270
7	265	250	240	230
8	230	220	210	200
9	200	195	185	175
10	180	170	165	155
11	165	155	150	140
12	150	140	135	130

Finally, we assess the performance in terms of acquisition time. To have another point of view, we also express the performance in terms of number of equivalent correlators.

B. Resources Evaluation

For the CD method, the critical point is the coherent accumulator and the multiplexer in terms of logic elements, because there will be as many accumulators as correlators (which is in the order of hundreds or thousands), and the size of the multiplexer is proportional to it. The sampling frequency plays a role in the resolution of the accumulators, e.g., summations of up to 2048 times 5 bits values results in a value on 16 bits, for 4096 additions it will be 17 bits, etc.

The number of correlators for one channel will influence the number of channels implementable into the FGPA. The maximum number of correlators by channel for one to twelve channels is given in Table IX.

TABLE X. NUMBER OF CODE CORRELATORS IMPLEMENTABLE FOR PFS METHOD

	Sampling Frequency (MHz)			
Number of channel Implemented	***2.048***	***4.096***	***8.192***	***16.384***
1	83	83	83	82
2	41	41	41	41
3	27	27	27	27
4	20	20	20	20
5	16	16	16	16
6	13	13	13	13
7	11	11	11	11
8	10	10	10	10
9	9	9	8	8
10	8	8	8	8
11	7	7	7	7
12	6	6	6	6

TABLE XI. MAXIMUM NUMBER OF CHANNEL IMPLEMENTABLE FOR PCS METHOD

	Sampling Frequency (MHz)			
FFT Implementation	***2.048***	***4.096***	***8.192***	***16.384***
Streaming	6	3	1	0
Buffered	12	6	3	1

For the PFS method, the critical point is the FFT in terms of logic elements, and the coherent accumulator into a lesser extent. As before the sampling frequency only plays a role for the resolution of the coherent accumulator (and the resolution of the FFT behind).

The maximum number of correlators implementable by channel for one to twelve channels is given in Table X.

For the PCS method, the critical point is the FFT and the accumulators in terms of memory. Here the resources required are so huge that the number of channels implementable is limited. The maximum number of channels implementable is given in Table XI.

C. Performance Evaluation

The time to process 1 ms of data depends on the ratio between the sampling frequency and the FPGA frequency and is expressed by (6)

$$T_{1ms} = 10^{-3} \frac{f_s}{f_{FPGA}} \tag{6}$$

The time to process N ms of data is therefore defined by (7)

$$\begin{aligned} T_{Nms} &= N\, T_{1ms} \\ &= N\, 10^{-3} \frac{f_s}{f_{FPGA}} \end{aligned} \tag{7}$$

The acquisition time (i.e. time to browse all the search space) is given by (8) for the Correlator Duplication method, by (9) for the Parallel Frequency Search, and by (10) for the Parallel Code Search. Note that only the processing time is considered, the time needed to store the data into the buffer is not included.

$$T_{A_{CP}} = T_{Nms}\ N_{fb} \frac{N_{cb}}{N_{corr}} \tag{8}$$

$$T_{A_{PFS}} = T_{Nms} \frac{N_{cb}}{N_{corr}} \tag{9}$$

$$T_{A_{PCS}} = T_{Nms}\ N_{fb} \tag{10}$$

where N_{fb} is the number of frequency bin, N_{cb} the number of code bin, and N_{corr} the number of code correlator implemented.

Table XII summarizes the number of acquisition channels implementable for each method (for the first two the maximum number of correlators is used). The acquisition time for one channel is given in Table XIII for the standalone case and in Table XIV for the assisted case.

TABLE XII. NUMBER OF ACQUISITION CHANNEL IMPLEMENTABLE

	Sampling Frequency (MHz)			
Method	***2.048***	***4.096***	***8.192***	***16.384***
CD	1	1	1	1
PFS	1	1	1	1
PCS (streaming)	6	3	1	0
PCS (buffered)	12	6	3	1

TABLE XIII. ACQUISITION TIME FOR 1 CHANNEL IN STANDALONE CASE (MS)

	Sampling Frequency (MHz)			
Method	***2.048***	***4.096***	***8.192***	***16.384***
CD	463.8	1180.7	3373.4	13493
PFS	51.7	131.6	376.0	1522.3
PCS (streaming)	442.1	562.7	803.8	-
PCS (buffered)	1184.0	1492.0	2223.0	4433.5

TABLE XIV. ACQUISITION TIME FOR 1 CHANNEL IN ASSISTED CASE (MS)

	Sampling Frequency (MHz)			
Method	***2.048***	***4.096***	***8.192***	***16.384***
CD	63.8	162.2	463.6	1854.6
PFS	51.7	131.6	376.0	1522.3
PCS (streaming)	60.8	77.3	110.5	-
PCS (buffered)	162.7	205.1	305.6	609.3

Tables XV and XVI give the acquisition times for 12 satellites. As it is shown, the Parallel Frequency Search (containing also a part of Correlator Duplication) is very effective when there is no assistance because the frequency search space is very large. But when the assistance is available, the Parallel Code Search is the most effective method. Use a coherent integration times different than 10 ms would give different values, but the conclusions would be the same.

Table XVII presents the results in another way, stating the number of equivalent correlators implemented. Here by number of equivalent correlators, we mean the number of cells (i.e. one code bin and one frequency bin) processed compared to a classical one correlator receiver. This value depends on the number of correlators really implemented and on the ratio between the FPGA frequency and the sampling frequency. For the PFS method, this value depends also on the search space, since the larger it is the more bins are tested at the same time.

In our case, as we can rely on assistance, we have chosen the Parallel Code Search method, with a sampling frequency of 4.096 MHz giving better precision than 2.048 MHz and keeping a small acquisition time.

TABLE XV. ACQUISITION TIME FOR 12 SATELLITES IN STANDALONE CASE (MS)

	Sampling Frequency (MHz)			
Method	***2.048***	***4.096***	***8.192***	***16.384***
CD	5566.1	14168	40480	161921
PFS	620.4	1579.2	4511.9	18268
PCS (streaming)	884.2	2250.7	9645.7	-
PCS (buffered)	1093.0	2983.9	8892.1	53202

TABLE XVI. ACQUISITION TIME FOR 12 SATELLITES IN ASSISTED CASE (MS)

	Sampling Frequency (MHz)			
Method	***2.048***	***4.096***	***8.192***	***16.384***
CD	765.0	1947.3	5564.7	22254
PFS	620.4	1579.2	4511.9	18268
PCS (streaming)	121.5	309.3	1325.7	-
PCS (buffered)	150.2	410.1	1222.1	7312.1

TABLE XVII. NUMBER OF EQUIVALENT CORRELATORS

	Sampling Frequency (MHz)			
Method	***2.048***	***4.096***	***8.192***	***16.384***
CD	204'960	97'755	46'699	22'352
PFS (standalone)	1'838'865	919'433	459'716	227'089
PFS (assisted)	252'735	126'368	63'184	31'211
PCS (streaming)	1'290'240	645'120	215'040	0
PCS (buffered)	1'043'700	486'570	233'258	77'963

VI. CONCLUSIONS

In this paper, we have discussed the basics of acquisition, with the Doppler effect and the receiver oscillator deviation impacts.

Then we presented the acquisition methods, their implementations into a FPGA, and their little pitfalls.

Finally we have defined a methodology for their assessments, and applied it for the standalone case for which we found that the Parallel Frequency Search (using also duplication) is the most effective architecture, and for the assisted case for which the best performing architecture is the Parallel Code Search.

As shown, there are lots of parameters that influence the evaluation, e.g., the availability of assistance or not (impacting the frequency search space), the oscillator specifications, the sensitivity expected (impacting the integration time), the sampling frequency, the targeted circuit. But the rigorous methodology we presented allows managing them, and our methodology can also be easily reused for other GNSS signals.

ACKNOWLEDGMENT

We would like to thanks Mr. Grégoire Wälchli for his support and the productive discussions, and Mr. Xavier Horsot for his help and expertise in VHDL and processor areas.

REFERENCES

[1] http://www.gnsssignature.org

[2] F. van Diggelen, "A-GPS: Assisted GPS, GNSS, and SBAS", Artech House, 2009.

[3] J. B.-Y. Tsui, "Fundamentals of Global Positioning System Receivers: A Software Approach" John Wiley & Sons, 2005.

[4] H. Mathis, P. Flammant and A. Thiel, "An analythic way to optimize the detector of a post-Correlation FFT acquisition algorithm", ION GPS/GNSS 2003, Portland, Oregon, USA, September 9-12, 2003.

[5] J.-L. Nagel, C.Botteron, P.-A. Farine, "Fast GPS acquisition using the fast Fourier transform (FFT)", Internal Report, University of Neuchâtel, Switzerland, 2005.

[6] K. Borre, D. M. Akos, N. Bertelsen, P. Rinder, S.H. Jensen, "A Software-Defined GPS and Galileo Receiver", Birkhäuser Boston, 2006.