Università degli Studi di Padova

Dipartimento di Fisica e Astronomia

Corso di Laurea Magistrale in

Fisica

Real-time processing of neuronal network activity measured by a high density microelectrode matrix through a FPGA card

Relatore
Prof. Gianmaria Collazuol

Controrelatore
Prof. Alberto Garfagnini

Laureando
Daniele Guarrera

Correlatore
Prof. Stefano Vassanelli

ANNO ACCADEMICO
2016-2017

Contents

ABSTRACT

This work aims at measuring the electrical activity of a network of rat neurons with Multi-Transistor Array technology. The recorded signal analysis is made using the spike detection algorithm proposed after A. Lambacher ("Identifying firing mammalian neurons in networks with high-resolution multi-transistor array (MTA)", *Applied Physics A* 102.1 (2011), pp. 1-11).
After locating a group of neurons, the spike frequency of one of these is measured. The time correlation between spikes of different neurons allows the meaurement of inter-neuronal signal transmission velocity. Preliminary results are discussed.
Furthermore the algorithm is implemented on a FPGA digital circuit. The on-line digital algorithm is tested and compared with off-line analysis to verify its working. A graphical interface is created to visualize neuronal activity on-line. This work aims at building an acquisition chain for real-time analysis of neuronal activity.

Ringrazio i miei genitori e mio fratello per avermi sempre supportato e per aver reso tutto questo possibile.

Ringrazio Violina, raggio di luce nei momenti più bui, che mi ha supportato nelle mie scelte standomi accanto.

Ringrazio i miei amici per essermi stati vicini in questi anni, regalandomi sempre tantissimi momenti felici.

Ringrazio i ragazzi del collegio che mi hanno accompagnato in questa lunga avventura.

Chapter 1

Introduction

In this chapter it is introduced the physiology of neural communication. Then it is explained the chip used to meausure the neuronal network activity, the cell-chip coupling model and the setup used to perform off-line data acquisition.

1.1 Communication between neurons

Neurons are the signal sources of the biophysical system that the experiment deal with. The neuron is a cell composed of a central body from which different branches originate: the dendrites and the axon. The dendrites receive signals from the afferent neurons and transmit them to the neuron in the centripetal direction. On the other hand the axon conducts the signal in centrifugal direction to other cells. The final part of the axon is an extension called the synaptic button. Through these buttons an axon can bind to dendrites or to the cell body of other neurons so that the nerve impulse propagates with a reaction chain along a neuronal circuit. The Figure 1.1 shows an example of neuronal network cultivated in vitro.

1.1.1 Neuron membrane and resting membrane potential

The membrane is the physical barrier between the intracellular and extracellular environments of the biological cell. It is composed by a lipid bilayer and some proteins that allow or don't allow ions to cross the membrane.
The concentration of ions is different in the cytosol and in the extracellular enviroment: extracellular liquids have an high concentration of Na^+, Cl^- and low concentration of K^+; intracellular liquids have high cencentration of K^+ and large organic anions A^- and low concentration of Na^+. The Figure 1.2 shows the ions concentration gradients scheme. Indeed the difference in concentration of cations and anions from exterior and interior generates the *membrane potential.* The membrane potenzial of a cell that is not conducting is called *resting membrane potential*, kept around -70 mV.
The concentration imbalance is produced by the ion transport mechanisms through the

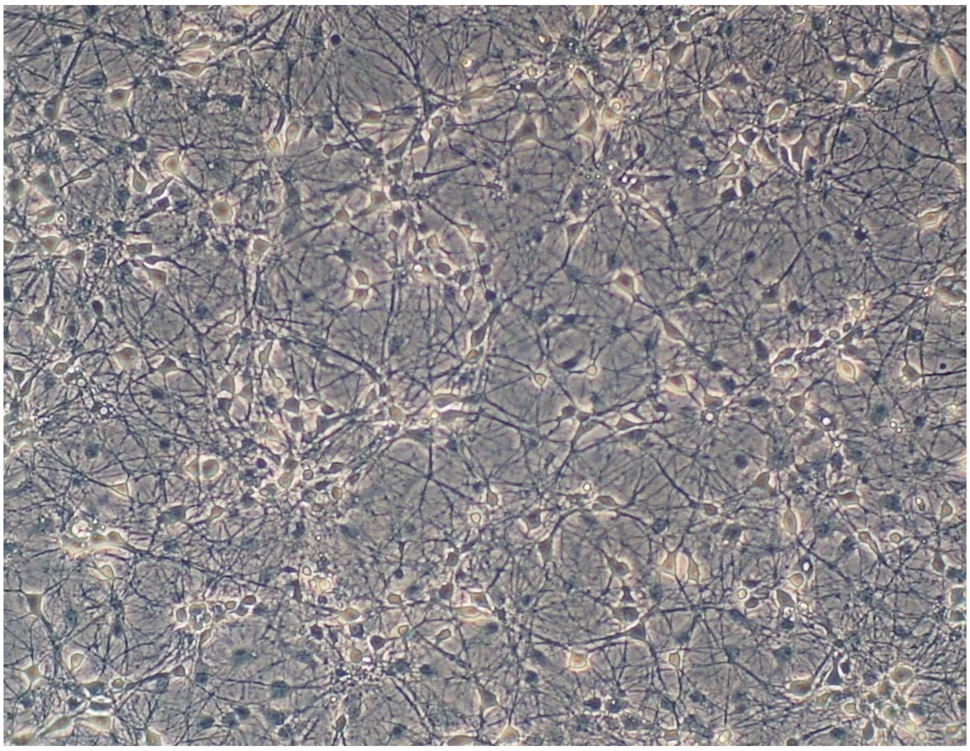


FIGURE 1.1: Example of primary hippocampal neuronal network, cultivated in vitro [15].

membrane itself. The only ions that can effectively cross the membrane of a neuron are Na^+ and K^+. But in a resting cell many potassium channels remain open while many sodium channels are closed. The membrane permeability to K^+ allow potassium to diffuse to exterior, according to the concentration gradient. Because of this A^- charge is not totally balanced from K^+.

1.1.2 Action potential and signal propagation

The most commonly used synonym for the *action potential* is a nervous impulse. An action potential is in fact an electric potential fluctuation that travels along the surface of the membrane of an excitable cell, reaching speeds up to 120 m/s [14].

Action potential phases:

- due to an appropriate stimulation some sodium channels open. Na^+ enter in the neuron and this establishes a **local depolarization**;
- if the depolarization value reaches or exceeds a potential threshold (about -50 mV) many sodium channels open;
- Na^+ internal concentration increases and the membrane potential reverses, reaching up to $+30$ mV;

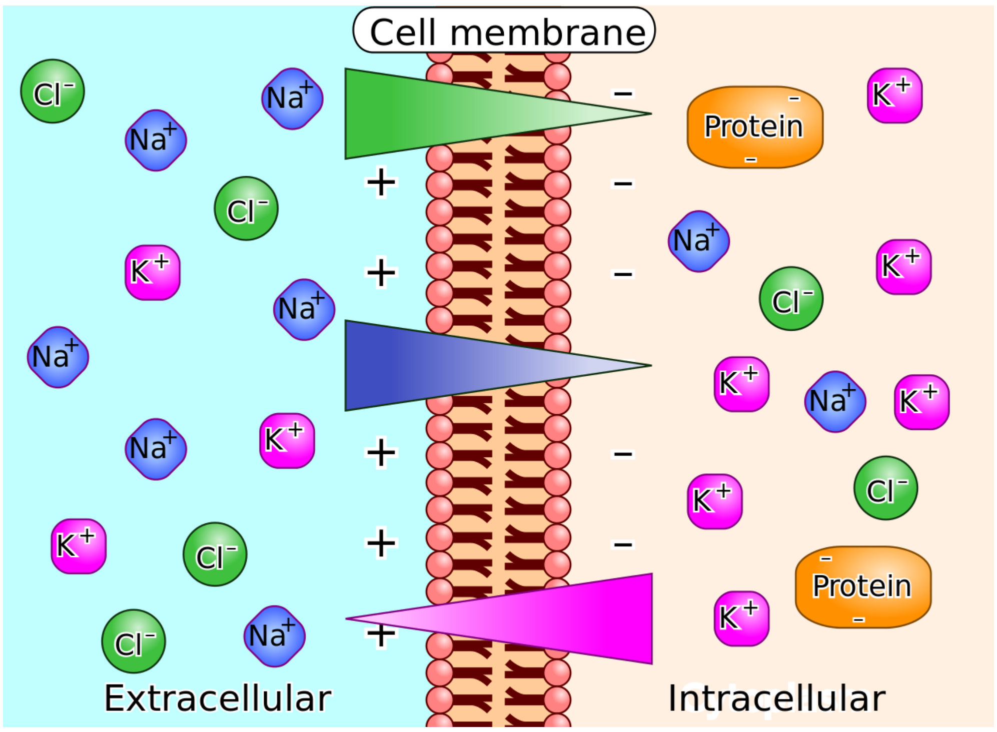


FIGURE 1.2: Neuronal membrane potential, generating by ion gradients between extracellular and intracellular space [11].

- the channels remain open for about 1 ms, causing ever-equal ddp;
- when the threshold of +30 mV is reached K^+ channels open restoring the original negative value of membrane potential;
- then the channel are closed and the $Na^+ - K^+$ pomp restores intial concentration: it carries two ions K^+ inside the cell and three ions of Na^+ outside every time it works.

The period from action potential start to resting membrane potential is called **refractory period** as described in Figure 1.3. One distinction is made between absolute and relative refractory period: during the first one sodium channels are deactivated and no local stimulus generates any response on the membrane (about 0.5 ms); during the second one the membrane responds only to high intensity stimuli (some milliseconds after absolute refractory period).

The local inversion of polarization induces a flux of current between the area where the action potential is generated and near the area of the membrane. Because of this near sodium channels open, generating a new action potential.

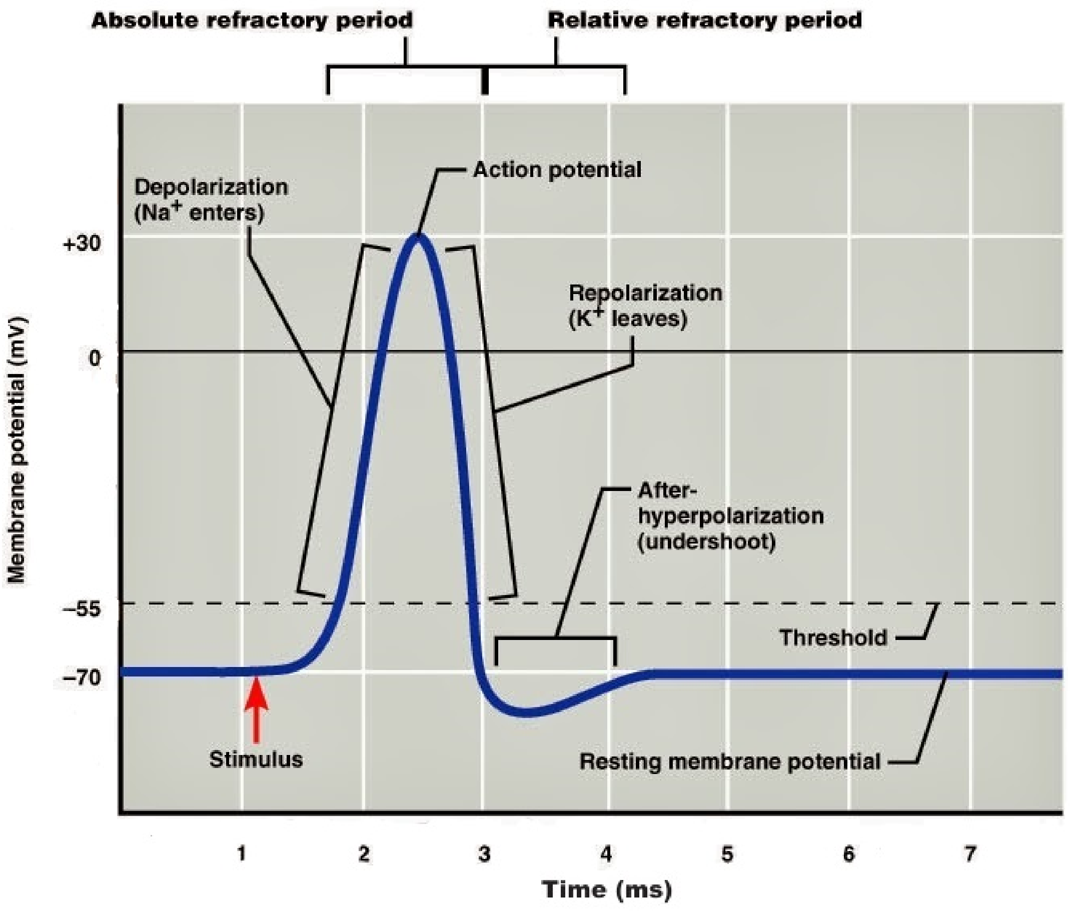


FIGURE 1.3: Schematic of an electrophysiological recording of an action potential, showing the various phases that occur as the voltage wave passes a point on a cell membrane [13].

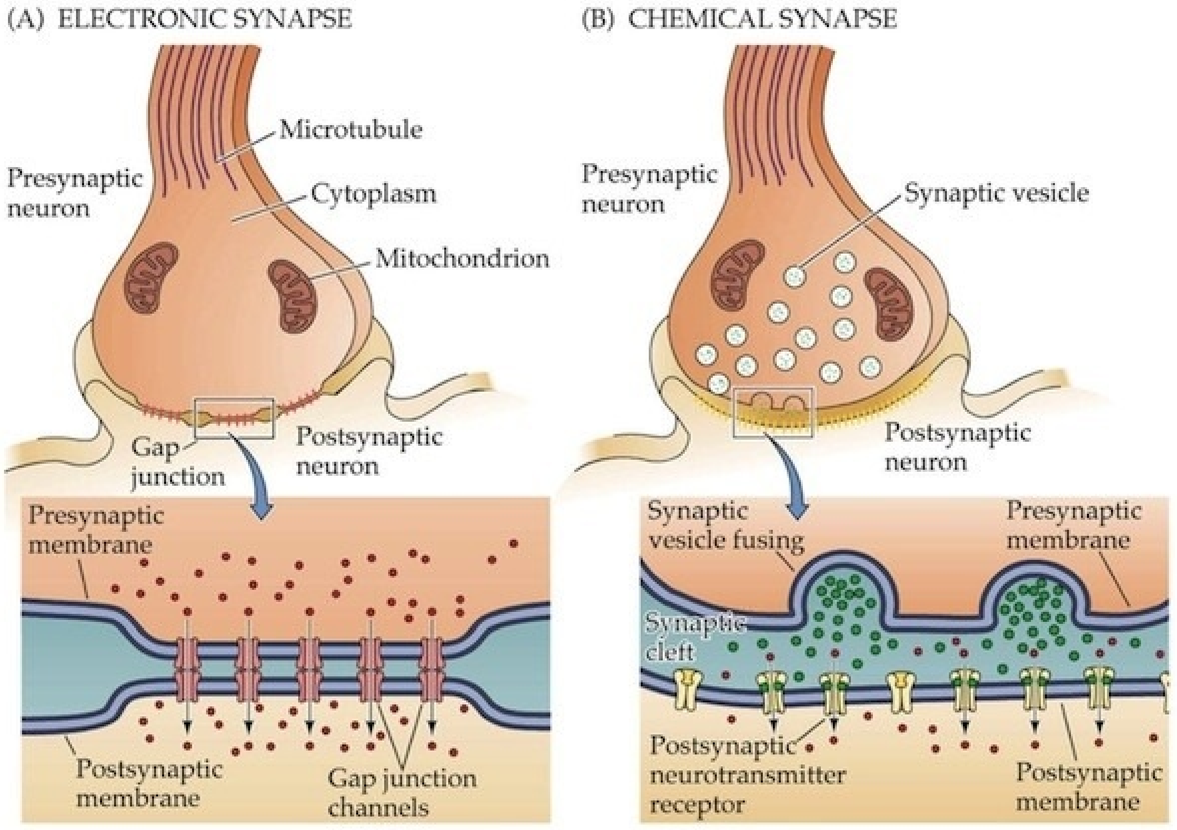


FIGURE 1.4: Schematic representation of electrical and chemical synaptic signal transmission [2].

This "spike train" produces the signal propagation on the membrane of the neuron. Along the axons the signal propagates **unidirectionally** due to the absolute refractory period.

Two neurons are connected by **synapses**. This structure allows neurons to communicate with each other. The action potential can be transmitted through two types of synapses: the chemical synapse or the electrical synapse (Figure 1.4).

The chemical synapse is composed by three elements: the pre-synaptic membrane, the post-synaptic membrane and the synaptic space. The pre-synaptic membrane of the first neuron axon releases neurotransmitters in the synaptic space. These chemical messengers reach the post-synaptic membrane of the second neuron dendrite where the action potential is generated. This kind of process takes about 0.3 ms, reducing inter-neuron signal transmission velocity.

The electrical synapse consists of the connection between two neurons through a gap junction, allowing the direct flux of current. This second type of synapse allows a faster communication and the signal can travel in both direction.

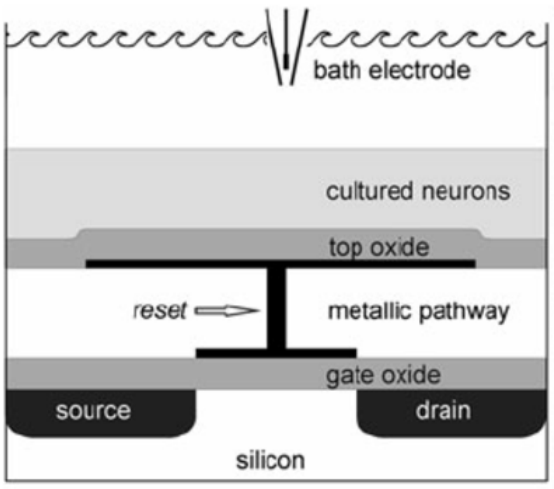


FIGURE 1.5: Schematic cross section of a sensor transistor with an EOMOSFET (not to scale). The neurons are cultivated on the top layer of TiO_2/ZrO_2 that insulates a metallic pathway to the gate oxide of a standard MOSFET with source and drain. Note that there is no floating gate node: the operating point of the MOSFET is adjusted by an autozeroing circuit connected to the gate of the sensor MOSFET, indicated by the "reset" arrow in the figure [1].

1.2 Mapping the neuronal network electrical activity

The diameter of a neuronal cell body is of the order of 20 μm. The action potential duration is about 1 ms. Therefore, a measuring instrument with sufficient spatial and temporal resolution is required. It is possible to couple neural cell with extended CMOS technology. Prof. S. Vassanelli's laboratory makes use of chips with an high density microelectrode matrix on which neuronal cell cultures are performed.

1.2.1 The EOMOSFET and the neuron-chip interface model

The fundamental unit used for measuring the electrical signals of a neuron is the EOMOSFET (Electrolyte Oxide Metal Oxide Semiconductor Field Effect Transistor). The EOMOSFET comes from the redesign of the MOSFET (Metal Oxide Semiconductor Field Effect Transistor). For neural activity recording, this device is used as voltage controlled current generator.
The neurons are cultivated on a top layer of TiO_2/ZrO_2 that insulates a metallic pathway to the gate oxide of a standard MOSFET with source and drain. The neuron culture is surrounded by an electrolyte solution with chemical compounds to allow them to survive for a period of time sufficient to record their electrical activity. The system scheme is reported in Figure 1.5.

The layer of oxide provides a capacitive coupling between the electrolyte and the

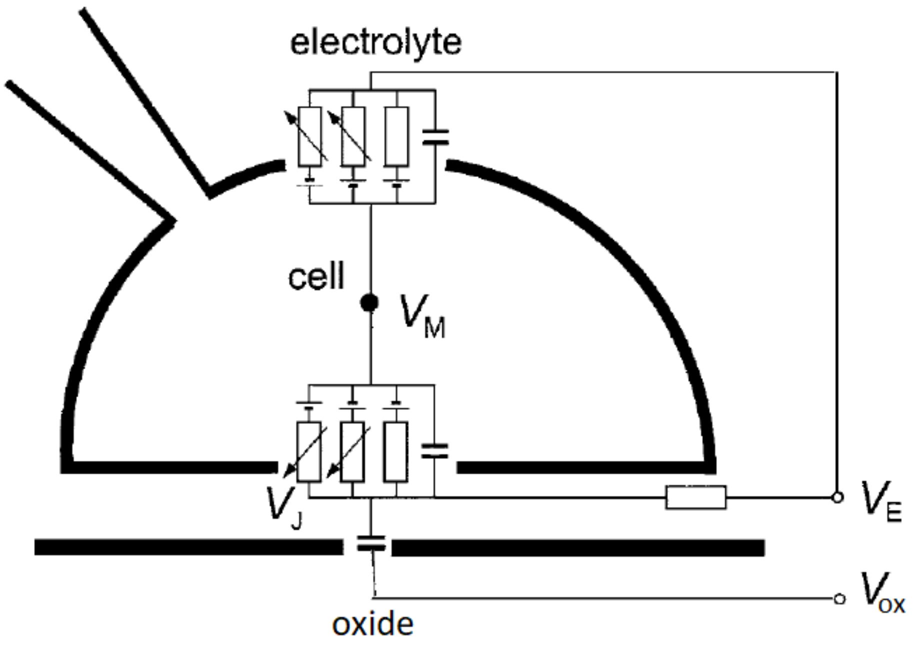


FIGURE 1.6: Point contact model to describe neuron-electrode coupling.

underlying transistor. Variations in the extracellular field potential modulate the gate voltage. By measuring the drain current of the transistor, it's possible to record neuronal activity.

1.2.2 Junction model

To describe the coupling between the neuron and the electrode the *point contact model* [5] is used, represented in Figure 1.6.

The intracellular voltage $V_M(t)$ and the extracellular voltage $V_J(t)$ are related during an action potential. Considering the circuit that describes the ionic and the capacitive currents from internal to the external of the cell (Figure 1.7) and using the Kirkoff's law, we obtain the following:

$$g_j(V_j - V_E) \approx \sum_i g^i_{JM}\left(V_M - V^i_0\right) + c_M \frac{dV_M}{dt} \tag{1.1}$$

$$\approx \sum_i \left(g^i_{JM} - g^i_{FM}\right)\left(V_M - V^i_0\right) \tag{1.2}$$

where c_M is the area-specific capacitance of the membrane. V^i_0 is the reversal voltage of the considered ion, due to difference in concentration from internal and external of the cell. $g_J = \eta_J/(r_J A_J)$ is the area-specific conductance of the cell-chip junction, where r_J

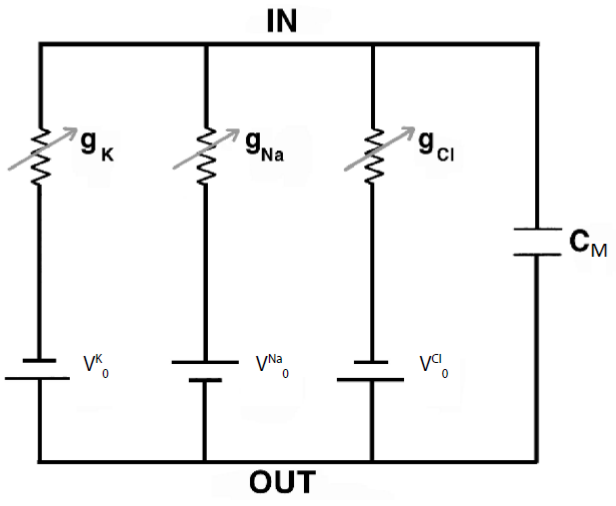


FIGURE 1.7: Parallel-Conductande Model for an Excitable Membrane (IN = intracellular, OUT = extracellular). Indipendent conductances represent K+, Na+ and Cl- channels. Polarity of batteries is chosen in order to associate negative membrane potentials to K+ and Cl- ions and positive to Na+. Cm quantifies the membrane capacitance [6].

and A_J are the sheet resistance and the area of the junction. η_J accounts for the position of the transistor, with $\eta_J = 4\pi$ for a recording in the center of a circular junction. g^i_{JM} is the area-specific ion conductances of the adherent membrane. V_E is the potential of the electrolyte. The Eq. 1.1 is a good approximation for small signals for the junction model ($V_J \ll V_M - V^i_0$ and $dV_J \ll dV_M$). The second approximation (Eq. 1.2) can be done taking into account the difference between the area-specific ion conductances of the adherent membrane g^i_{JM} and the area-specific ion conductances of the free membrane g^i_{FM}.
Then the signal of V_J is capacitively coupled to V_{ox} via the bio-compatible oxide substrate.

1.2.3 Multi-Transistor Array and CAN-Q setup

To measure the electrical signals coming from a neuronal network is used a MTA (Multi-Transistor Array) fabricated with the EOMOSFET technology. The MTA is composed by 256x384 hexagonal pixels with a column pitch of 5.625 μm and a row pitch of 6.5 μm. The area of a single hexagonal pixel is 30 μm^2. A rat neuron covers about 7 near pixels. The full frame sampling frequency is equal to 9375 Hz. In other words the sampling period on the single pixels is about 106 μs, which is adeguate for accurate measurement of the action potential waveform that lasts about 1 ms.
The MTA is in the center of a support designed to allow neuronal cultivation on it. This support can be coupled with the CAN-Q ® Station [8] for the data acquisition, as shown in Figure 1.8 and 1.9.

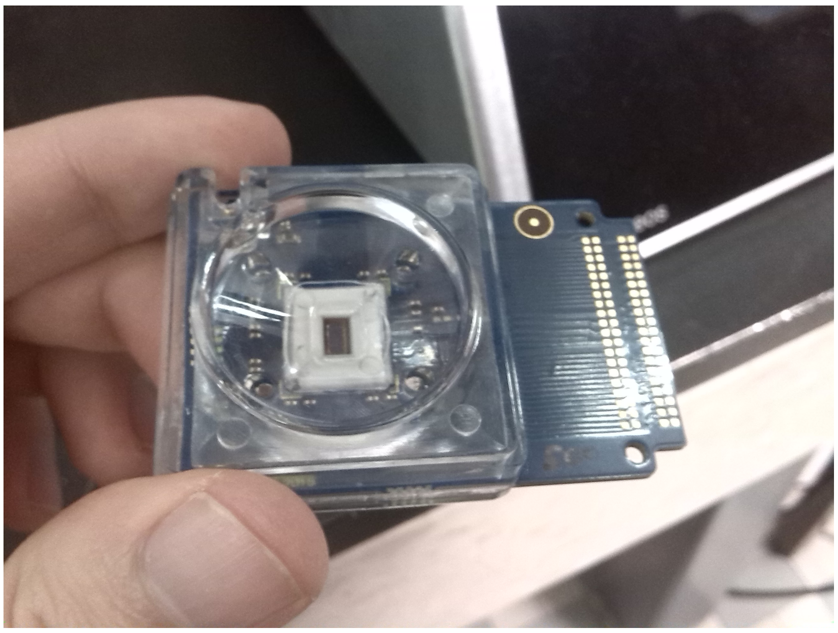


FIGURE 1.8: $9.2\,mm^2$-MTA chip with support. It is designed to immerse the chip in a elecrolyte solution bath.

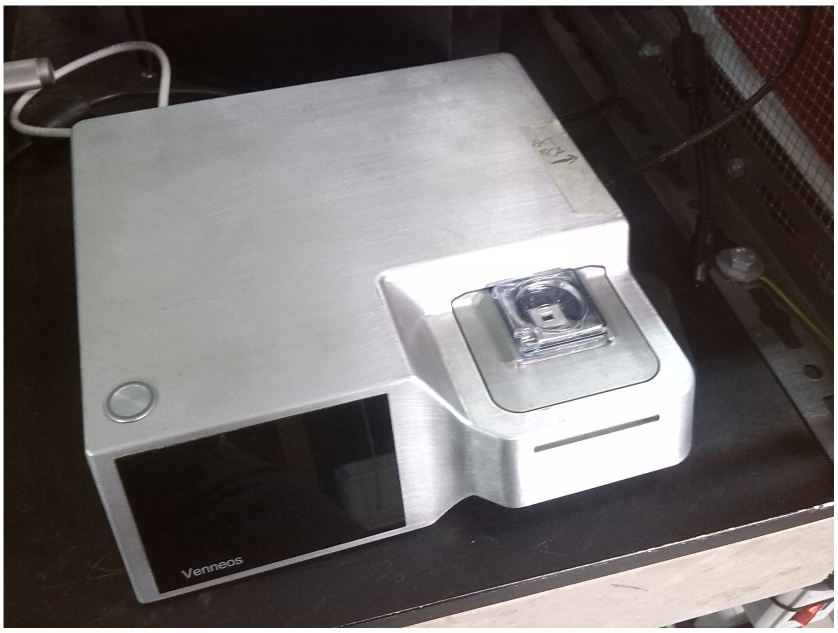


FIGURE 1.9: CAN-Q setup coupled with the chip support, put in the right slot.

The Figures 1.10 and 1.11 show the block scheme of signal flow of the CAN-Q system. On the left side, there is the CAN-Q chip including the active matrix of 256x384 pixels. The matrix is divided in west and east side. Each pixel can be connected to the west row amplifiers or east row amplifiers according to pixel's side: there are 256 1^{st}-stage amplifiers on each side.
Also there are 256 2^{nd}-stage amplifiers on each side, which include a sample and hold circuit. This allows the whole column (per side, so actually 2 columns) to be sampled at the same time.
Finally there are 16 3^{rd}-stage amplifiers on each side, which are connected subsequently to 1 of 16 2^{nd}-stage amplifiers each.
Sequencers on the chip control the connection of columns to the readout amplifier chain and the multiplexing from stage 2 to stage 3.
The outputs of the 2x16 3rd stage amplifiers are current outputs. Transimpedance amplifiers translate the currents into voltages for acquisition by the ADCs (Analog to Digital Converter). The ADCs have a sampling rate of currently 18 MHz and a resolution of 14-bit. The 4 ADCs have 8 channels each, the samples are serialized and sent to the DSPs.
The 2 DSP (Digital Signal Processing) FPGAs (Field Programmable Gate Array) are Xilinx Artix-200. They work in parallel, one on the upper half of the matrix, the other on the lower half. Inside, the data from the ADCs are deserialized. The the samples are streamed to the DDR (Double Data Rate) memory with a datamover IP (Intellectual Property) core. The same core can retrieve the data from memory and send it via a serial link to system handling FPGA.
The system handling FPGA is a Xilinx Zynq (FPGA + Arm core). It is a module from National Instruments (sbRIO-9651). The sbRIO runs a real time Linux. The sbRIO controls the whole system including the DSP modules and streams the acquired data from the DSPs to the connected host via a gigabit ethernet connection. The data are serialized as a bus of 16-bit signed numbers.
In order to control data in real-time is available the CAN-Q acquisition software [9], to be installed on an host PC. It can be used by connetting the CAN-Q to internet. Because of this the CAN-Q and the PC are connected to a router.

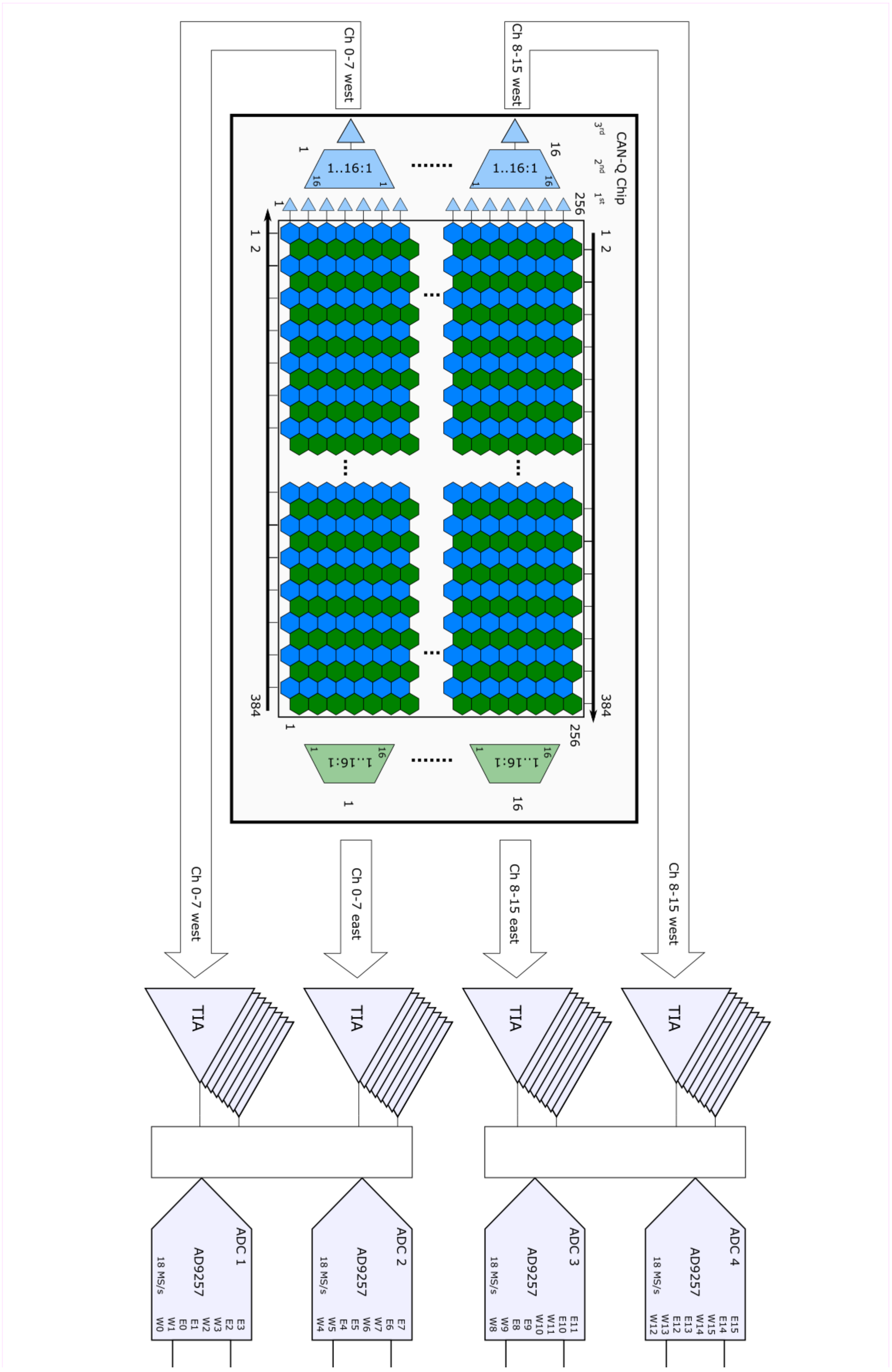


FIGURE 1.10: Block scheme of signal flow of the CAN-Q system, provided by Venneos.

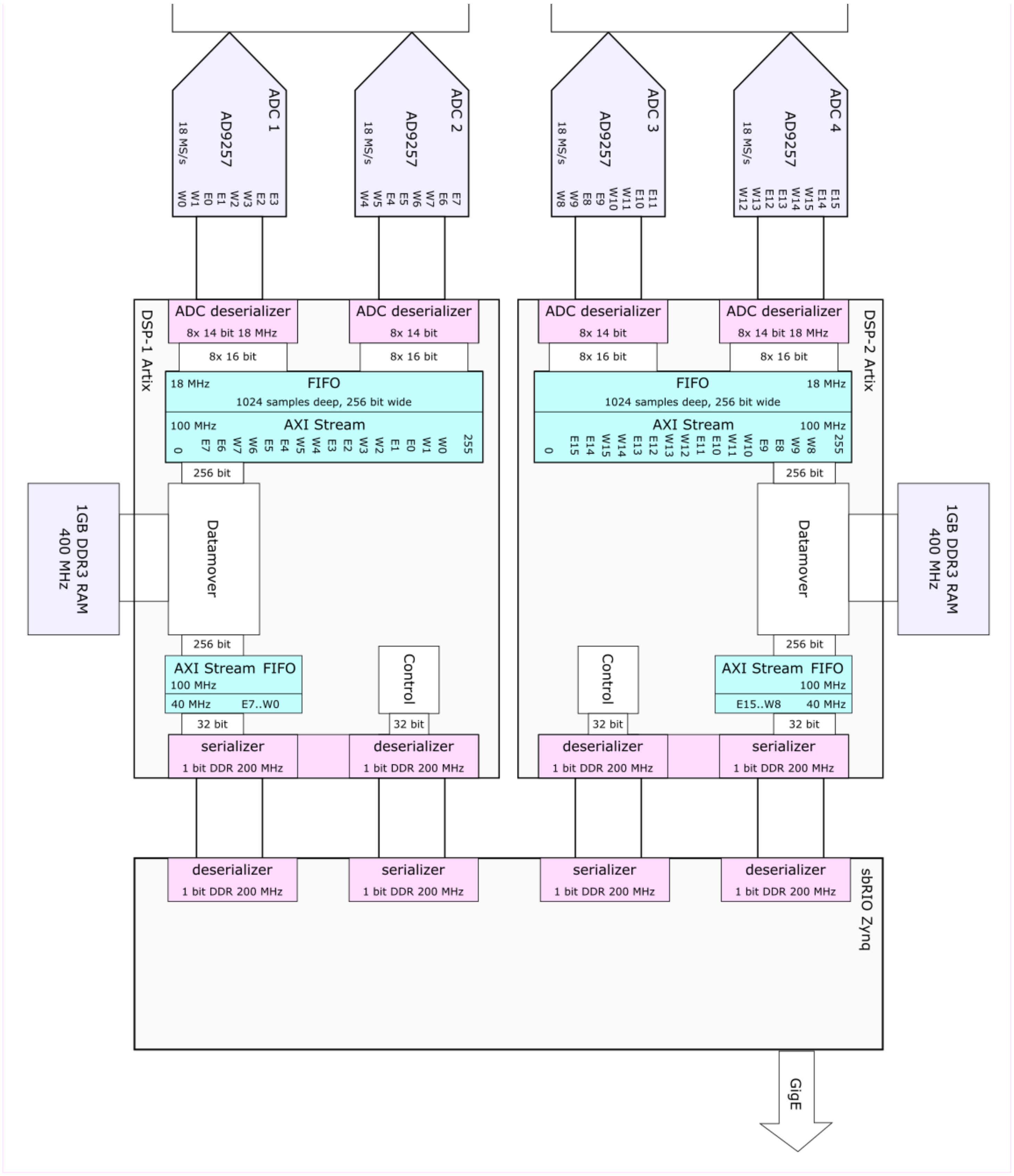


FIGURE 1.11: Block scheme of signal flow of the CAN-Q system, provided by Venneos.

Chapter 2

Spike detection

To identify neuron spikes from chip noise it is important to reduce the number of false positive per second and to detect correlated events. The noise has a gaussian distribution around the zero baseline, as it can be seen in Figure 2.1.

The probability to have only one false positive per second on a pixel is calculated as the inverse of sampling rate of a pixe. So we can impose that

$$P(|V| > V_0) = \int_{V_0}^{\infty} dv \, \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{v^2}{2\sigma^2}\right) = (9375)^{-1} \tag{2.1}$$

Resolving Eq. 2.1 numerically we obtain that $V_0 = 3.8\,\sigma$. However the number of false positive events can be reduced thanks to time and space resolution.

2.1 Correlation algorithm

It is used the spike detection algorithm of A. Lambatcher [1] this algorithm takes advantage of space-time-correlation. Since a pixel is sampled multiple times during an action potential event and the matrix pixel is smaller than the size of a neuron, the same action potential is detected in several consecutive time frames and at several adjacent sites. As a matter of fact the duration of an action potential is about 1 ms and the samplig period is about 106 μs; a neuron covers about 7 adjacent pixels.
It can be considered the joint probability to see the same event on 9 sampling period intervals and 7 adjacent exagonal pixel, doing the substitution $v_i' = v_i/\sigma_i$

$$P(|\mathbf{V'}| > V_0') = \prod_{i=1}^{9\times7} \int_{V_0'}^{\infty} dv_i' \, \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{v_i'^2}{2}\right) = (9375)^{-1} \tag{2.2}$$

where $\mathbf{V'}$ is a 9×7-dimensional vector and V_0' is the radius of a 9×7-dimensional sphere. To resolve Eq. 2.2 numerically, it must be put in a better form. The general problem is

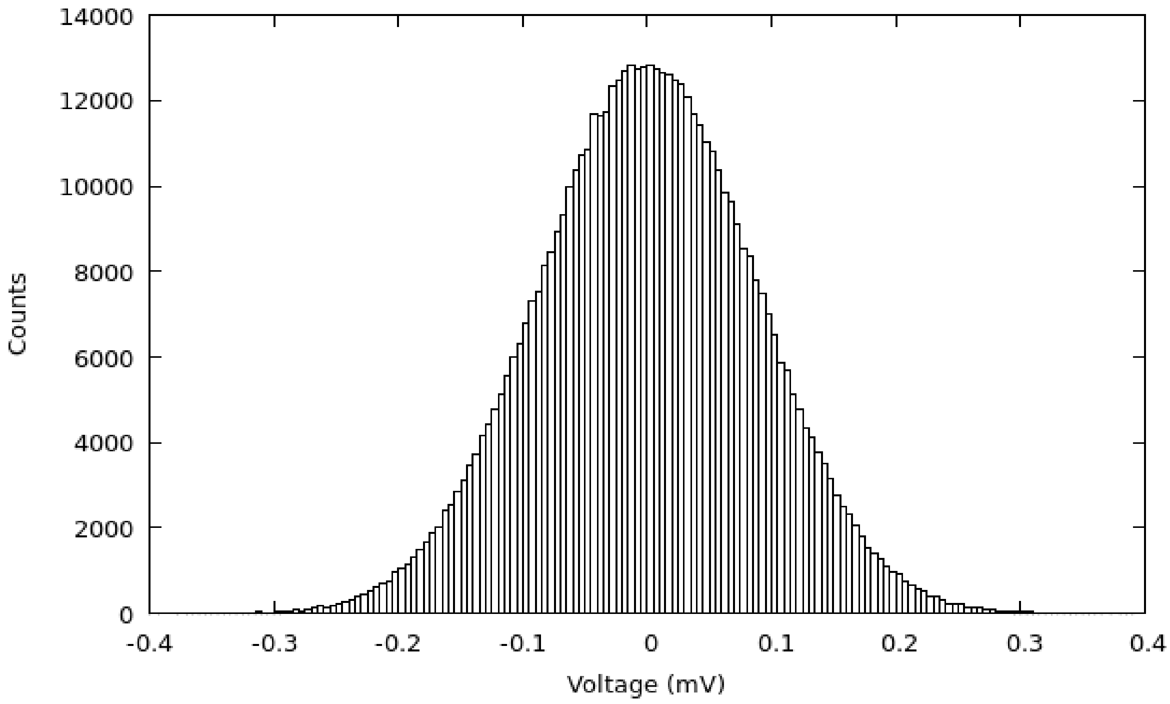


FIGURE 2.1: Gaussian distribution of a pixel filtered signal.

solved for the n-dimensional case:

$$P(|\mathbf{V'}| > V_0') = \int_{V_0'}^{\infty} dv_1' ... \int_{V_0'}^{\infty} dv_n' \, (2\pi)^{-\frac{n}{2}} \exp\left(-\sum_{i=1}^{n} \frac{v_i'^2}{2}\right) \tag{2.3}$$

Passing to n-dimensional spherical coordinates [17] it's possible to write:

$$r^2 = \sum_{i=1}^{n} v_i'^2 \tag{2.4}$$

$$P(|\mathbf{V'}| > V_0') = \int_{r_0}^{\infty} dr \;\; (2\pi)^{-\frac{n}{2}} \, r^{n-1} \exp\left(-\frac{r^2}{2}\right) \int_{\Omega_n} d\Omega_n \tag{2.5}$$

where Ω_n is the n-dimensional solide angle [16] and it can be computed explicitly as follows:

$$\Omega_n = \frac{(2\pi)^{\frac{n}{2}}}{\Gamma\left(\frac{n}{2}\right)} \tag{2.6}$$

$$\Omega_n = \begin{cases} \frac{1}{\left(\frac{n}{2}-1\right)!} \, (2\pi)^{\frac{n}{2}}, & n \text{ even} \\ \frac{\left(\frac{1}{2}(n-1)\right)!}{(n-1)!} \, 2^n \pi^{\frac{1}{2}(n-1)} & n \text{ odd} \end{cases} \tag{2.7}$$

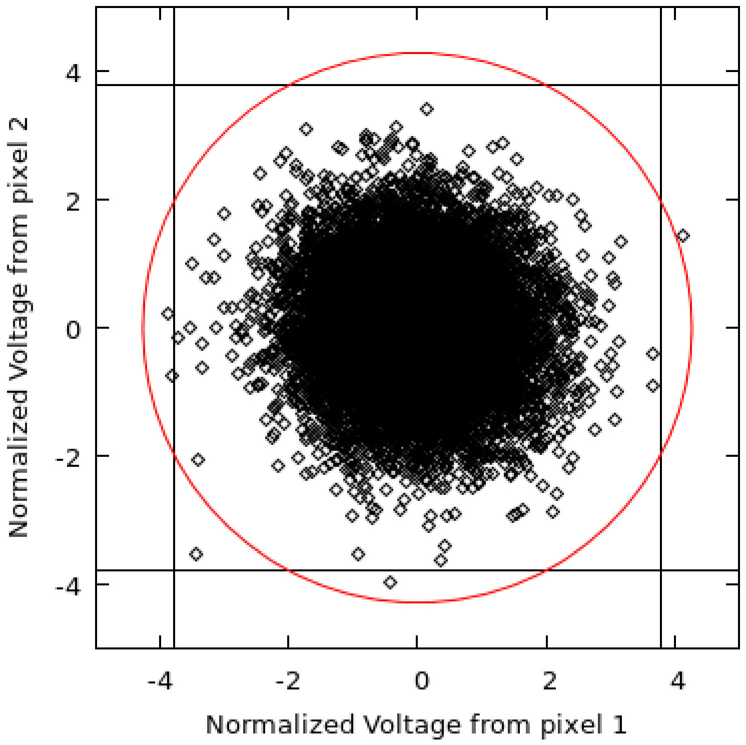


FIGURE 2.2: Plot of normalized couples $(V_1'(t), V_2'(t))$ of two near pixels. The vertical and horizontal lines identify the uncorrelated pixels threshold (3.8), the radius of the circle identify the correlated pixels threshold (4.3).

The final equation to compute r_0 is obteined imponing the value of probability:

$$\int_{r_0}^{\infty} dr \;\; (2\pi)^{-\frac{n}{2}} \, r^{n-1} \exp\left(-\frac{r^2}{2}\right) \Omega_n = b \tag{2.8}$$

$$\int_{r_0}^{\infty} dr \; r^{n-1} \exp\left(-\frac{r^2}{2}\right) = \frac{(2\pi)^{\frac{n}{2}} \, b}{\Omega_n} = \Gamma\left(\tfrac{n}{2}\right) \cdot b \tag{2.9}$$

where $b = (9375)^{-1}$ is the inverse of the sampling rate of one pixel.
For a better understanding it is shown the two-dimensional case in Figure 2.2.
The points in the graphic are the couple $(V_1'(t), V_2'(t))$ of two adjacent pixels. It is computed the threshold in the case of uncorrelated events and spatial correlated events. As can be seen from the figure the 2-pixels threshold **reduces** the number of uncorrelated false positive events, but also it **includes** a correlation event that is rejected by the previous threshold.
Resolving numerically Eq. 2.9 can be get the value of r_0 for different values of n. In the Table 2.1 are reported the values of r_0 for different values of timeframes and near pixels convolution.

Time Frames x Near Pixels	Threshold
1 x 1	3.8
2 x 1	4.3
3 x 1	4.7
4 x 1	5.0
5 x 1	5.3
6 x 1	5.6
7 x 1	5.8
8 x 1	6.1
9 x 1	6.3
9 x 2	8.0
9 x 3	9.4
9 x 4	10.6
9 x 5	11.7
9 x 6	12.7
9 x 7	13.6

TABLE 2.1: Table of threshold values for some values of consecutive timeframes and near pixels.

In Figures 2.3, 2.4, 2.5, and 2.6 show the number of spikes detected over the 32x32 matrix with increasing degrees of correlation. As can be seen the algorithm reduces the number of false positive events, allowing the localization of neurons.

The final condition of spike detection is that the module of the 9x7-dimensional vector is higher than r_0(9x7)

$$\sum_{i=1}^{9\times 7} \frac{V_i^2}{\sigma_i^2} > (13.6)^2 \tag{2.10}$$

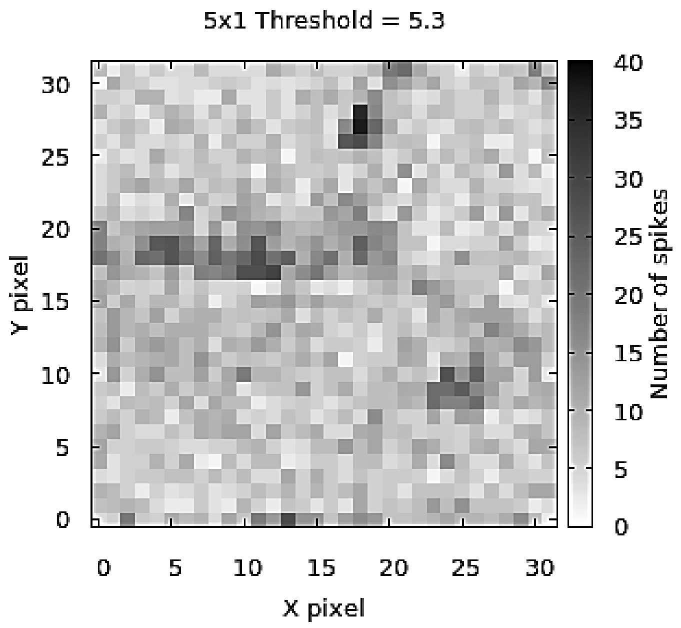


FIGURE 2.3: Activity map recorded over 1s, obteined with 5x1 degrees of correlation.

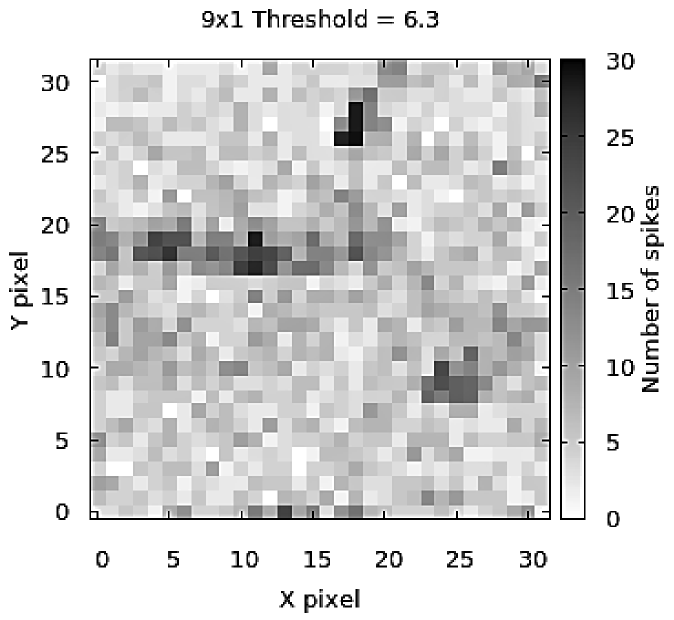


FIGURE 2.4: Activity map recorded over 1s, obteined with 9x1 degrees of correlation

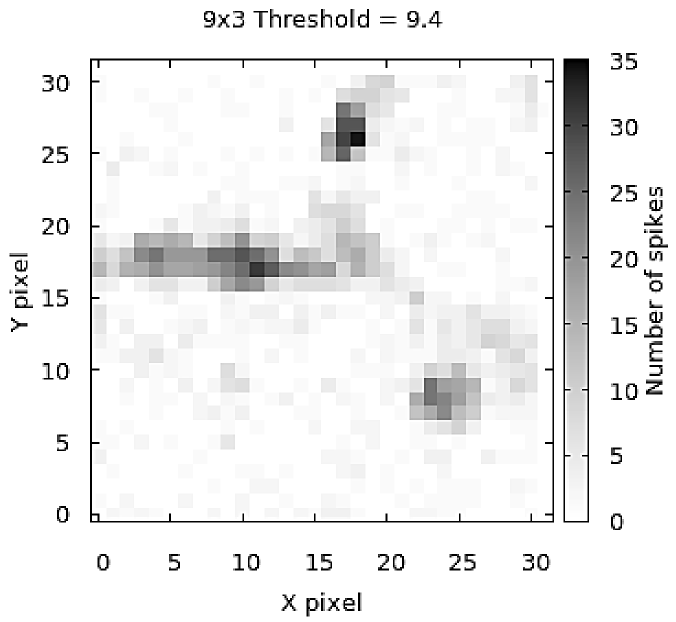


FIGURE 2.5: Activity map recorded over 1s, obteined with 9x3 degrees of correlation

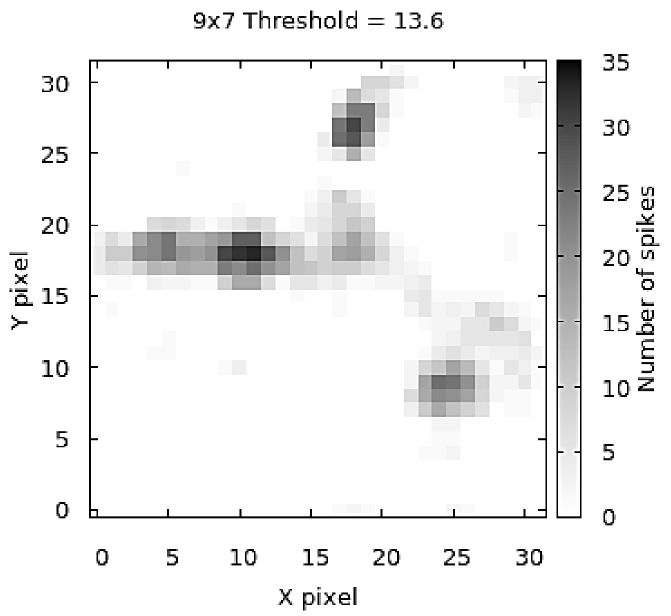


FIGURE 2.6: Activity map recorded over 1s, obteined with 9x7 degrees of correlation

Chapter 3

Measurements and off-line data analysis

In this chapter are illustrated the neuronal cells plating procedure on chips and the procedure of data analysis after the data acquisition. After explained how to process the signals we focus on the neuronal activity analysis of the chip 678 recording. We manage to measure the spiking frequency of a neuron and the inter-neuron signal transmission velocity.

3.1 Chip preparation

Before the neuronal network activity measurement, the plating of neurons on the chips takes place. It is foundamental to follow some steps to ensure the best culture possible on the chip.

3.1.1 Chip cleaning and sterilization

First of all the chips must be cleaned and sterilized: 1 minute of gentle cleaning with 70°C warmed 5% (v/v) Tickopur R33 detergent (Bandelin), followed by extensive rinsing with deionized water; under sterile laminar hood, 30 minutes of incubation with 70% (v/v) ethanol, followed by three washings with sterile deionized water. After drying of the chip under the laminar flow, the chips are exposed to a germicidal lamp for an additional 30 minutes sterilization.
This procedure must be done for every chips after measuraments.

3.1.2 Getting neurons

The neurons are obtained from E18/19 Wistar rats embryos. The procedure for hippocampi dissection is explained in the article [4]. The hippocampus is dissected during the 18th or 19th day of rat pregnancy because its development becomes complete in that period. Furthermore in the hippocampus there are neurons but also glial cells. Neurons don't divide through mytosis but glial cells yes, so the glial cells number increases exponentially

in time. Because of this it's crucial not to wait beyond the 19th day.
Primary neurons are dissociated from hippocampi through the action of a dissociating protein (trypsin). Then there is the so called "pre-plating" step. It allows to reduce the amount of glial cells in the cultures, without removing them completely and avoiding the use of chemical compounds. It consists in a physical separation of glial cells: neurons and glial cells are plated together on a Petri dish where only glial cells are able to adhere, thus making it possible to separate them from the neurons.

3.1.3 Neuronal cells plating

Sterile chips are incubated with a 20 $\mu g/ml$ poly-L-Lysine solution at room temperature under a sterile laminar hood to coat the chip surface with the adhesive protein. Then the protein is removed and the chip is dried under the laminar hood. This protein is necessary for a good adhesion of the neurons on the chip surface.
1600 neuronal cells/mm^2 are seeded on the chip surface ($A_{chip} = 9.2\ mm^2$) diluted in NeuroBasal medium added with 1% (v/v) Glutamax-1, 2% (v/v) B27 supplement (Gibco) and 25 μM Glutamate. Every 3 days, 2/3 of the medium is changed with NeuroBasal medium added with 1% (v/v) Glutamax-1 and 2% (v/v) B27 supplement.
Then chips with plated neuronal cells are maintained in a humidified incubator at around 37°C and 5% CO_2.

3.2 Blue chips for measurement

Every chips with plated neurons are maintained in the incubator at least until DIV (Days In Vitro) 14 to let neurons grow and connect each other. After DIV 14 dentrites and axons are properly developed, branched and well spread to allow connetions between neurons [7]. The measurements reported are done at DIV 18.

3.2.1 Conditions of measurement

The first condition to choose is the type of electrolyte for the neuronal culture bath.
For these measuraments three different solutions are used, with the following compositions (the components are given in mM unit, unless otherwise stated):

- Solution 1: 135.0 NaCl, 5.4 KCl, 1.0 $MgCl_2$, 1.8 $CaCl_2$, 10.0 glucose, 5.0 HEPES (adjusted to pH 7.4 with 1 M NaOH)
  This solution provide the essential minerals for neurons to work in a living-type ambient;

- Solution 2: has the same components of solution 1, but with 10 μM of $1(S), 9(R) - (-) - BICUCULLINE\ METHIODIDE$ (Sigma Aldrich)
  This solution should increase the spontaneous activity of neurons.

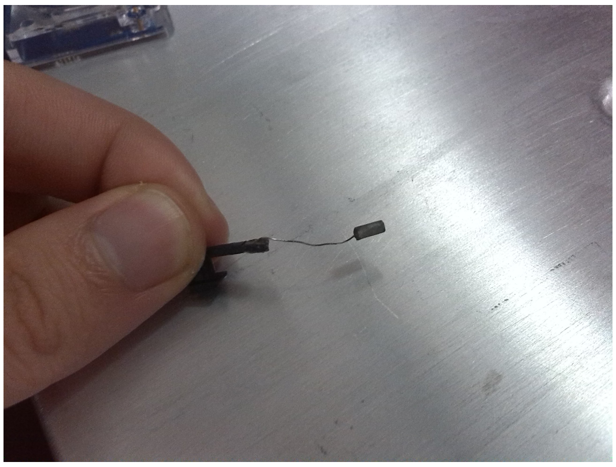


FIGURE 3.1: Ag | AgCl bath electrode. It is used to set the ground terminal.

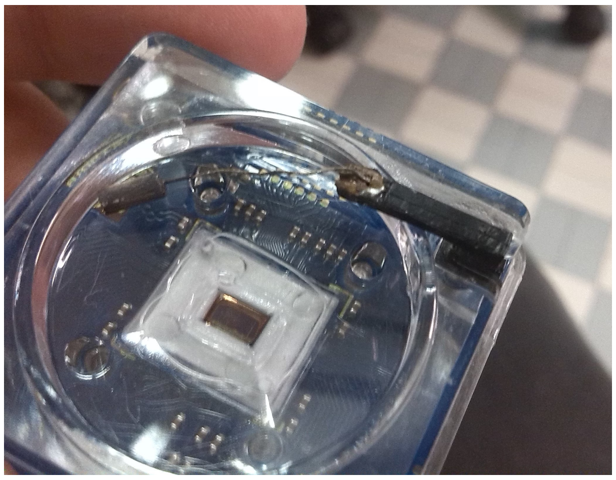


FIGURE 3.2: Bath electrode coupled with the chip slot.

- Solution 3: it is solution 2 with the addition of 250 μL of KCl 1M. The final concentration of KCl is about 100 mM.

The culture medium used during neuronal network growing is replaced carefully and slowly by the new solutions using a pipette.
The next very important step is to keep the bath temperature stable. The CAN-Q setup has a Peltier device under the chip slot. By the CAN-Q software it's possible to set the temperature: for each measure it is set equal to $37°C$, the physiological temperature to let neurons to survive.
At the end, before recordings, the ground electrode is immersed in the electrolytic solution. The electrode and the electrode coupling are shown in Figures 3.1 and 3.2.

3.2.2 RMS map

To understand where the neurons clustered on the chip the RMS map of the full 256x384 matrix can be recorded. The CAN-Q software shows an RMS heatmap of the matrix. The difference of RMS is due to the different value of conductance between the electrolyte solution and the recording electrode. Indeed it is an index of cells adhesion on the chip surface. This imaging technique is usefull to localize neurons. Sometimes the presence of glial cells growing near neurons can disturb their localization.
The immage of a neuron culture is also photographed thanks to an immage acquisition device coupled with an optical microscope. The immage of the neuron culture is shown in Figure 3.3.
To show that the RMS map is closed to the microscope immage it is built the overlap, as can be seen in Figure 3.4.

3.2.3 Data acquisition

Identified the areas where probably there are clusters of alive neurons on the chip, the CAN-Q software allows to choose a 32x32 pixels area for data acquisition. It is possible to select the acquisition time in ms, for a maximum of 60 s.
Then data are saved in a HDF5 file in a 3-dimensional matrix (2 dimension for the array position and 1 dimension for the time). The value of a voltage measurament is a 16-bit signed integer. For a 1 minute acquisition it is generated a file of about 1 GB.
The data are recorded in arbitrary unit so it is necessary to perform a calibration. The CAN-Q software allows the acquisition of a known-voltage-value signal to calibrate the data at a later time.

3.3 Measurements and data analysis

For each chip about 15 recordings of 60 s are acquired: the signals with the three different solutions are recored for 4/5 minutes. The off-line data analysis is performed by Python scripts.

3.3.1 Amplitude calibration

Raw CAN-Q data aren't calibrated in amplitude and present an offset. This is due to the variability of transistors parameters. The CAN-Q system allows to calibrate by injecting a signal of 50 Hz with amplitude peak to peak of 4 mV. In one acquisition it's possible to record the calibration signal for the 32x32 matrix. It is recorded for about 15 seconds every time chip solution is changed. The Figure 3.5 shows an example of it. As can be seen the signal hasn't a good squared shape.

The Figure 3.6 shows the distribution of the calibration signal values. As can be seen there are two predominant peaks. The distance between the two maximum points is computed as a measurement of the amplitude A. Then the amplitude calibration factor

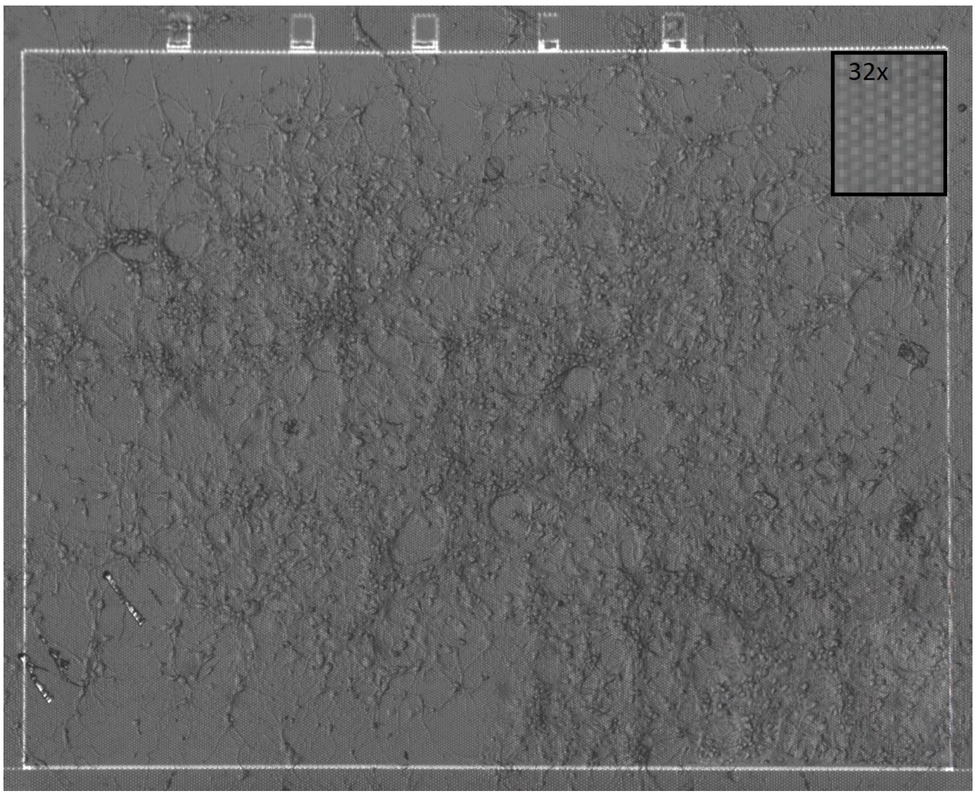


FIGURE 3.3: Example of neuronal culture photographed through an optical microscope.

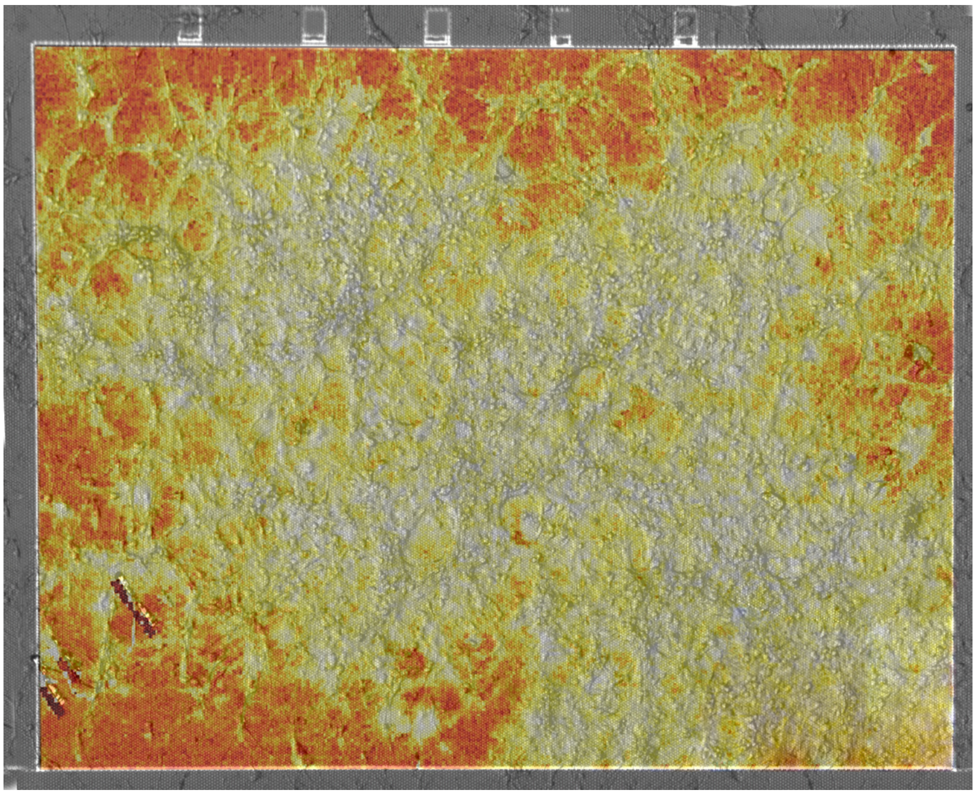


FIGURE 3.4: Overlap between neuronal culture immage and RMS map recorded with CAN-Q Analyzer.

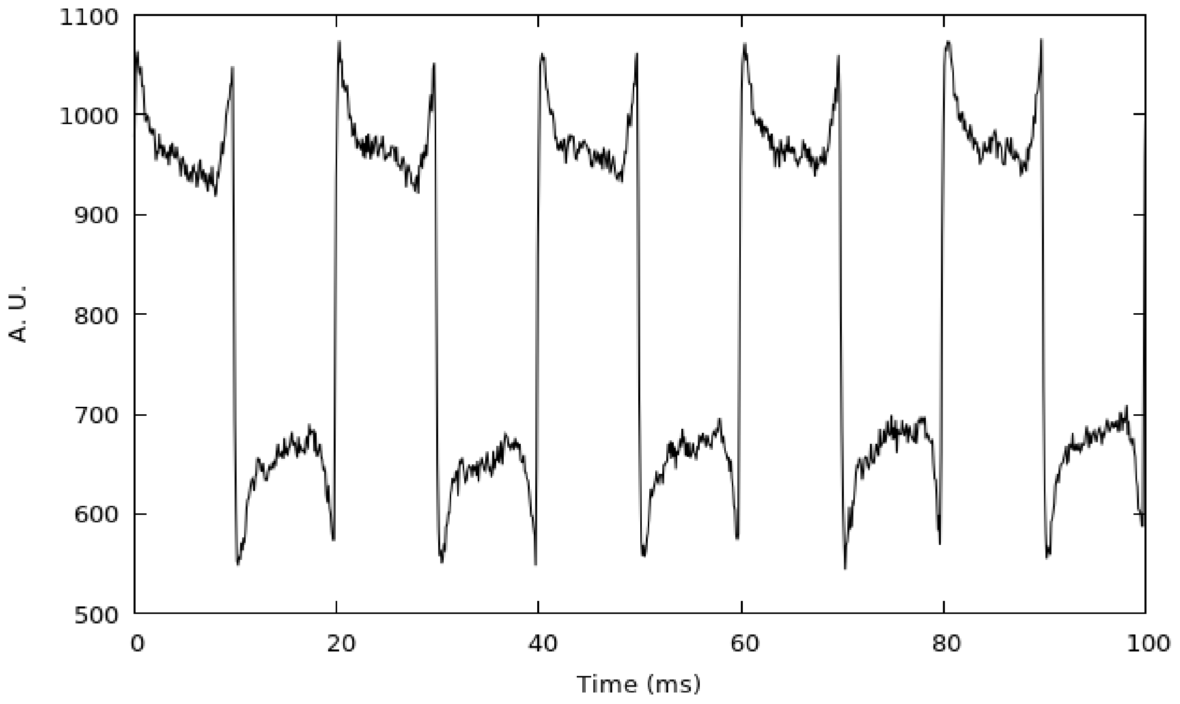


FIGURE 3.5: Signal recorded for data calibration. It is supposed to be a 50 Hz square wave signal of 4 mV peak to peak amplitude.

$CF = A/4\ mV$ is calculated. The Figure 3.7 shows the calibration factor map. As can be seen from it the factor value changes significantly from different pixels.

To calibrate the signal $V_i'(t)$ of the pixel i the following formula is used:

$$V_i(t) = \frac{V_i'(t)}{CF_i} \tag{3.1}$$

The Eq. 3.1 doesn't care about the signal offset, correcting only the peak to peak amplitude of the signals. The Figure 3.8 shows an example of an amplitude-calibrated signal.

3.3.2 Filtering

When measured by means of extracellular currents, spikes usually range in the bandwidth between 300 Hz and 3 kHz. Because of this the signals are filtered with a bandpass butterworth filter in that range. The Figure 3.9 shows the signal in the Figure 3.8 filtered. As can be seen the original offset is canceled by the filtering. This is a fundamental condition to perform the spike detection algorithm. The signal distribution is reported in Figure 2.1, in chapter 2.

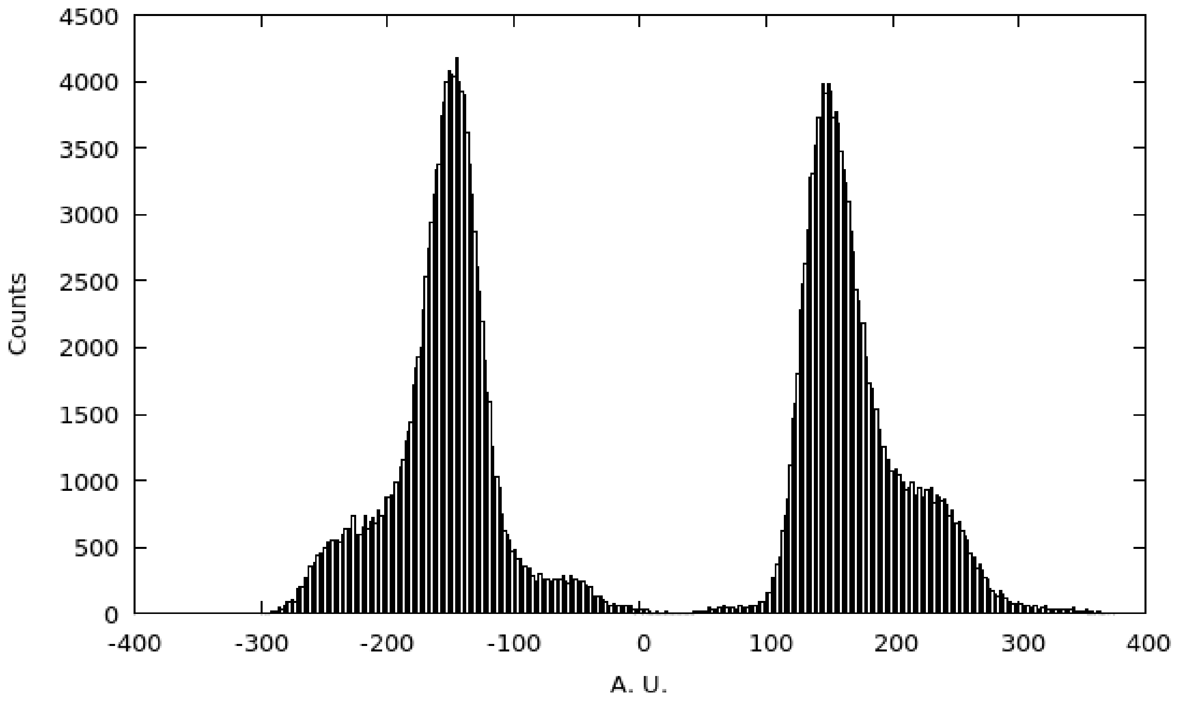


FIGURE 3.6: Distribution of 15 s rocorded calibration signal.

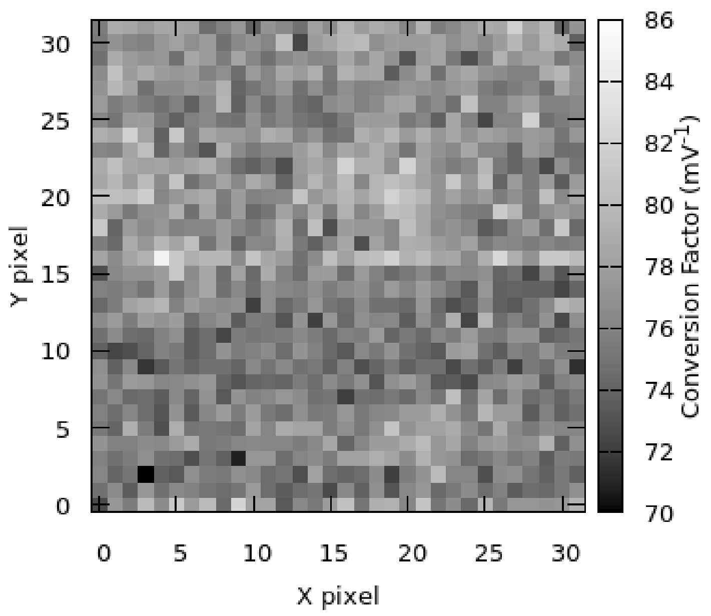


FIGURE 3.7: Calibration factor value over the 32x32 matrix.

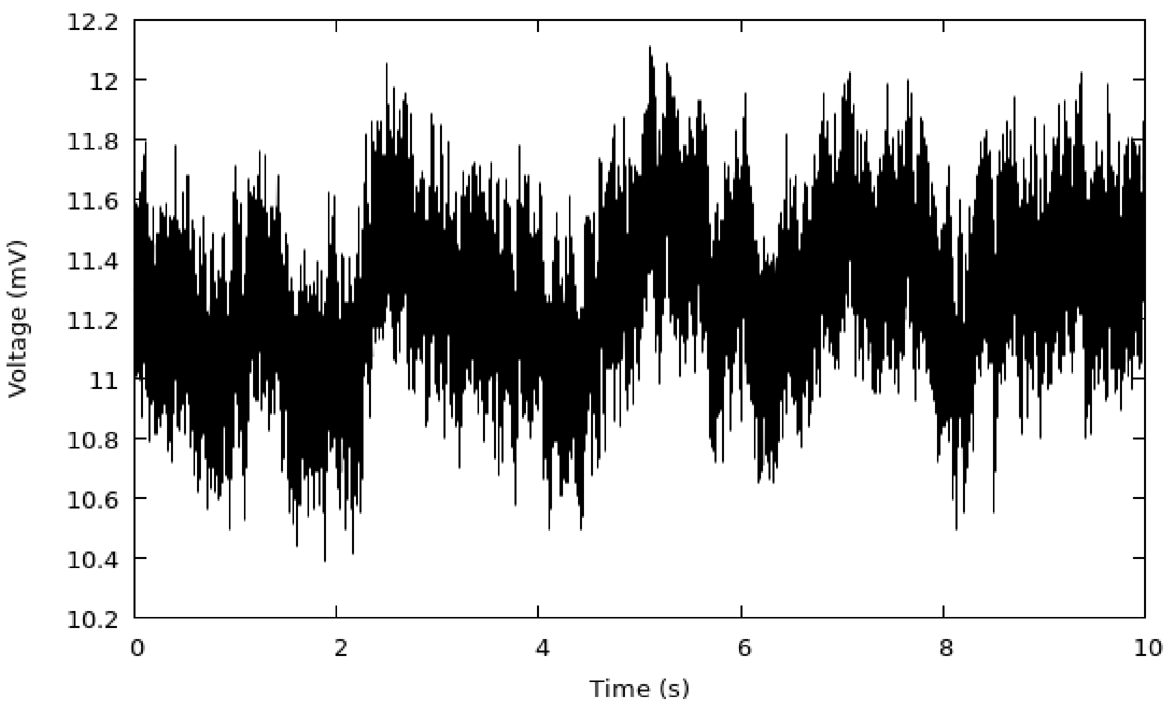


FIGURE 3.8: Calibrated signal from a matrix pixel of 10 s duration.

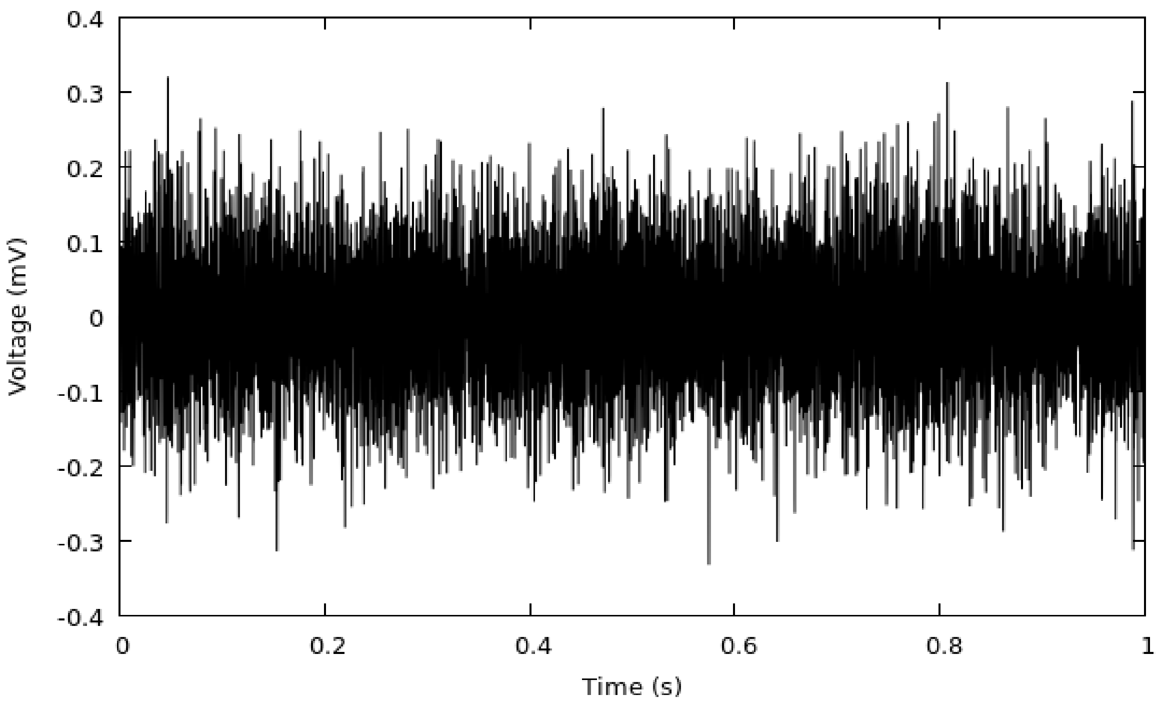


FIGURE 3.9: Filtered signal of 1 s duration. The offset and low frequencies disappeared.

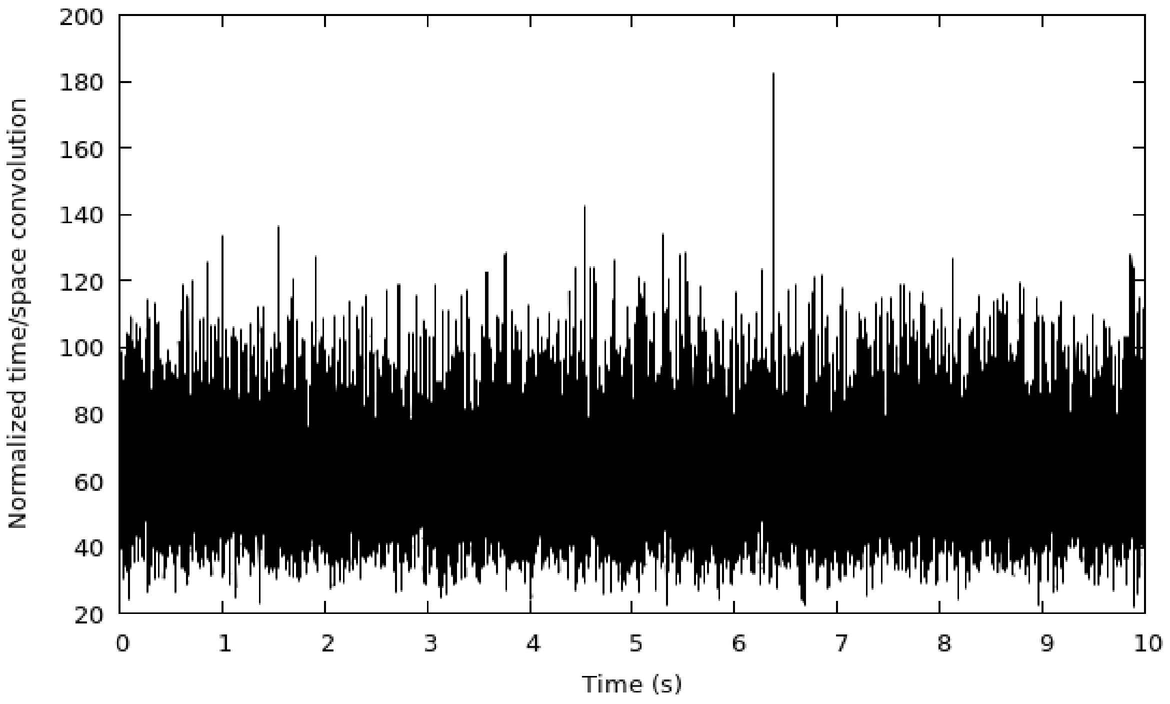


FIGURE 3.10: 9x7 convolved signal over 10 s.

3.3.3 Correlation algorithm

The correlation algorithm condition (Eq. 2.10) is applied to data. The significant steps of signal processing by this condition are:

- squaring of the signal;
- summation over 9 consecutive timestamps;
- division by σ^2;
- summation over 7 near pixels;
- comparison the output with the threshold.

The Figure 3.10 shows the signal 9x7-convolved over 10 s.

As can be seen from Figures 3.11 and 3.12 in the case of extracellular solution and the extracellular solution with bicuculline the number of spikes recorded over 5 minutes is **very low and distributed evenly** (maybe false positive events). Furthermore the number of spikes with bicuculline solution is lower than the solution without over the same time. In other cultures analyzed it happens that the number of spikes is higher with the bicuculline solution, but **not systematically** for every cultures as we expected. Because of this we decide to not use solution 1 and solution 2 for more analisys, at this

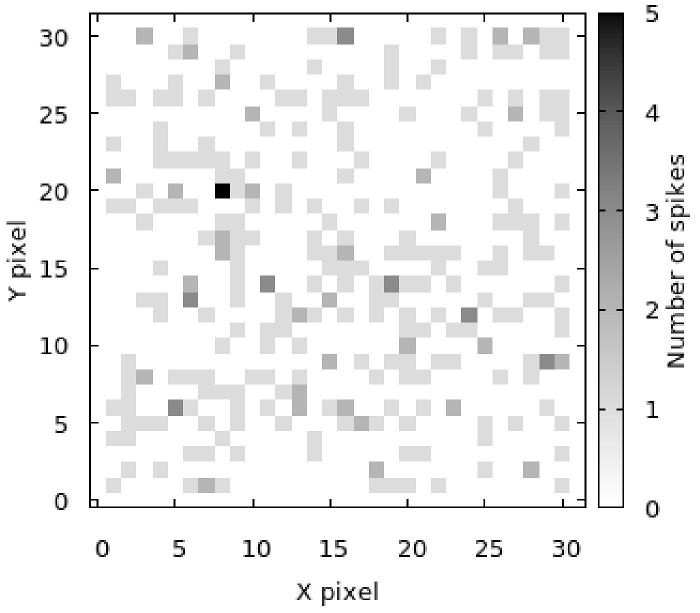


FIGURE 3.11: Activity map of 678 chip over 5 minutes with solution 1.

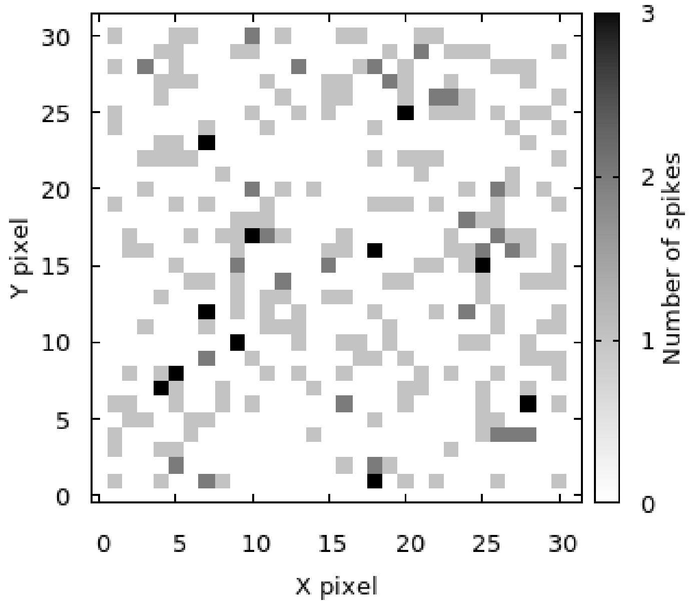


FIGURE 3.12: Activity map of 678 chip over 5 minutes with solution 2.

stage of understanding.

On the other hand with KCl solution the number of spikes detected is significantly higher than the previous cases. The Figure 3.13 shows the moment during which the neurons start to fire. As can be seen the signal coming from a real action potential is significantly higher than signal noise. The convolved signal coming from the correlation algorithm is shown in Figure 3.14.

The Figures 3.15, 3.16 and 3.17 show the activity maps recorded over 4 minutes of three chips.

As can be seen the presence of spike detection is concentrated in specific areas. This result confirm the presence of neurons. In particular it confirms the fact that after the neuron plating the cells are still alive.
The spike detection allows to localize neurons over the 32x32 matrix and to analyze the network activity.

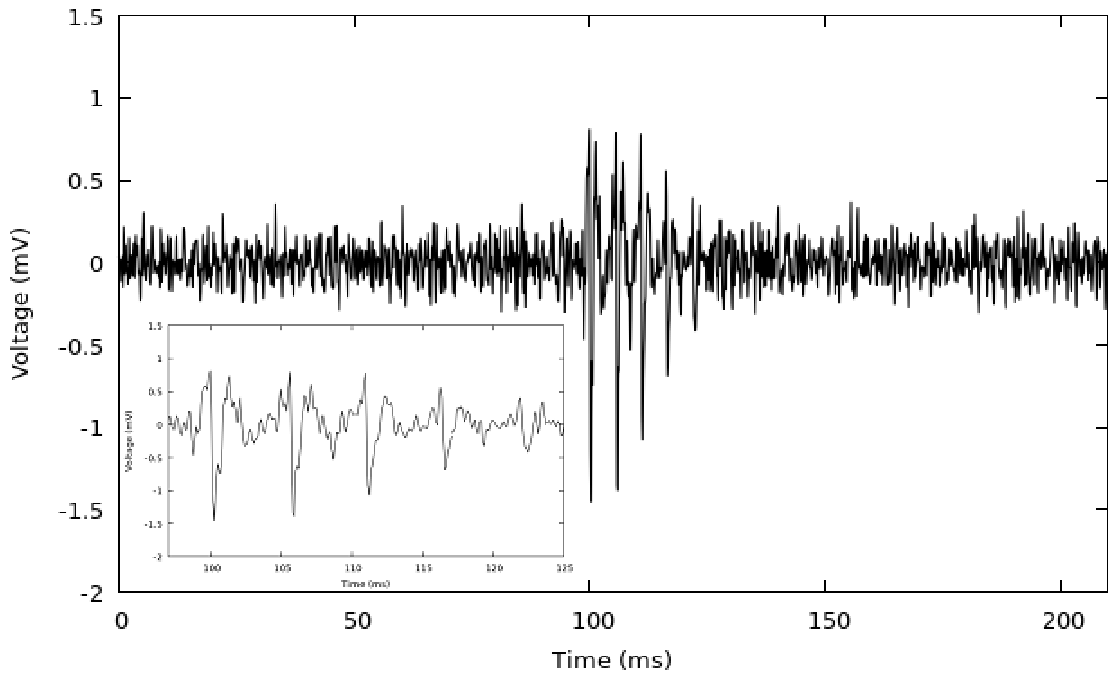


FIGURE 3.13: Signal showing the moment when a neuron starts to fire with the solution 3.

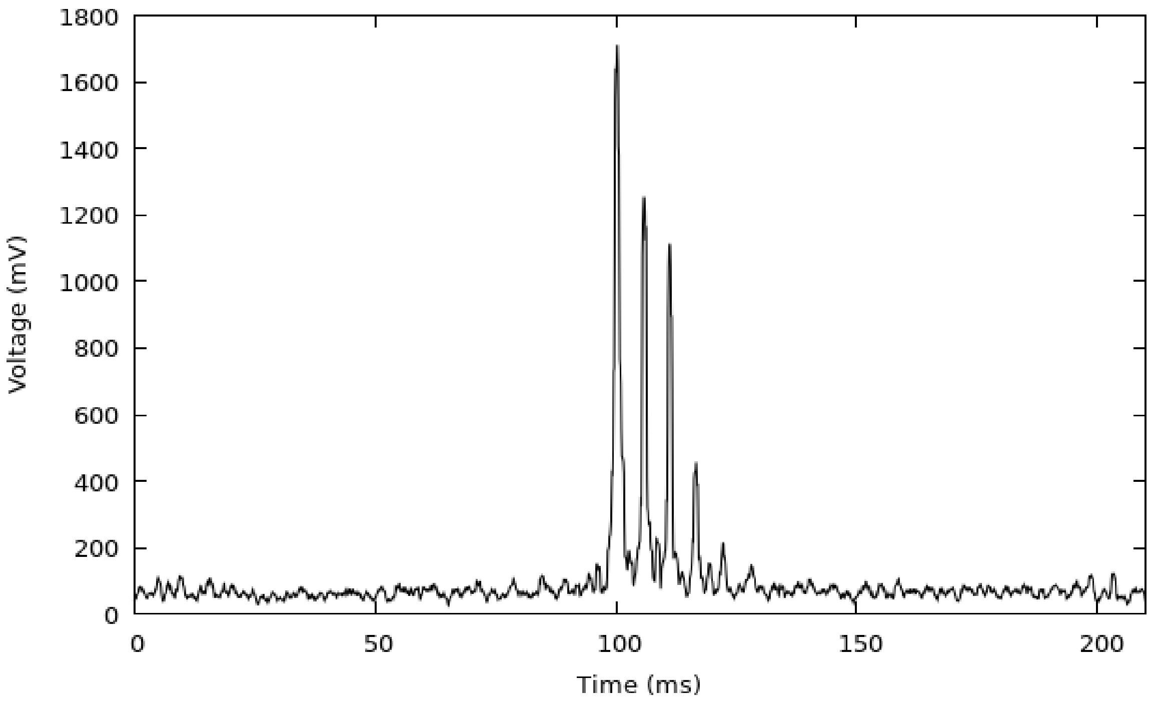


FIGURE 3.14: 9x7 convolved signal from the signal shown in Figure 3.13.

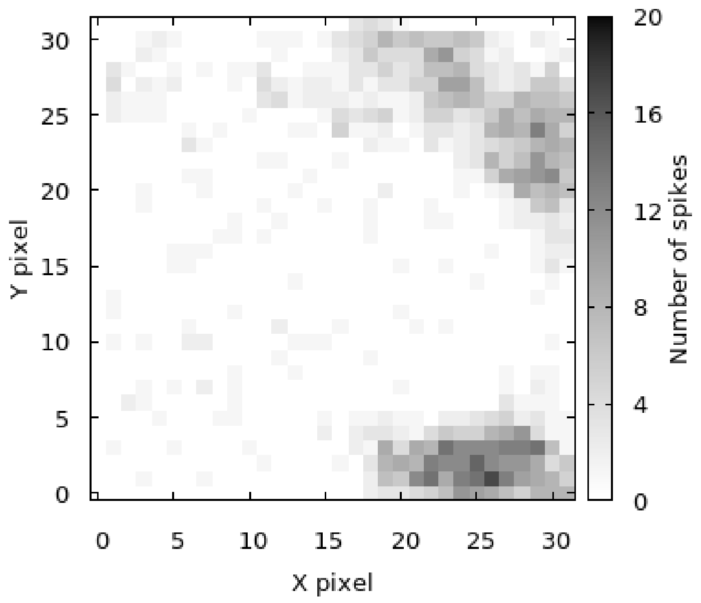


FIGURE 3.15: Activity map from 682 chip over 4 minutes with solution 3.

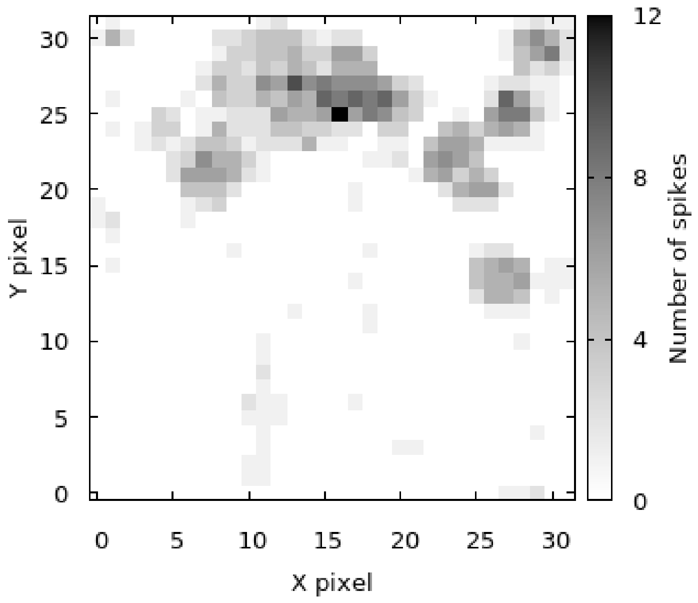


FIGURE 3.16: Activity map from 669 chip over 4 minutes with solution 3.

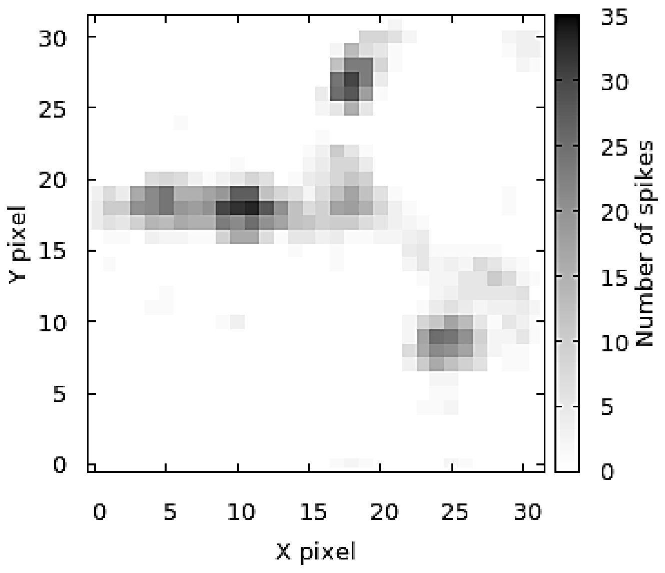


FIGURE 3.17: Activity map from 678 chip over 4 minutes with solution 3.

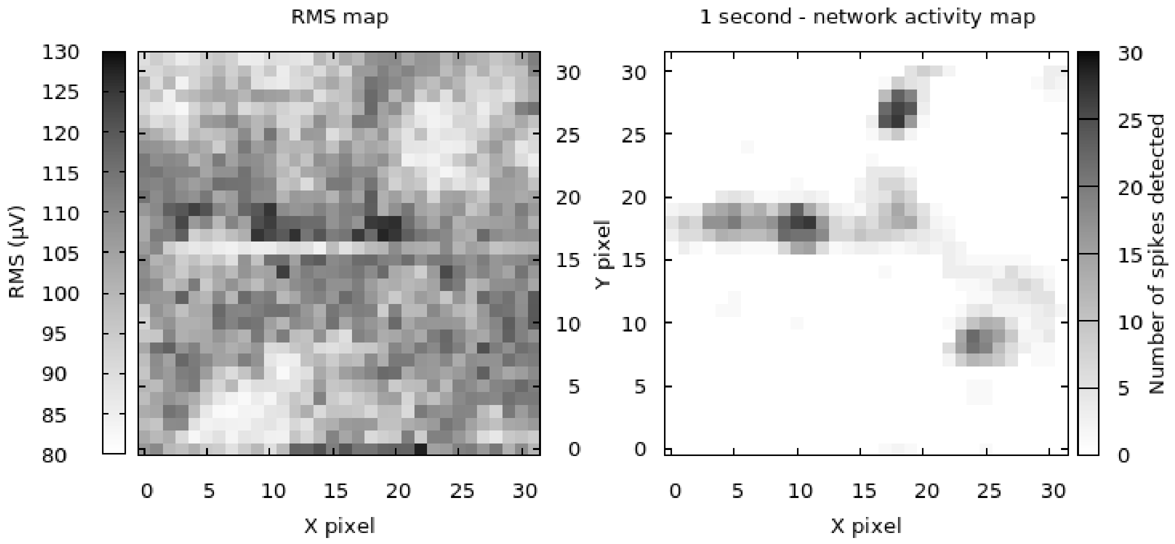


FIGURE 3.18: Comparison between RMS map and activity map.

3.4 Neuronal activity analysis

It is chosen to analyze the track recorded from the 678 chip with the presence of KCl solution because it allows to localize neurons and the number of spikes is higher than every other chip analyzed. The most of spikes are recorded during 1 second after KCl dilution. The neuronal activity on this chip allows to perform a good localization of neurons over the 32x32 matrix, the measurement of the firing frequency of a neuron, to verify the time correlation between different neurons spikes and the inter-neuron conduction velocity.

3.4.1 Neurons localization

It is interesting to compare the neuronal activity map with the rms map. As can be seen from the Figure 3.18 the rms map doesn't allow to localize neurons as well as activity map. Indeed the areas with higher rms is due to glial cells that in most cases surrond neurons. Probably the horizontal area with low rms is where glial cell failed to adhere because the presence of neurons. Neurons have a spherical shape and a little surface to adhere.

The Figure 3.19 shows the activity heatmap and the 3D-graphic, showing the number of spikes detected, from which can be seen the presence of four main peaks.

The neurons are localized in the five areas and numbered as shown in Figure 3.20.

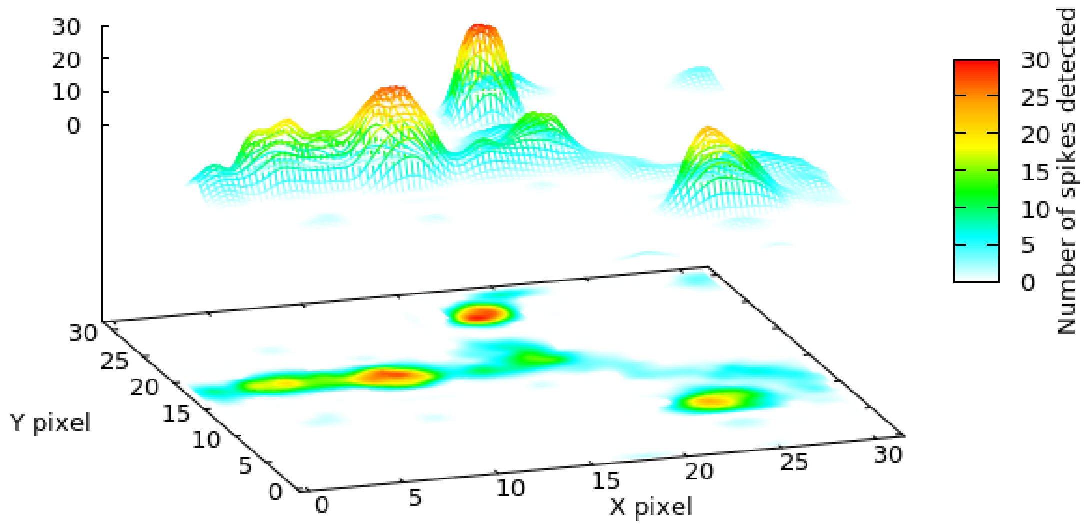


FIGURE 3.19: Heatmap and 3D-graphic of neuronal activity.

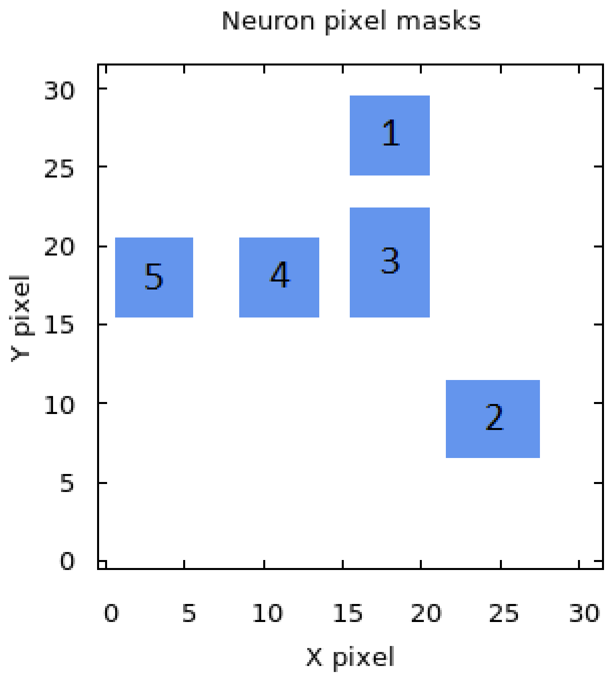


FIGURE 3.20: Numbered masks areas over the five neurons.

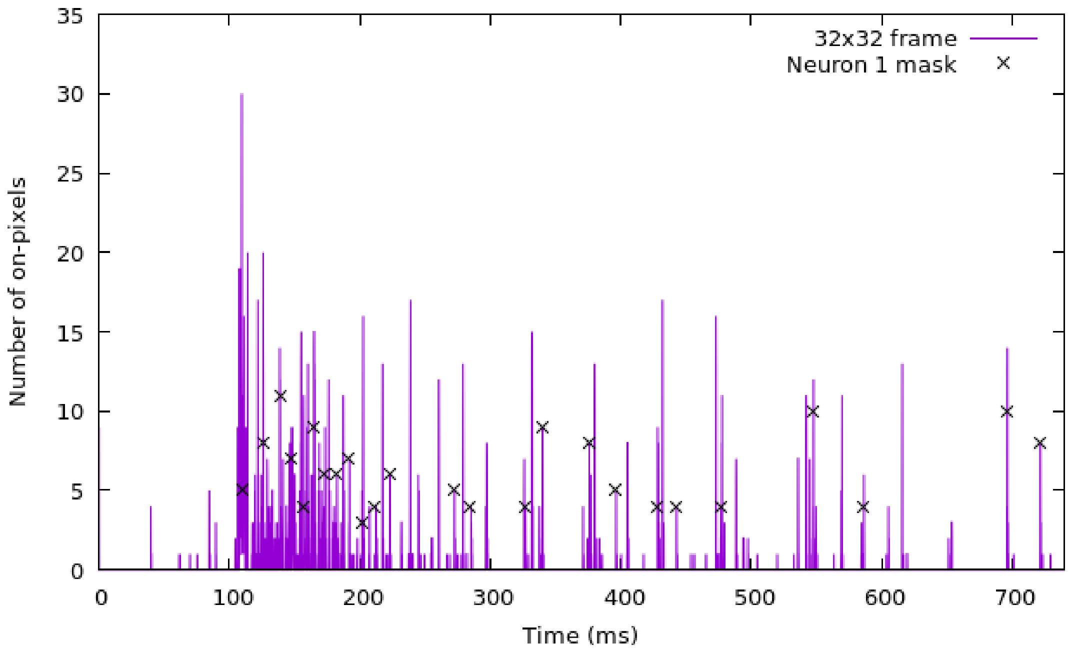


FIGURE 3.21: Full frame neuronal network activity and neuron 1 mask activity. The ordinate value corresponds to the number of on-pixels in a frame.

3.4.2 Firing frequency measurement

The firing frequency of a neuron can be measured from the recorded activity. An index of neuronal network activity can be consider the number of "on-pixels" (pixels that turn on) detected per full frame. The Figure 3.21 shows the presence of an high neuronal network activity after putting the KCl in the solution. It is possible to filter the number of "on-pixels" counting only pixels over one neuron. The masks used to filer the activity are shown Figure 3.20. In Figure 3.21 the activity from mask 1 and the neuronal network activity are superimposed, resulting a systematic activity due to the firing neuron.

In Table 3.1, 3.2 and 3.3 the time value during which the neuron 1 fired are reported. The time zero coincides with the beginning of the time window chosen.
The Figure 3.22 shows the spiking events over the time. It is possible to notice that there are three zone of different spike frequency. To measure the frequency three linear fit are performed with the linear function $f(x) = mx + q$. The value of parameters are reported in Table 3.1, 3.2 and 3.3 and the frequency value in Hz is calculated from m parameter. It is possible to notice that the frequency value decreases over time. This phenomenon can be explained considering that when the KCl is added the neuron membrane is depolarized too much. In this condition the neuron becomes inactive, reducing its spike frequency over the time.

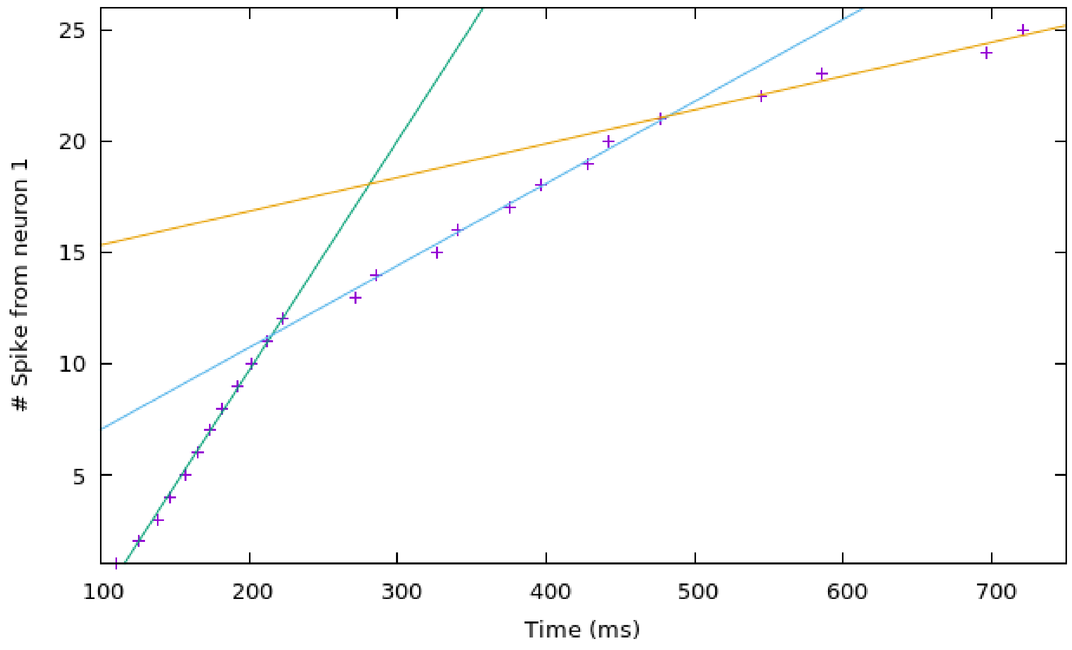


FIGURE 3.22: Spike events over time from neuron 1 mask. Linear fits parameter and frequency are report in Table 3.1, 3.2 and 3.3.

# Firing event	Time (ms)
1	111.680 ± 0.009
2	126.400 ± 0.009
3	139.093 ± 0.009
4	147.947 ± 0.009
5	157.333 ± 0.009
6	165.867 ± 0.009
7	173.867 ± 0.009
8	182.613 ± 0.009
9	192.320 ± 0.009
10	202.133 ± 0.009
11	212.693 ± 0.009
12	223.253 ± 0.009

Fit parameters

m (1/ms)	q
0.103 ± 0.002	-11.0 ± 0.4

Frequency = (103 ± 2) Hz

TABLE 3.1: High frequency data with fit parameters and frequency value.

# Firing event	Time (ms)
12	223.253 ± 0.009
13	272.853 ± 0.009
14	285.547 ± 0.009
15	327.147 ± 0.009
16	340.267 ± 0.009
17	376.107 ± 0.009
18	396.480 ± 0.009
19	428.693 ± 0.009
20	442.987 ± 0.009
21	477.013 ± 0.009

Fit parameters

m (1/ms)	q
0.036 ± 0.001	3.3 ± 0.4

Frequency = (37 ± 1) Hz

TABLE 3.2: Medium frequency data with fit parameters and frequency value.

# Firing event	Time (ms)
21	477.013 ± 0.009
22	545.067 ± 0.009
23	586.067 ± 0.009
24	696.533 ± 0.009
25	721.707 ± 0.009

Fit parameters

m (1/ms)	q
0.015 ± 0.001	13.8 ± 0.9

Frequency = (15 ± 2) Hz

TABLE 3.3: Low frequency data with fit parameters and frequency value.

3.4.3 Time correlation between different neurons spikes

The neurons growing form connection among them. Because of this when a neuron fires it can induce a near neuron to fire too. The Figures 3.23, 3.24, 3.25 and 3.26 show the overlap between activity of near neurons, as the number of "on-pixels" (number of pixels that turn on in the selected neuron mask during a spike). It is possible to notice that there are many spikes colse in time between neurons (1, 3) and (4, 5), less between (3, 4) and (3, 2). In this way it is possible to find sequential events to measure signal transmission velocity, as shown in the next section.

3.4.4 Inter-neuron conduction velocity

From different neurons activities overlap it is possible to find some cases during which more than two neurons communicate. As shown in Figure 3.27, a sequence of frame during which occurs the signal transmission between four neurons is found. This sequence of frame is used to estimate the inter-neuron conduction velocity.
To measure the position of the signal we choose one pixel per frame. The frames during which occurs the sequential events are reported in Figures 3.28, 3.29, 3.30, 3.31, 3.32, 3.33, 3.34 and 3.35. The red pixels used to measure the signal position is the first along the direction of signal propagation. Frame 6 and 8 are discarded because the head pixels were outliers. Probably in that frames the signal was traveling throgh axons without been recorded.
To measure the time of spike detection on those pixels we considere that a whole column signal is recored in the same time with a frequency of 30000 Hz = 32 x 9375 Hz. So the time is calculated with the following formula:

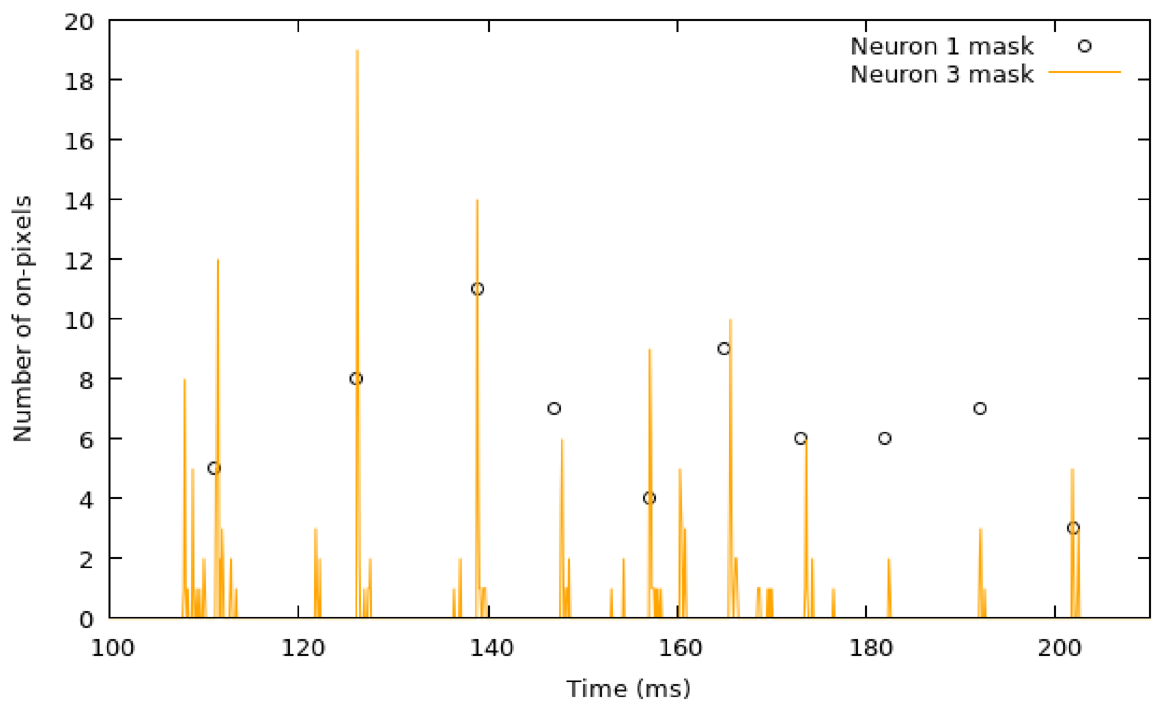


FIGURE 3.23: Overlap between activity of neuron 1 mask and neuron 3 mask.

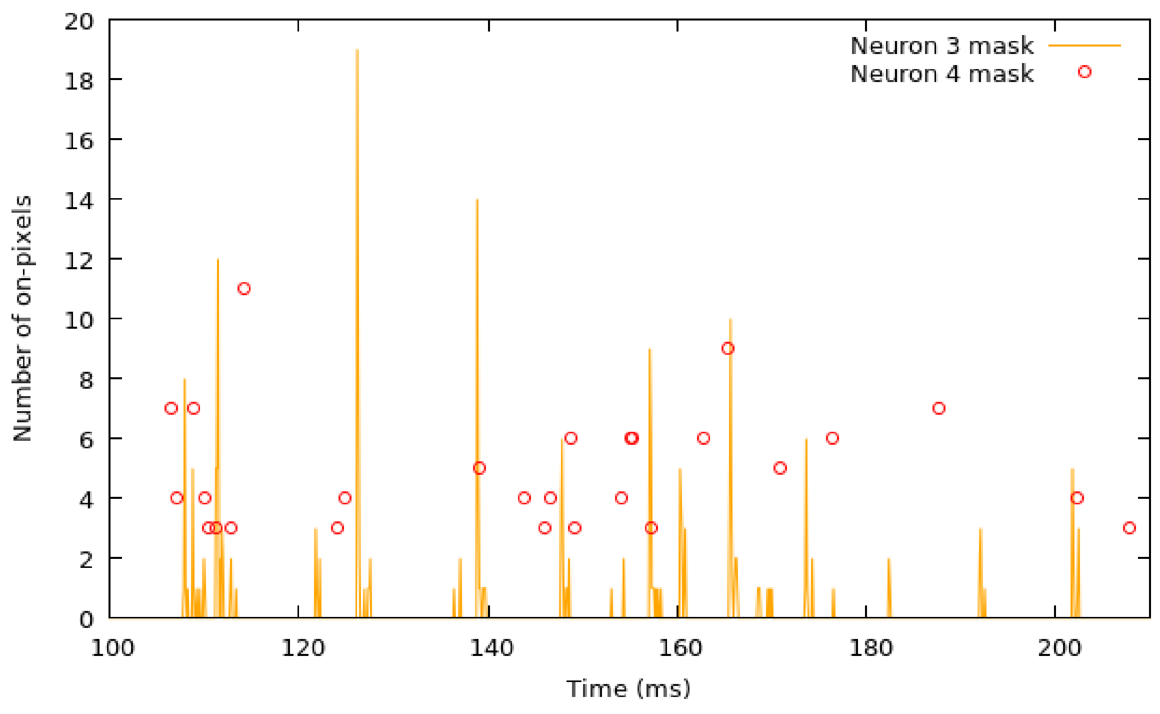


FIGURE 3.24: Overlap between activity of neuron 3 mask and neuron 4 mask.

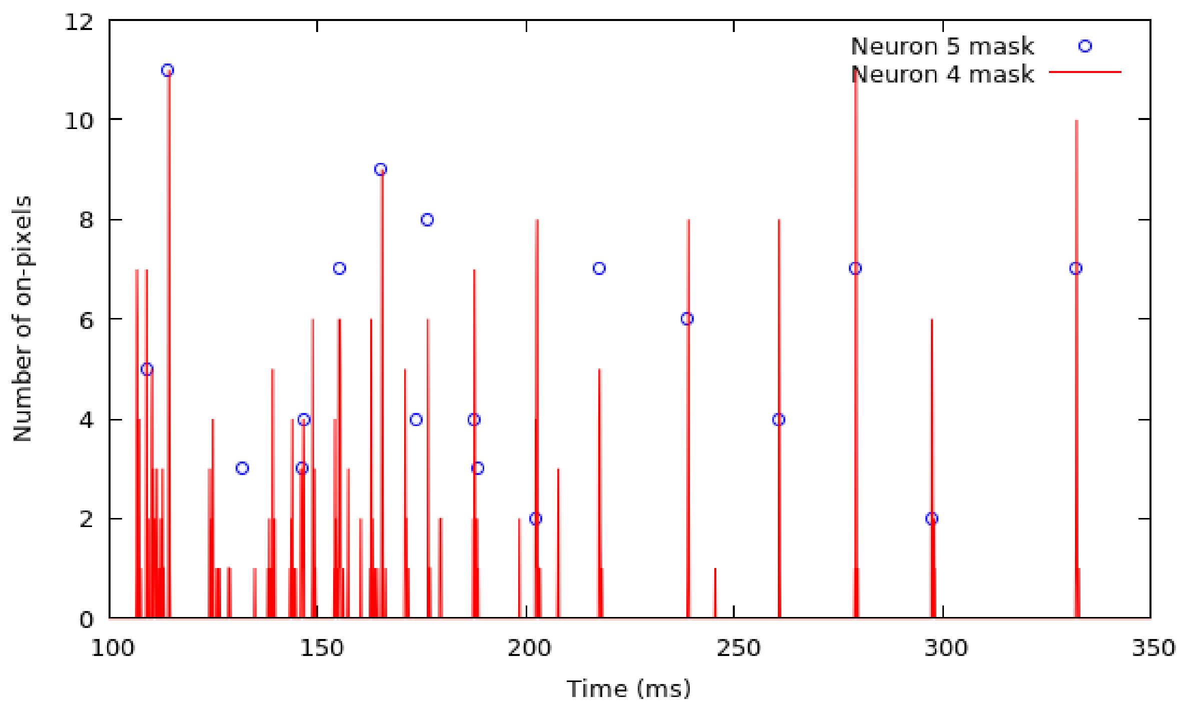


FIGURE 3.25: Overlap between activity of neuron 4 mask and neuron 5 mask.

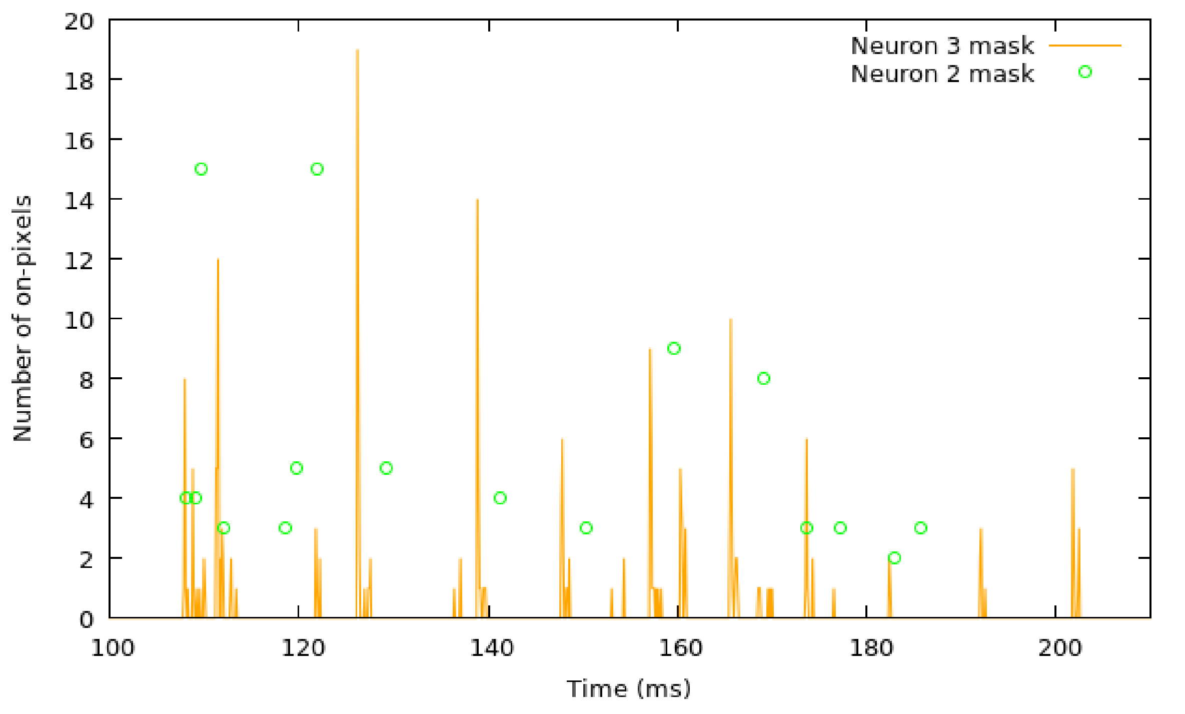


FIGURE 3.26: Overlap between activity of neuron 3 mask and neuron 2 mask.

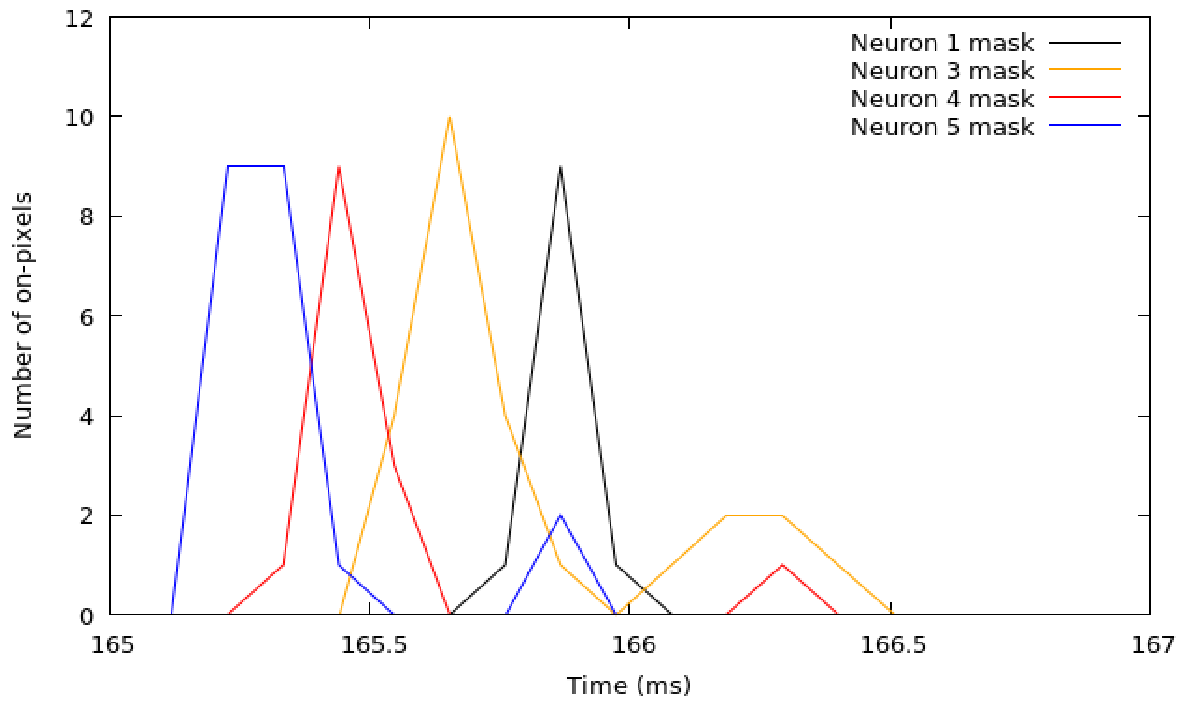


FIGURE 3.27: Sequential spikes between 4 different neurons (5,4,3,1).

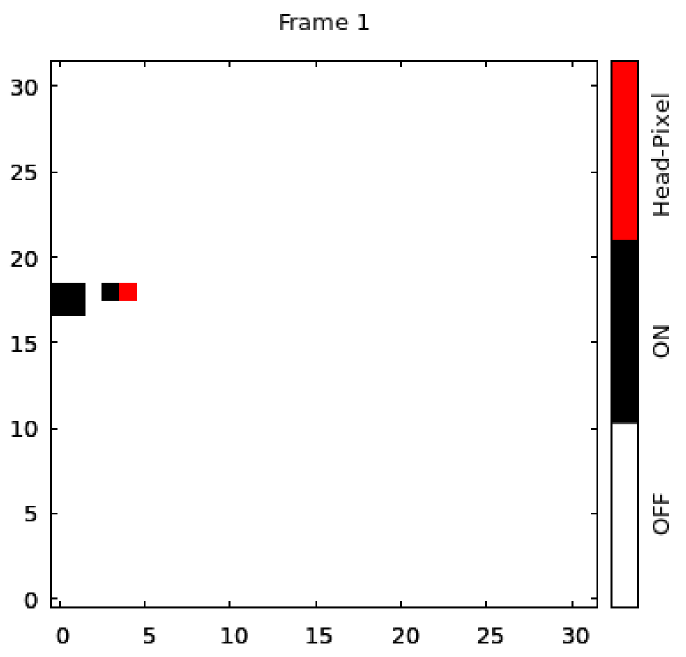


FIGURE 3.28: Frame 1 of signal transmission. The red pixel is the head of signal propagation.

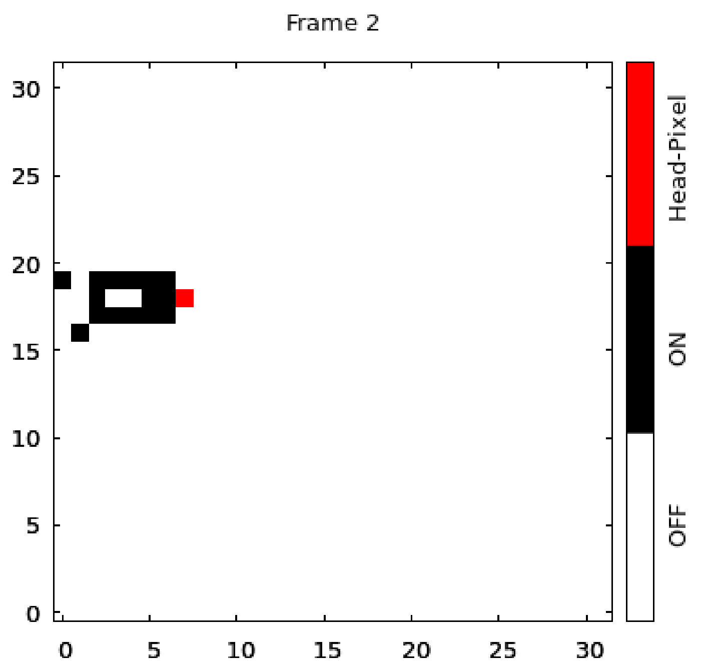


FIGURE 3.29: Frame 2 of signal transmission. The red pixel is the head of signal propagation.

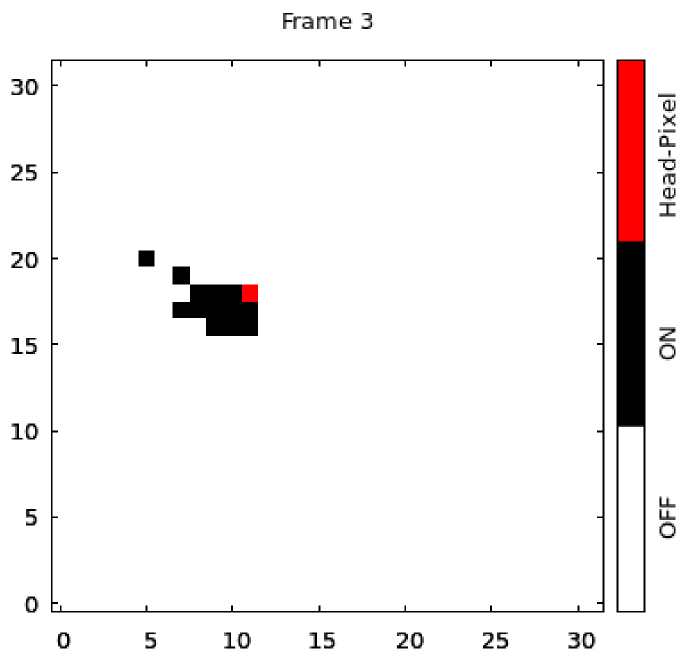


FIGURE 3.30: Frame 3 of signal transmission. The red pixel is the head of signal propagation.

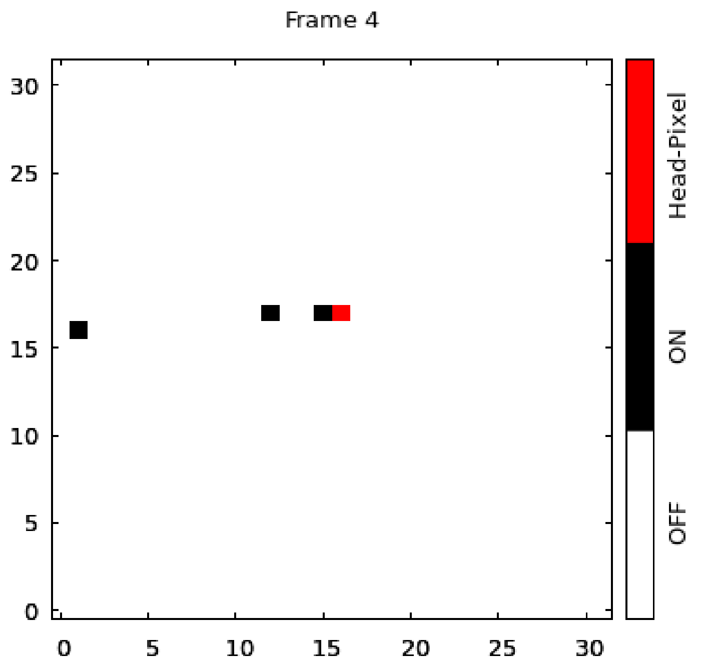


FIGURE 3.31: Frame 4 of signal transmission. The red pixel is the head of signal propagation.

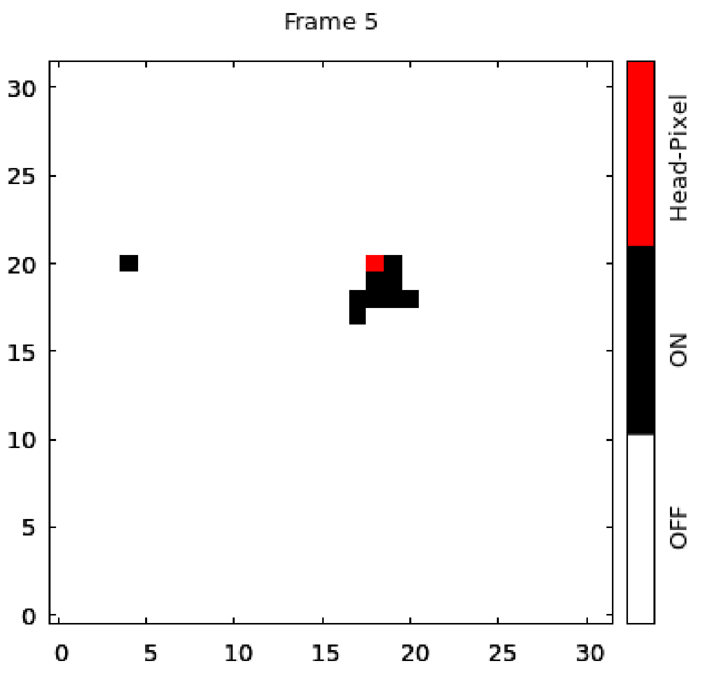


FIGURE 3.32: Frame 5 of signal transmission. The red pixel is the head of signal propagation.

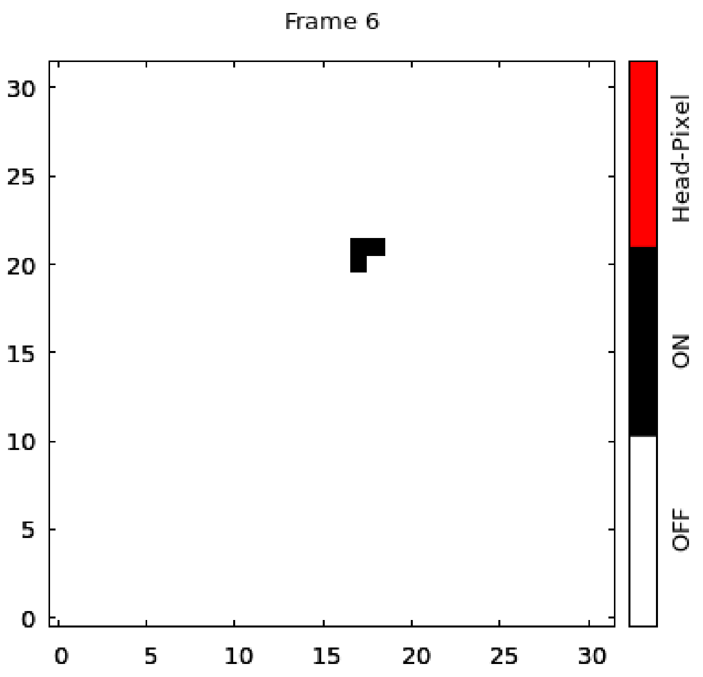


FIGURE 3.33: Frame 6 of signal transmission. No head pixel is recorded.

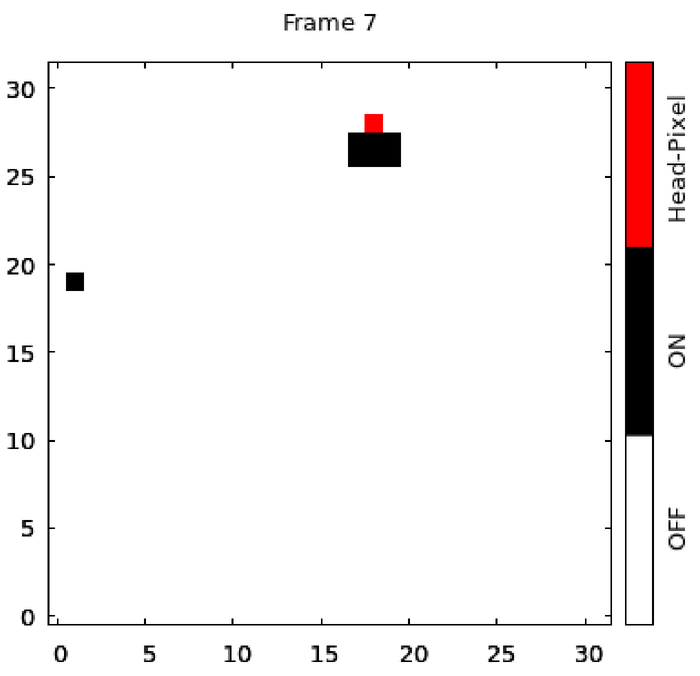


FIGURE 3.34: Frame 7 of signal transmission. The red pixel is the head of signal propagation.

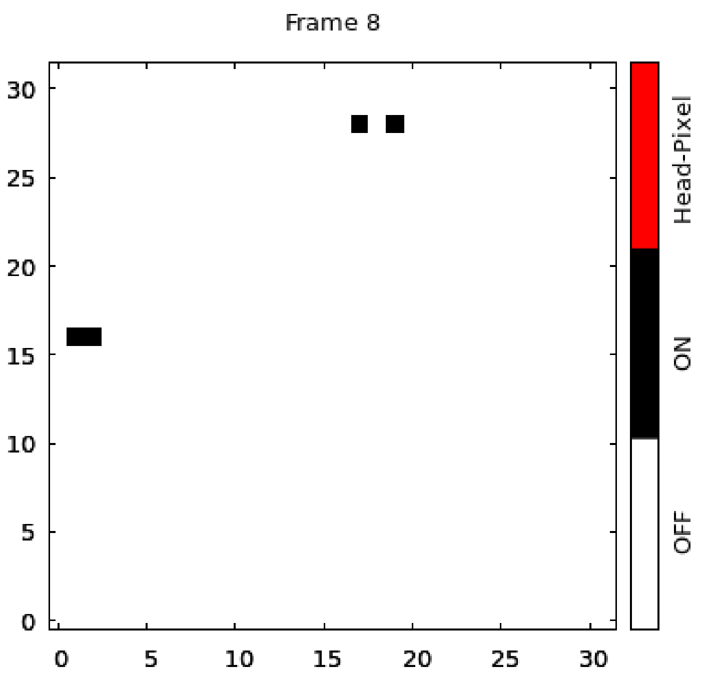


FIGURE 3.35: Frame 8 of signal transmission. No head pixel is recorded.

$$t_{N,y} = \frac{32 \times (N-1) + y}{f} \tag{3.2}$$

where N is the frame number, starting from $N = 1$; $y \in [0, 31]$ is the column coordinate of the pixel considered; $f = 30000$ Hz is the column sampling frequency.
The error gave to the time measurement is the uniform distribution error calculated as the time intervall between two measures divided by $\sqrt{12}$.
The Figure 3.36 shows the hexagonal grid and the pixels used to calculate space intervals. The number on a pixel is the value of N. To calculate the distance between the center of near red pixels it is used the value of column pitch ($5.625\,\mu m$) and row pitch ($6.5\,\mu m$). The first red pixel of signal propagation is put at position zero. The space covered by the signal is calculated as the sum of distances between near red pixels.
The Figure 3.37 shows the graphic of space covered over time. It is performed a linear fit with the function $y = mx + q$. The space error is calculated with ordinate deviations e_i from linear fit as:

$$\sigma_s = \sqrt{\frac{\sum_{i=1}^{n} e_i^2}{n-2}} \tag{3.3}$$

$$e_i = y_i - (mx_i + q) \tag{3.4}$$

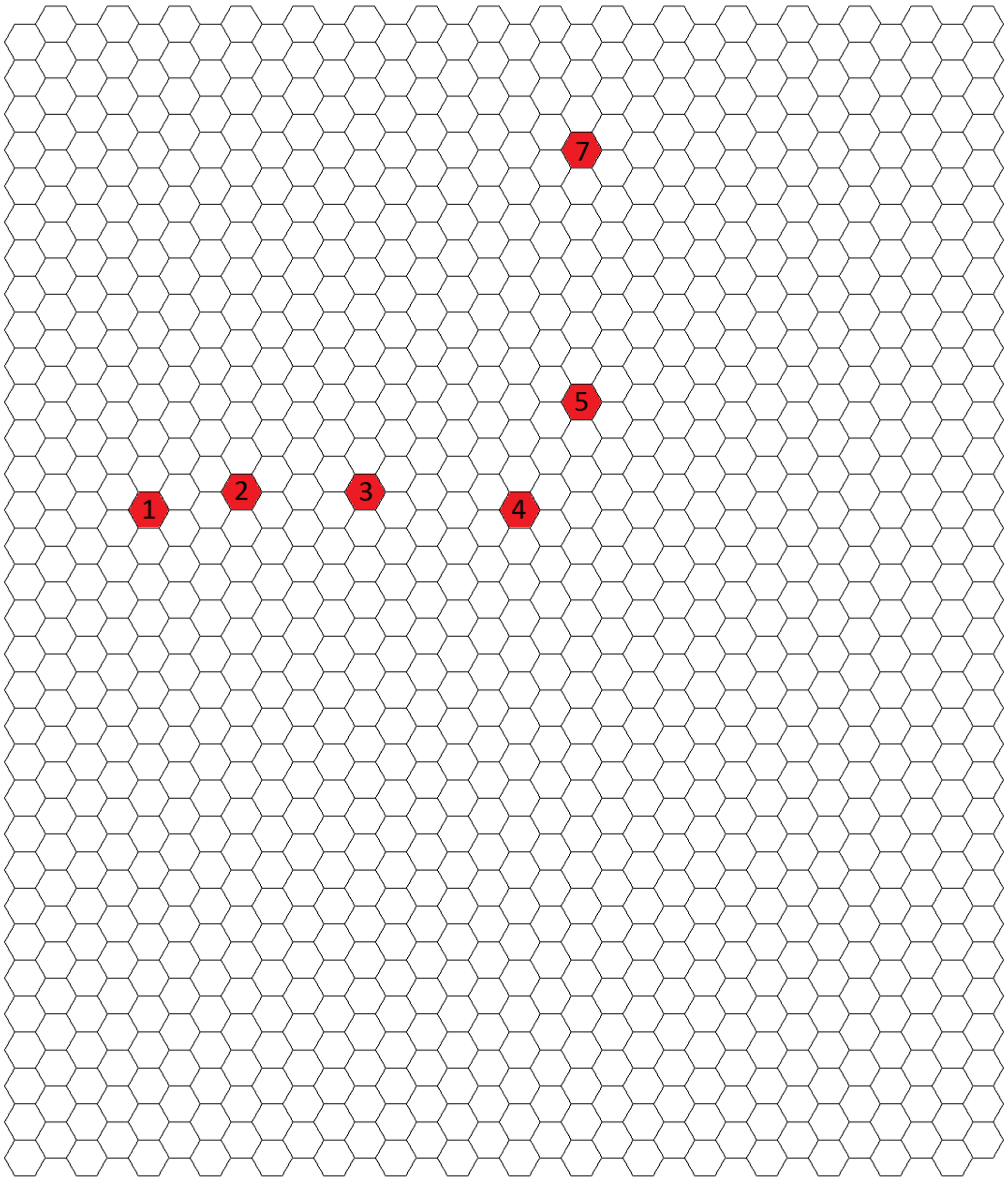


FIGURE 3.36: 32x32 hexagonal grid. The red pixels are the pixels where the head of signal propagation is recorded. The number on the pixel is the value of the frame during which the signal is detected.

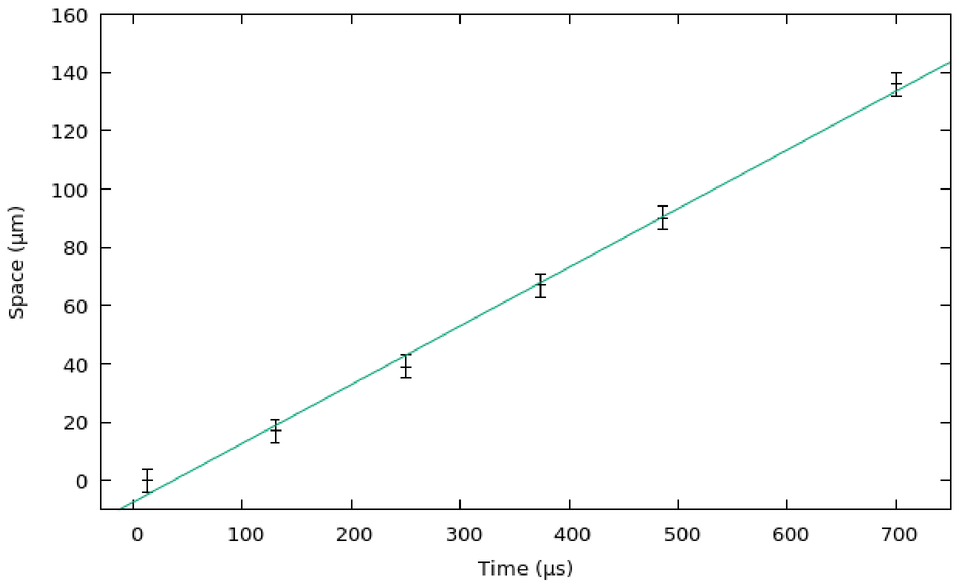


FIGURE 3.37: Space covered by signal between neurons over time. It is performed a linear fit and from parameters the conduction velocity is extimated (Table 3.4).

In Table 3.4 are reported the time and space values and fit parameters.

# Frame	Time (μs)	Space (μm)
1	13 ± 1	0 ± 3
2	130 ± 1	17 ± 3
3	250 ± 1	40 ± 3
4	373 ± 1	68 ± 3
5	487 ± 1	91 ± 3
7	700 ± 1	136 ± 3

Fit parameters

m (m/s)	q (μm)	Velocity (cm/s)
0.201 ± 0.006	-7 ± 2	20.1 ± 0.6

TABLE 3.4: Time and space values of signal propagation throgh neurons. The parameters of linear fit and the velocity extimation are reported.

The time takes from the signal to propagate from the first neuron to the fourth in about $0.7\,ms$. So the time scale of transmission is compatible with the trasmition synaptic delay of about 0.3 ms. In future can be possible to perform a measurement of neuronal signals transmission timing to confirm this result.

Chapter 4

Real-time data acquisition chain

Much time has been devoted to set up an acquisition chain to real-time process the streaming data coming from the CAN-Q, through a FPGA card. In this chapter the choices made to set up the acquisition chain are explained. It focuses on the digital implementation of correlation algorithm on FPGA for action potential detection and on the graphical user interface. The Figure 4.1 shows the acquisition chain scheme: the single parts of it are described in the following sections.

4.1 Zedboard

The Zedboard [19] is an evaluation and development board based on the Xilinx Zynq-7000 [20] that integrates the software programmability of an Dual-Core ARM-based processor with the hardware programmability of an FPGA (Artix-7). On the board there is a 100 MHz oscillator. The Figure 4.2 shows the Zynq Z7020 Bank Assignments.
The CAN-Q has only an ethernet output port. So the Zedboard must receive the data through its ethernet input port. As can be seen from the Bank Assignment scheme, the ethernet port is connected to the Processing System (PS) and not directly with the Programmable Logic (PL). Because of this it was decided to implement the Xillinux OS on the Processing System. Thanks to this embedded OS it is possible to send data to FPGA of the Programmable Logic side.

4.1.1 Vivado Design Suite

To implement the application logic on Zedboard FPGA it is used the Vivado software produced by Xilinx. It is a design enviroment used for digital circuit disign, using the programming language VHDL and Verilog. Thanks to this software a bitstream file can be generated: it contains the informations to implement the digital circuit on the FPGA.

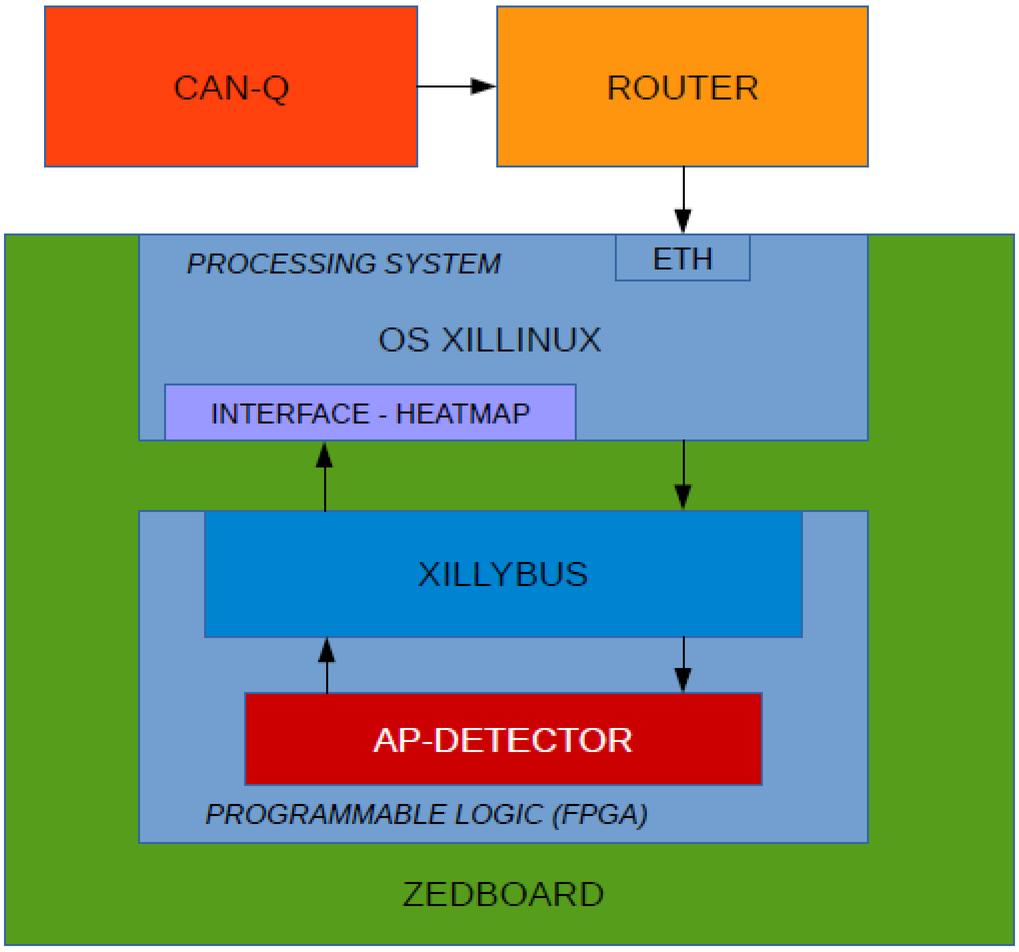


FIGURE 4.1: Acquisition chain to analyze data coming from the CAN-Q in real time. The CAN-Q is connetted to a router from which data are sent to a Zedboard through the ethernet port. Xillinux, an embedded OS, drives data to an FPGA on which it is implemented the correlation algorithm. The output is sent to a graphical user interface to show real-time the neuronal activity map.

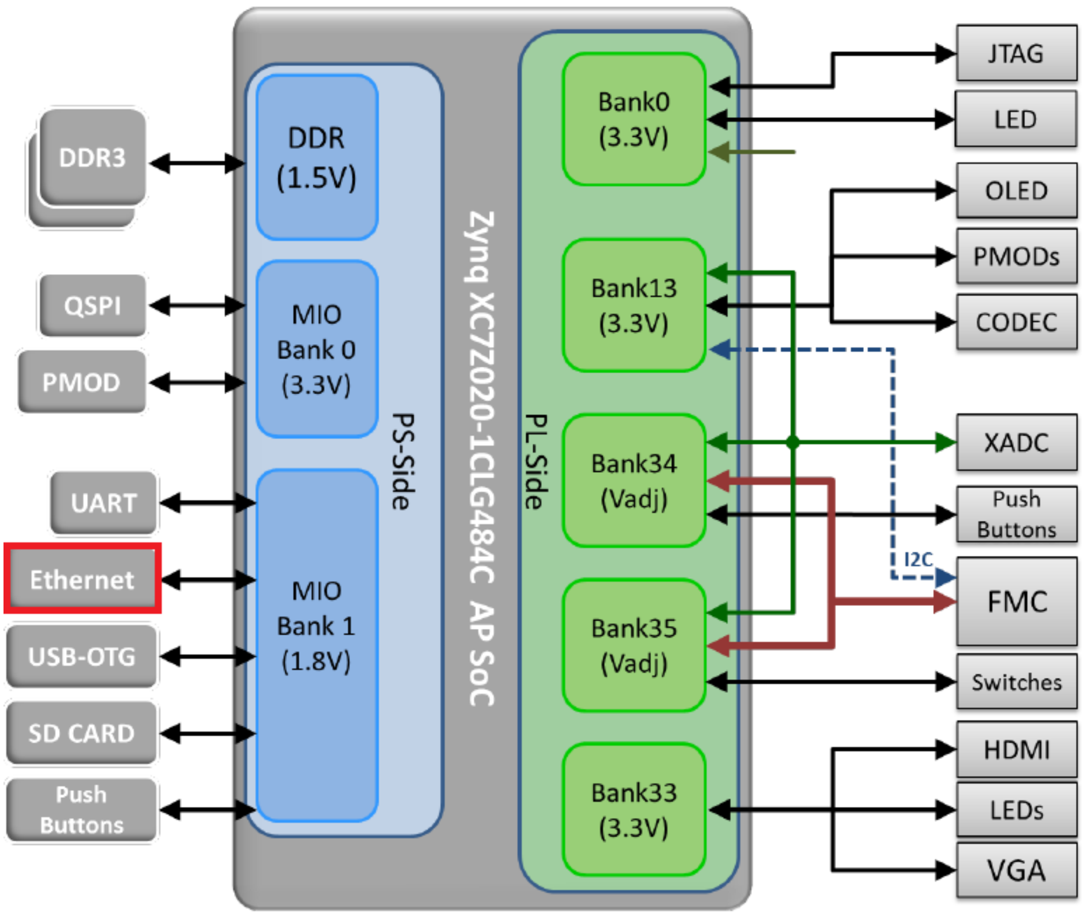


FIGURE 4.2: Zynq Z7020 Bank Assignments [19]. The ethernet port is connected to the Processing System side.

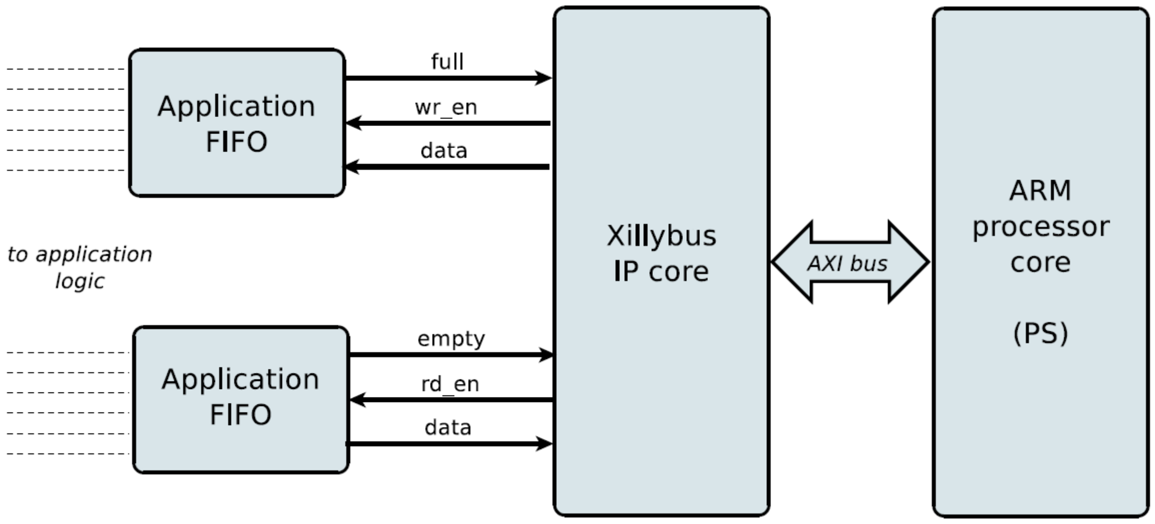


FIGURE 4.3: Block scheme of communication between ARM processor and the Programmable Logic [12].

4.2 Xillinux

The Xillinux [18] distribution is a software + FPGA code kit for running a graphical desktop on the Zedboard attaching a monitor, keyboard and mouse to the board itself. The useful feature is the possibility to make integration between the Linux host (PS) and the FPGA (PL). As a matter of fact there are four device ports that virtualize the input and the output of two FPGA FIFOs (First In First Out), one of 32-bit and one of 8-bit. Xillinux is based upon Ubuntu LTS 12.04 for ARM. It makes the board behave like a PC with the SD card as its hard disk.

4.2.1 The Xillybus IP core

The communication between the ARM processor and the Programmable Logic side takes place through the Xillybus IP (Intellectual Property) core. The Xillybus IP core communicates with the ARM processor through the AMBA (Advanced Microcontroller Bus Architecture) bus protocol. On the other side it communicate with application logic through FIFOs.
To implement the correlation algorithm for action potential detection just need to add the application logic between writing-FIFO and reading-FIFO, as shown in Figure 4.3.

To generate the Xillybus IP core it is necessary to load the `xillydemo-vivado.tcl` script into Vivado software. Then it is possible to generate the corrispondent bitstream file: `xillydemo.bit`.
In Linux OS it is possible to find the files that virtualized FIFOs in devices folder: `\dev\xillydemo_32bit_write` and `\dev\xillydemo_32bit_read`;
`\dev\xillydemo_8bit_write` and `\dev\xillydemo_8bit_read`.

4.2.2 Building Xillinux on SD card

To boot the Xillinux distribution from the SD card (2 GB or more), it must have two components:

- a FAT32 filesystem in a boot partition (16 MB), consisting of boot loaders, the configuration bitstream for FPGA, and the binaries for booting the Linux kernel;
- an ext4 root file system partition (the remaining memory).

Xillinux provides the Xillinux SD image `xillinux-1.3.img`. The image contains a partition table, a partly populated FAT file system for placing initial boot files, and the Linux root file system of ext4 type. To load the Xillinux SD image on the SD card it was used Win32 Disk Imager, generating the uImage file in the boot partition and the Linux root file system in the second partition.
In order to boot, four files need to exist in the SD card's first boot partition:

- `uImage` – The Linux kernel binary. This is the file in the boot partition after writing the Xillinux SD image to the card. The kernel is board-independent;
- `boot.bin` – The initial bootloader, provided by Xillinux. This file contains the initial processor initializations and the U-boot utility, and it is significantly different from board to board;
- `devicetree.dtb` – The Device Tree Blob file, provided by Xillinux, which contains hardware information for the Linux kernel;
- `xillydemo.bit` – The bitstream file for the FPGA.

4.3 Implementation of action potential detection algorithm on FPGA

The action potential detection algorithm was designed in VHDL in collaboration with E. A. Vallicelli (Bicocca in Milan) [3]. In this section the block scheme and the implementation of the algorithm on the FPGA are explained.

4.3.1 The Action Potential detector block scheme

The idea is to implement a digital circuit that, receiving the data in input, computes the condition

$$\sum_{i=1}^{9\times7} \frac{V_i^2}{\sigma_i^2} > (13.6)^2 \tag{4.1}$$

returning the boolean value 1 or 0 if the condition 4.1 is true or false respectively.

In order to do that, these steps are followed:

- calculate the σ^2 map of the 32x32 matrix;
- sum over nine squared points consecutive in time of the same pixel;
- divide the result by the σ^2 of the respective pixel;
- sum over seven near pixel;
- compare the final value with the threshold and return a boolean value.

The action potential detector block scheme is shown in Figure 4.4, between the writing and reading FIFOs.

4.3.2 Pipeline and triggers

The digital algorithm works with a maximum input and output data rate of 100 MHz. Using the "empty" signal of the first FIFO the input data rate is reduced down to the streamed data rate of 9.6 MHz. A time delay of 4 clocks is introduced: one by the MEM-T block, one by the DIV block, one by the MEM-S block and one by the C block (Figure 4.4). The reduced time delay of DIV block is assured because "the core is highly pipelined: the throughput of the core is configurable and can be reduced from 1 clock cycle per division to 2, 4, or 8 clock cycles per division to reduce resources" [10].
The most important triggers are two: the "newdata" trigger from F1 flagging to MEM-T and $\sigma^2 - CALC$ when a new data is available; the "division" trigger from DIV flagging to MEM-S and F2 when the last division is performed.

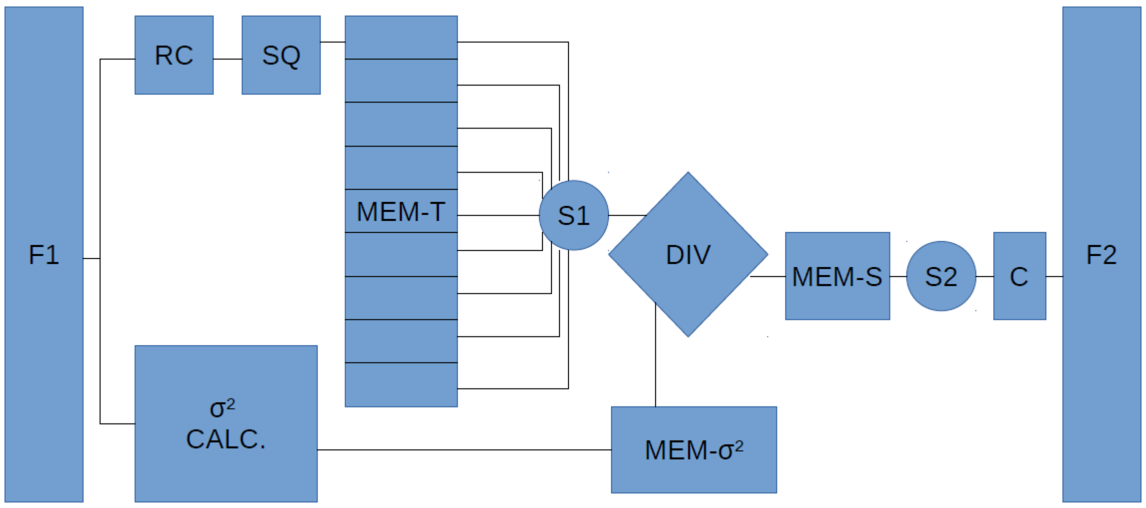


FIGURE 4.4: Action potential detection algorithm scheme:

F1 and F2: writing FIFO and reading FIFO
RC*: recoder to reduce data size;
SQ: it squares the data with a 32 bit output;
MEM-T: nine 32x32 memories of 32 bit; S1: it implements the sum of 9 consecutive data coming from the same pixel (time sum);
DIV: diveder to implement the division by σ^2 for the corrispondent pixel;
MEM-S: 32x32 memory of 32 bit;
S2: to implement the sum of data in 7 near pixels (space sum);
C: the output of the comparator is 1 or 0 if the result coming from S2 is higher or lower the threshold;
σ^2 CALC. : black box for σ^2 calculation from initial coming data;
MEM-σ^2 : 32x32 memory storing σ^2 map.

* RC blocks could be put in different parts of the algorithm to optimize memory utilization. At this alpha-test of algorithm implementation they are not really used.

4.3.3 Squared RMS map calculator and memory

The formula to calculate the σ^2 is

$$\sigma^2 = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N-1} \tag{4.2}$$

but in this form it is not very useful because of the calculation of $\bar{x}$ that needs a division module. It is better to implement less divisions since they take longer than simpler operations.
The formula can be rewritten in a better form:

$$\begin{aligned}
\sum_{i=1}^{N} (x_i - \bar{x})^2 &= \sum_{i=1}^{N} \left(x_i^2 - 2x_i\bar{x} + \bar{x}^2\right) && (4.3)\\
&= \left(\sum_{i=1}^{N} x_i^2\right) - 2\bar{x}\left(\sum_{i=1}^{N} x_i\right) + N\bar{x}^2 && (4.4)\\
&= \left(\sum_{i=1}^{N} x_i^2\right) - 2\bar{x}\left(N\bar{x}\right) + N\bar{x}^2 && (4.5)\\
&= \left(\sum_{i=1}^{N} x_i^2\right) - 2N\bar{x}^2 + N\bar{x}^2 && (4.6)\\
&= \left(\sum_{i=1}^{N} x_i^2\right) - N\bar{x}^2 && (4.7)\\
&= \left(\sum_{i=1}^{N} x_i^2\right) - N\left(\frac{\sum_{i=1}^{N} x_i}{N}\right)^2 && (4.8)\\
&= \left(\sum_{i=1}^{N} x_i^2\right) - \frac{\left(\sum_{i=1}^{N} x_i\right)^2}{N} && (4.9)\\
&= \frac{N\left(\sum_{i=1}^{N} x_i^2\right) - \left(\sum_{i=1}^{N} x_i\right)^2}{N} && (4.10)
\end{aligned}$$

$$\sigma^2 = \frac{N\left(\sum_{i=1}^{N} x_i^2\right) - \left(\sum_{i=1}^{N} x_i\right)^2}{N\left(N-1\right)} \tag{4.11}$$

To implement Eq. 4.11 it is necessary to use two 32x32 memories: one to store progressively the sum of data, one to store progressively the sum of squared data. After storing N frames it is implemented a subtractor between the second memory value multipied by N and the squared value of first memory. Finally a module implements the division by $N\left(N-1\right)$. The outup is sent to the 32x32 memory storing the final value of σ^2.

4.3.4 Summation over 9 time frames

First of all, the data needs to be squared. So it is implemented a module that riceve in input the serial data and return serially the squared value of them.
Then it is implemented the summation over 9 consecutive squared data of the same pixel. Sampling the data stream in a matrix of 32x32, the consecutive recorded data from the same pixel are separated by 1024 data. Because of this 9 32x32 memories are used to store 9 consecutive time frames. An iteretor loops over the full memories to fill them. And an iterator loop over the 32x32 memories to return the value stored from the same pixel for each memory. In this way a summer takes as input the same position element from the 9 memories and returns the sum.

4.3.5 Divider

The divider is generated by the IP catalog in the Vivado software: Divider Generator [10]. This block takes as input the dividend and the divisor and return the quotient and the reminder. The algorithm used by the divider is the Radix-2 algorithm.

The Radix-2 algorithm it is a methon to compute the quotient of a division, based on a standard recurrence equation:

$$R_{j+1} = 2 \times R_j - q_{n-1-j} \times 2^n \times D \qquad 0 \leq j \leq n-1$$

$$R_0 = N \ , \quad R_n = 2^n \times R$$

- N is the numerator with k digits;
- D is the divisor with h digits;
- R is the last remainder;
- $n = k - h$;
- q_{n-1-j} is the bit value of the quotient at the digit position $n - 1 - j$.

The Divider takes as input dividend the sum of nine time frames and as input divisor the σ^2 memory. The two inputs are synchronized on the same matrix pixel. The output of the divider gives the quotient and the remainder of the division. At this time the remainder is discarded, but it could be used in future algorithm optimization.

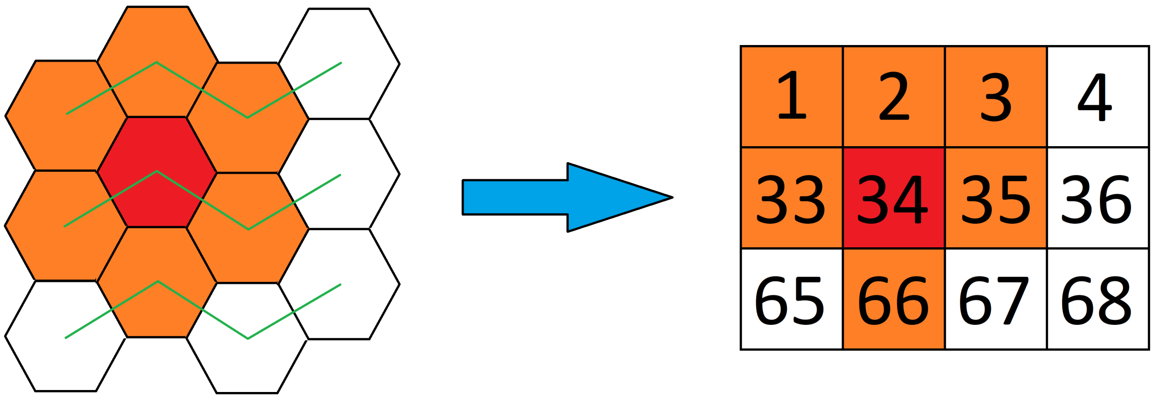


FIGURE 4.5: Grid reshaping. The numbers indicate the nth streamed value of the 32x32 frame. The first central pixel to perform the space convolution is the 34th, only when the 66th is available.

4.3.6 Summation over 7 near pixels and comparator

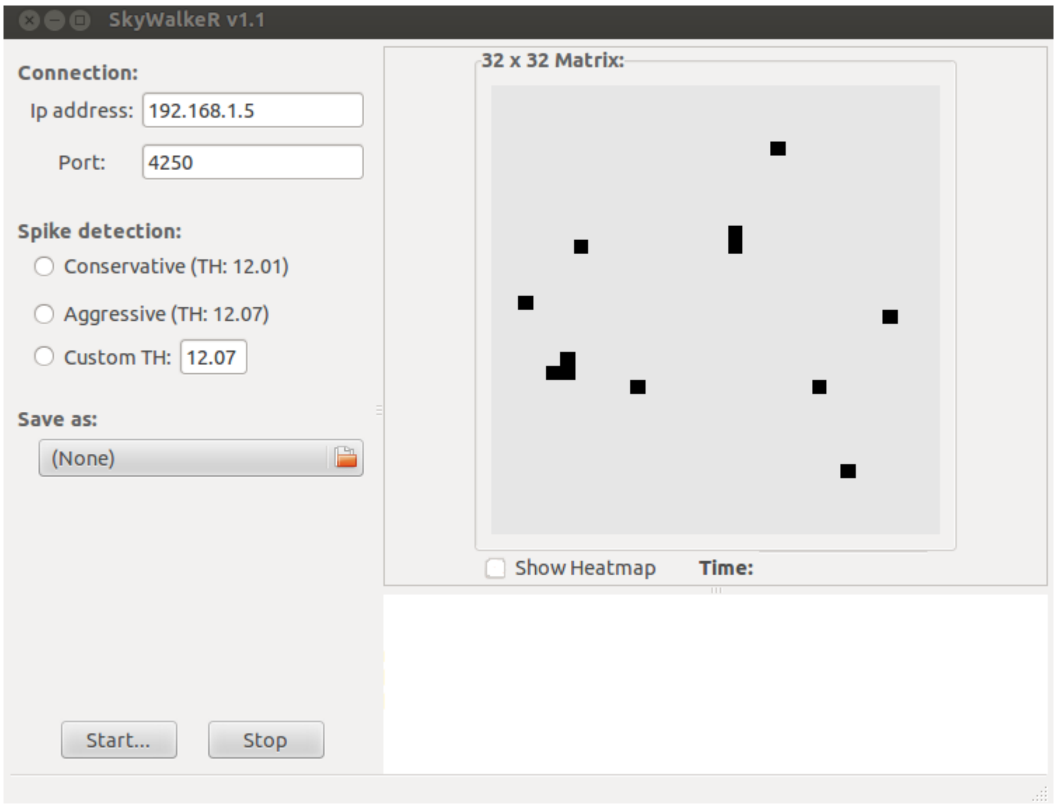
The quotient of the divider is the input of a 32x32 memory.
As can be seen in Figure 4.5 the 7 near pixel to sum are selected taking into account which are the rows and columns of an hexagonal grid. A very important consideration is that only the 30x30 central submatrix has pixels with 7 neighboring pixels. Because of this it isn't returned the convolution value of boundary pixels.
According to the 7 pixel position, the first summation of 34th pixel occurs only when the 66th data of a 1024-frame is available, as shown in Figure 4.5.

The last module is the comparator. It takes as input the summation over 7 near pixels and the output is 1 if the value is greater than a threshold, else it is 0.

4.4 C-program for data streaming

A program written in C language is developed to allow the streaming of data from the ethernet port of the Zeadboard to the input 32-bit FIFO of the FPGA. Furthermore the same program allows the reading of data from the output 32-bit FIFO of the FPGA. In the program two main threads work:

- the first thread performs the ethernet communication between the CAN-Q and Xillinux through the creation of a socket. After a reshaping of data from 16 bit to 32 bit the same thread writes the data into the `\dev\xillydemo_32bit_write` port;
- the second reads the data coming from the `\dev\xillydemo_32bit_read` port and sends them to a GUI (Graphical User Interface).


SkyWalkeR v1.1

Connection:
Ip address: 192.168.1.5
Port: 4250

Spike detection:
Conservative (TH: 12.01)
Aggressive (TH: 12.07)
Custom TH: 12.07

Save as: (None)

32 x 32 Matrix:

FIGURE 4.6: SkyWalkeR Graphical User Interface developed in GTK+2 to show the electrical activity map of neuronal network.

4.5 Showing analyzed data

At the end of the processing chain a key element is the possibility to show boolean value of the action potential detection. A simple GUI is developed in GTK+2 with the software Glade. It is a RAD (Rapid Application Development) tool that allows to save the user interface design as XML file. Using the GtkBuilder object of GTK+2 the C application can load it dynamically.

4.5.1 SkyWalkeR graphical user interface

As can be seen in Figure 4.6 the SkyWalkeR GUI is developed. The designed GUI allows to start the first thread for the Socket connection with the CAN-Q through a Start button, inserting the IP address and the port in the specific text boxs. It can also stop the two threads with the Stop button. A 32x32 pixel matrix is designed to show the correlation algorithm output. It can be set in the two modes: the frame-mode, showing sequentially the OR between N consecutive frames; the heatmap mode to collect spike events and showing the electrical activity map of neuronal network. Some buttons to chose the algorithm threshold or save a video are also designed but not already implemented.

Chapter 5

Algorithm implementation tests

To test the correlation algorithm the data recorded from 678 chip are used. A PC is used to simulate the CAN-Q, sending the data through the ethernet port to the router that streams the data to the Zedboard. In this chapter it is shown that the correlation algorithm implemented on FPGA works correctly. The data sent to the Zedboard are filtered before and multiplied by 100 to process integer numbers with better precision.

5.1 Squared RMS map calculator test

This module processes the first N frames streamed to calculate the σ^2 value of the signal fro each pixel. It is possible to set the number N of frames to perform the calculation. The data used for this calculation are lost, so the value is set to 4000, about half second of recording. It will be shown this time is long enough.

The output of this module is sent to the output FIFO and saved in a binary file. Than it is analyzed and compared with the σ^2 value calculated off-line from the same 4000 frames. In Figures 5.1 and 5.2 the σ^2 map is shown, calculated on-line and off-line respectively. As can be seen the output data from FPGA is very closed to the off-line one. The Figure 5.3 shows the percentage error distribution over the 1024 pixels. As can be seen the percentage error results lower than 2%.

5.2 Summation over 9 timeframes comparison

The next algorithm step to test is the sum over 9 consecutive time frame. This test is the evidence of the proper filling of 32x32 memories and working of iterators.
The $(17, 27)$ pixel signal is chosen to show the output from time convolution. The Figures 5.4 and 5.5 show the time convolved signal, calculated on-line and off-line respectively. There aren't differences between the two graphics: this happens because from square and multiply logical operations no approximations occur.

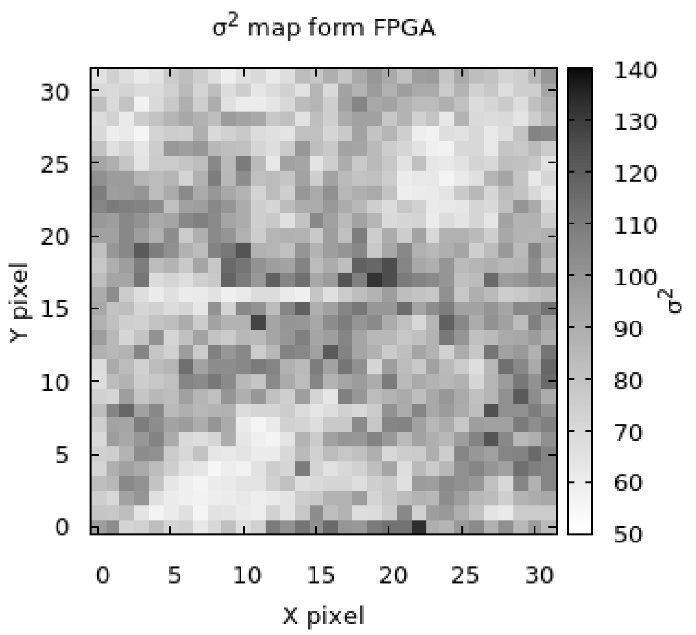


FIGURE 5.1: σ^2 map, calculated by FPGA over 4000 frames, about 0.4 s.

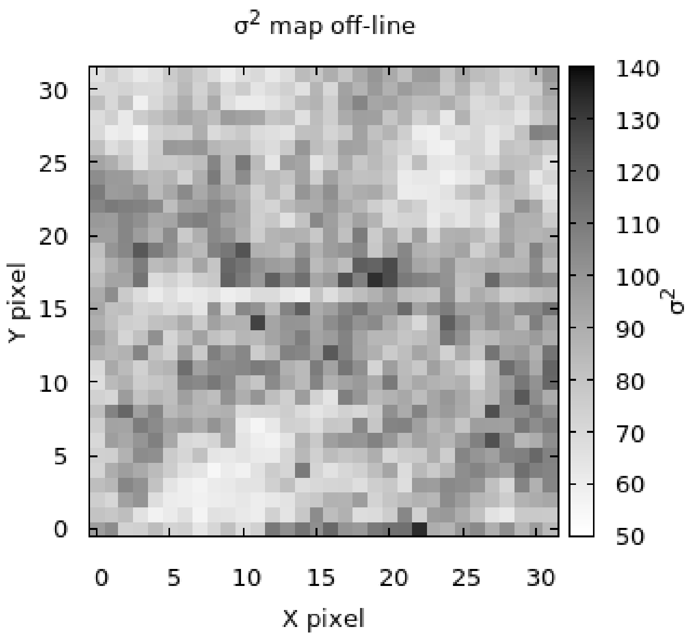


FIGURE 5.2: σ^2 map calculated by a Python script over 4000 frames, about 0.4 s.

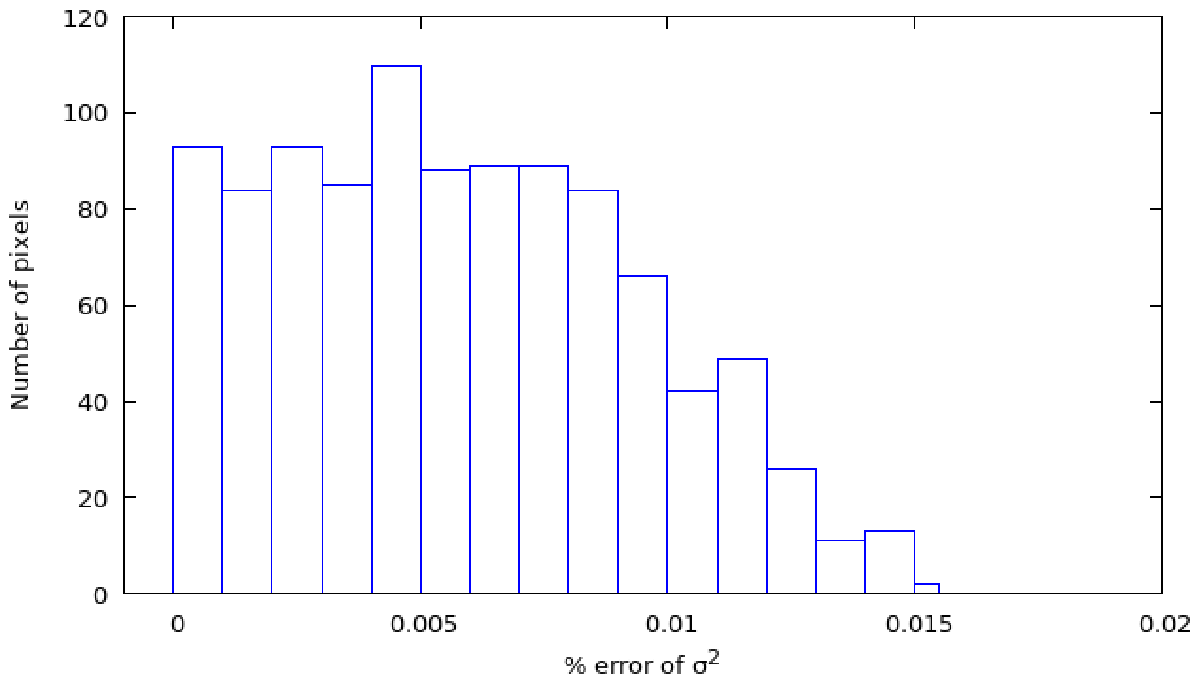


FIGURE 5.3: Percentage error of σ^2 values from FPGA.

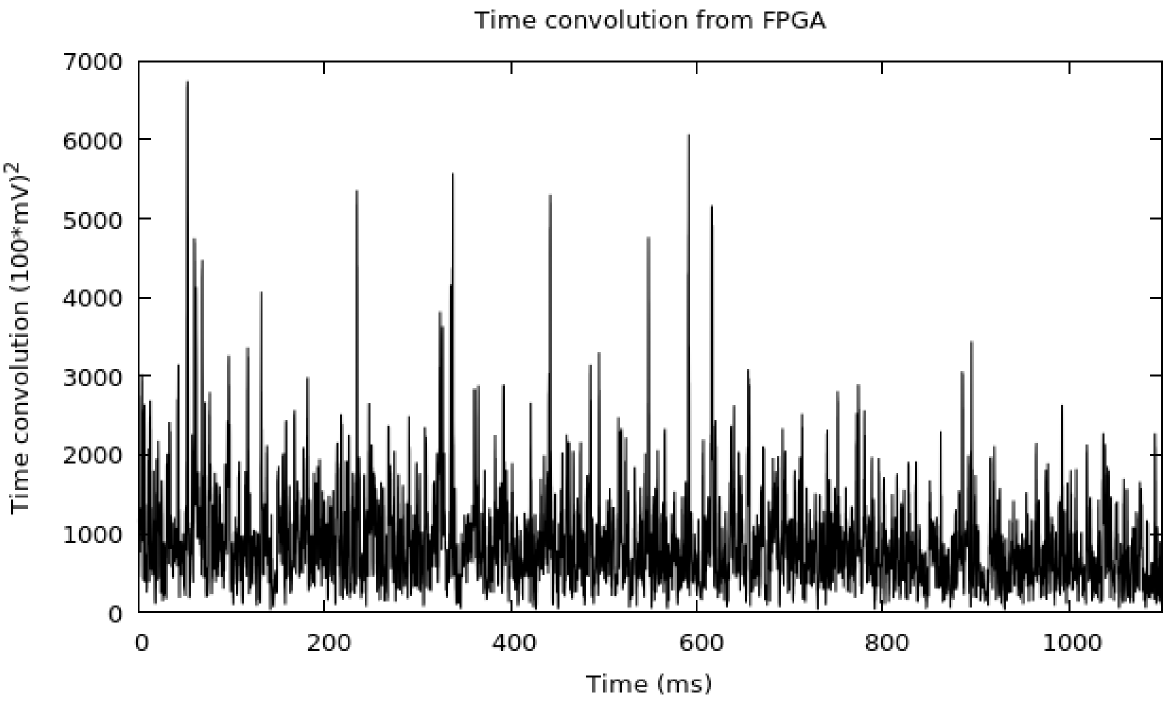


FIGURE 5.4: Time-onvolved signal coming from (17,27) pixel, processed by FPGA.

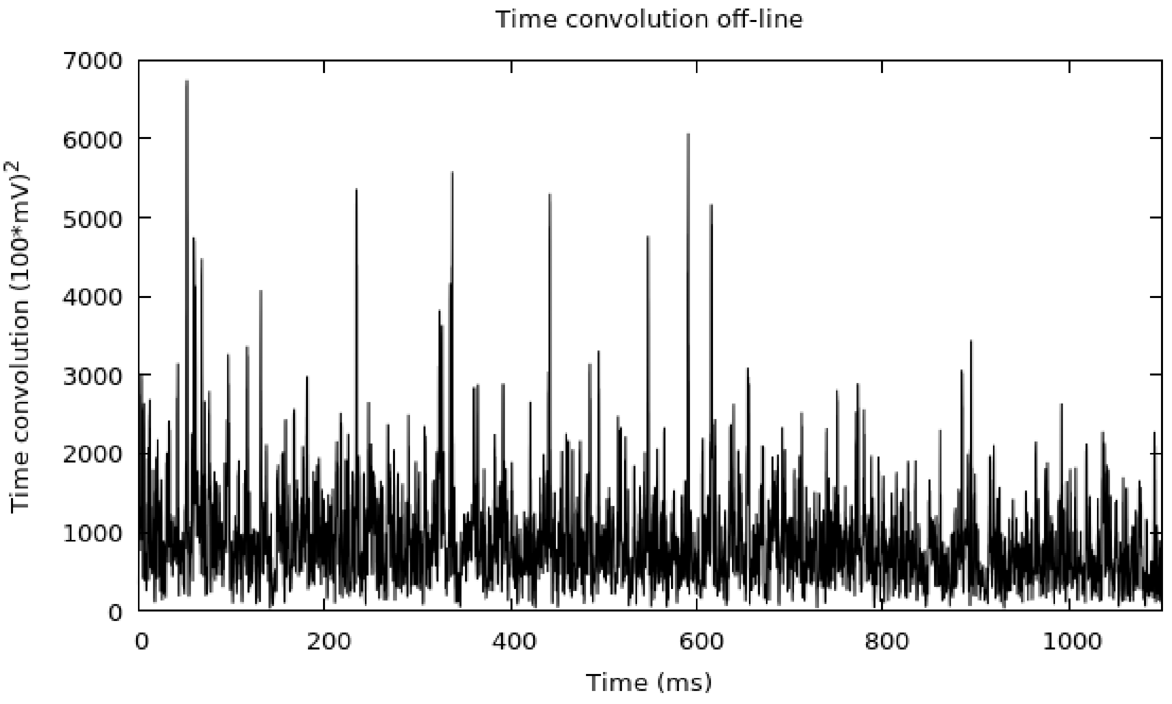


FIGURE 5.5: Time-onvolved signal coming from (17,27) pixel, processed by a Python script.

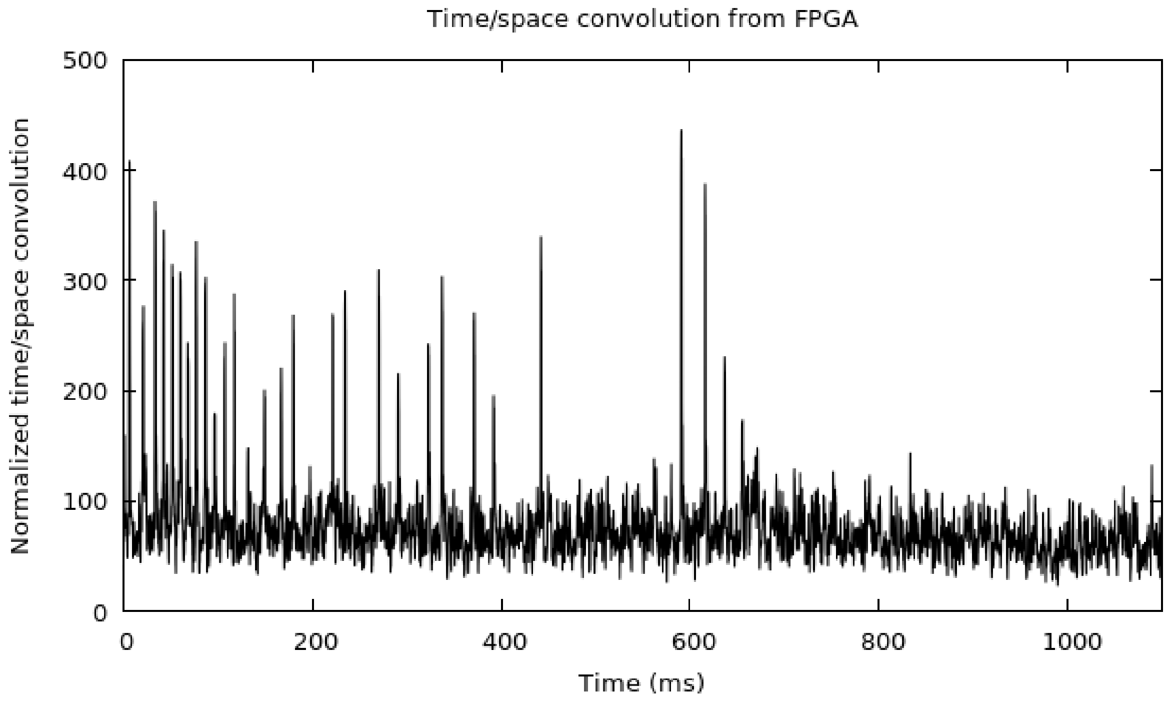


FIGURE 5.6: Normalized time/space-convolved signal coming from (17,27) pixel, processed by FPGA.

5.3 Division and sum over 7 near pixels test

After the time convolution occurs the division by σ^2. The quotient is stored in a 32x32 matrix thanks to which then occurs the sum over 7 near pixels. The Figures 5.6 and 5.7 show the space convolved signal, calculated on-line and off-line respectively. As can be seen they are almost the same. The Figure 5.8 shows the difference between the off-line signal and the on-line one. As can be seen there is a little difference due to the division approximation. This little difference is not a real problem for action potential spike detection because the final threshold can be adjusted accordingly with the offset.

5.4 Activity map showing

The last test performed is the proper showing of activity map on the GUI, coming from FPGA processed data. In the Figure 5.9 can be seen the heatmap recorded through the FPGA. On the GUI there isn't the color bar yet, but can be seen it is almost the same activity map of Figure 3.18, recorded over 1 s.

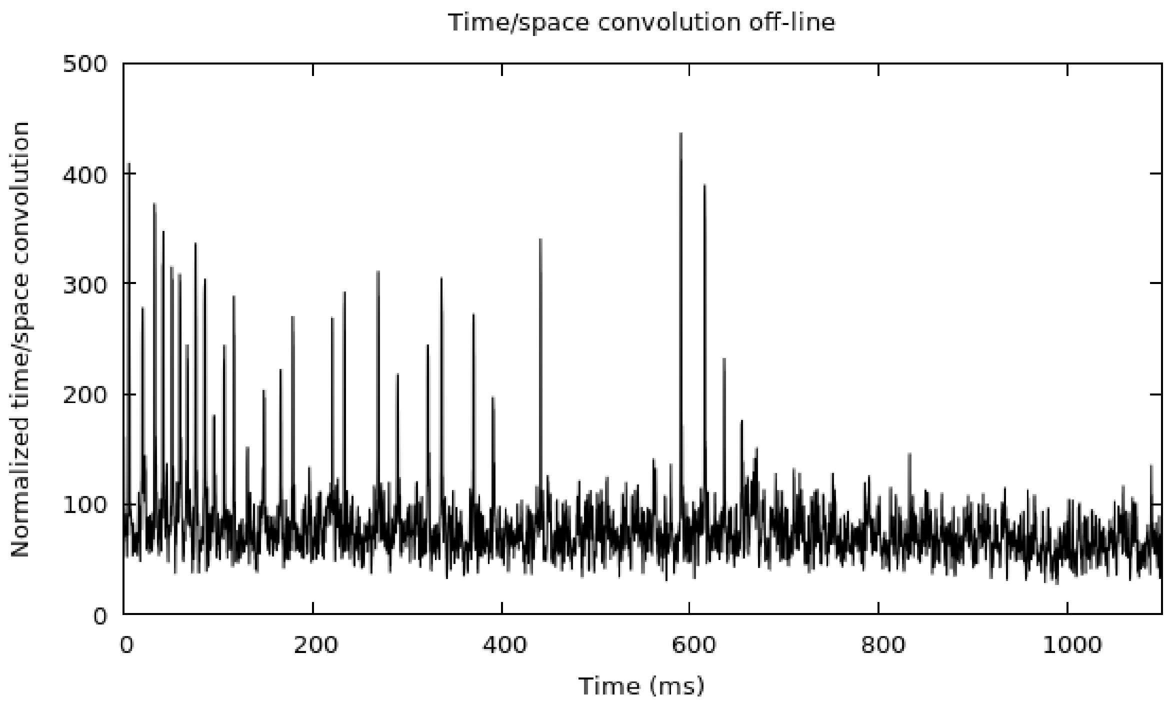


FIGURE 5.7: Normalized time/space-convolved signal coming from (17,27) pixel, processed by a Python script.

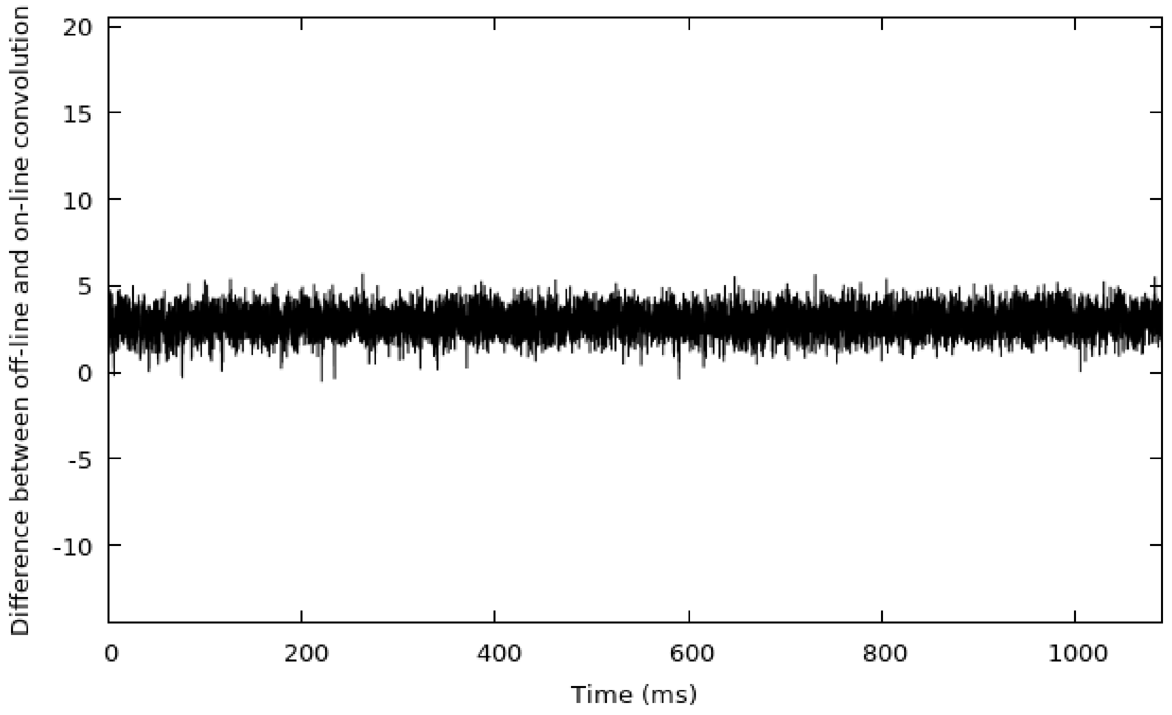
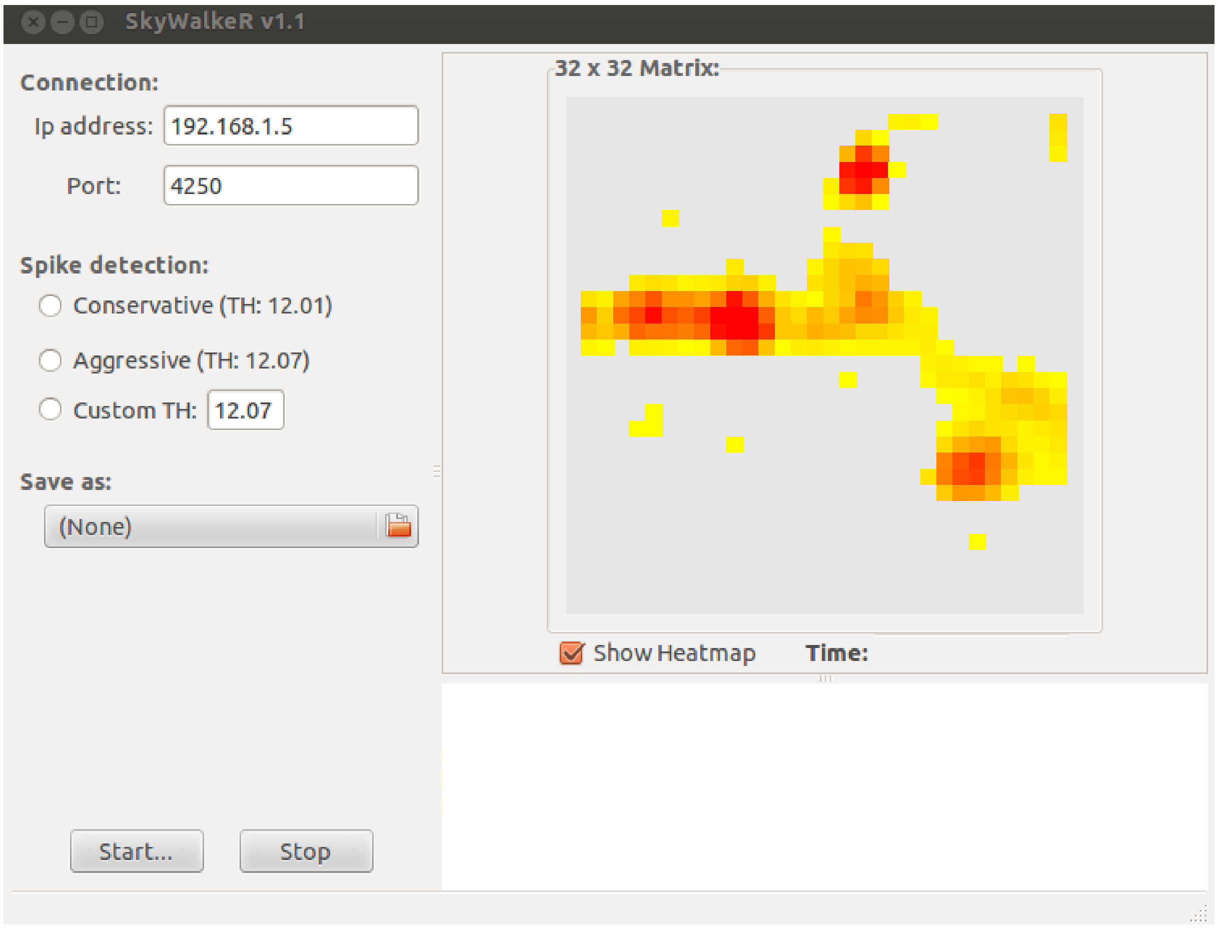


FIGURE 5.8: Difference between convolved signals shown in Figure 5.7 and 5.6.



Connection:
Ip address: 192.168.1.5
Port: 4250

Spike detection:
○ Conservative (TH: 12.01)
○ Aggressive (TH: 12.07)
○ Custom TH: 12.07

Save as: (None)

[✓ Show Heatmap] Time:

FIGURE 5.9: Activity map recorded through FPGA from 678 chip data, shown on the GUI.

Chapter 6

Conclusions and perspectives

In this work it is explained the algorithm used to detect spikes of a neuronal network cultivated on a chip built with the MTA tecnology. In particular it is used the time and spatial correlation probably distribution of measured signals to reduce the number of false positive events.

The measurement are performed with three different solution on the chip: the extracellular solution, the bicuculline solution and the KCl solution. The measured signals are calibrated and filtered off-line and then it is calculated the activity map. The analysis of neuronal activity is possible only with the KCl solution because of cells stimulation. As a matter of fact it is calculated the spiking frequency of a neuron. It results to decrease in less than 1 s from (103 ± 2) Hz to (37 ± 1) Hz and then to (15 ± 2) Hz. This is due to due to the progressive inactivity from too much membrane depolarization using KCl. Secondly it is observed the time correlation between near neurons spikes finding a signal that moves through 4 neurons in about 0.7 ms. Using it the inter neuron signal conduction velocity is measured, resulting (20.1 ± 0.6) cm/s. The time scale of transmission is compatible with the trasmition synaptic delay of 0.3 ms. In future could be possible to understand better the compatibility of this result measuring the timing of neuronal signals transmission.

The second part of the work is devoted at building the acquisition chain of data streaming to show a real-time activity map on a GUI. To do this the correlation algorithm is implemented on FPGA and then it is tested, with a great similarity with off-line data analisys. In future new neuronal activity measurements can be performed with this modality, but first must be implemented a filter too. The algorithm can be optimized and the GUI can perform other tasks, like threshold selection, σ^2 recalculation, video recording, color bar showing.

In perspective of a future development of CAN-Q setup could be possible to record the full 256x384 matrix. This would allow to study the whole neuronal network activity on the chip.

Bibliography

[1] A. Lambacher et al. "Identifying firing mammalian neurons in networks with high-resolution multi-transistor array (MTA)". In: *Applied Physics A* 102.1 (2011), pp. 1-11.

[2] Dale Purves. *Neuroscience*. Sinauer Associates, 2012. ISBN: 9780878936953.

[3] E. A. Vallicelli et al. "Neural Spikes Digital Detector/Sorting on FPGA". 2017 IEEE Biomedical Circuits and Systems Conference (BioCAS 2017), Conference Proceedings.

[4] M. Maschietto et al. "Sodium channel $\beta2$ subunit promotes filopodia-like processes and expansion of the dendritic tree in developing rat hippocampal neurons". In: *Frontiers in Cellular Neuroscience* (25 January 2013).

[5] P. Fromherz. "Neuroelectronic Interfacing: Semiconductor chips with ion channels, nerve cells and brain". In: *Nanoelectronics and Information Technology* (2003), pp. 781–810.

[6] Robert Plonsey and Roger C Barr. *Bioelectricity: a quantitative approach.* Springer Science & Business Media, 2007.

[7] T. L. Fletcher et al. "The Distribution of Synapsin I and Synaptophysin in Hippocampal Neurons Developing in Culture". In: *The Journal of Neuroscience* 11 (June 1991), pp. 1617-1626.

[8] *CAN-Q Station*, Venneos company:
`<http://www.venneos.com/product/can-q-station.html>`

[9] *CAN-Q Analyzer*, Venneos company:
`<http://www.venneos.com/product/can-q-analyzer.html>`

[10] *Divider Generator*:
`<https://www.xilinx.com/support/documentation/ip_documentation/div_gen/v5_1/pg151-div-gen.pdf>`

[11] *Electrochemical gradient*:
`<https://en.wikipedia.org/wiki/Electrochemical_gradient>`

[12] *Getting started with Xillinux for Zynq-7000 EPP*:
<http://xillybus.com/downloads/doc/xillybus_getting_started_zynq.pdf>

[13] *Human Medical Physiology*, on-line blog:
<http://hmphysiology.blogspot.it/2013/10/q-absolute-vs-refractory-period-of.html>

[14] *Nerve conduction velocity*:
<https://en.wikipedia.org/wiki/Nerve_conduction_velocity>

[15] *Primary Hippocampal neurons cultured*:
<http://www.neuvitro.com/customer-feedback-neuron-culture.htm>

[16] *Solid angles in arbitrary dimensions*:
<https://en.wikipedia.org/wiki/Solid_angle#Solid_angles_in_arbitrary_dimensions>

[17] *Spherical volume element*:
<https://en.wikipedia.org/wiki/N-sphere#Spherical_volume_element>

[18] *Xillinux*:
<http://xillybus.com/xillinux>

[19] *ZedBoard Hardware User's Guide*:
<http://zedboard.org/sites/default/files/documentations/ZedBoard_HW_UG_v2_2.pdf>

[20] *Zynq-7000 datasheet*:
<https://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf>