Theses of the PhD thesis

Implementation of neurobiological and various signal processing algorithms using FPGA circuits

László Schäffer

Supervisors:
Szilveszter Pletl, PhD, college professor
Zoltán Kincses, PhD, assistant professor

Doctoral School of Computer Science
University of Szeged

Department of Technical Informatics

2021

1 Introduction

Signal processing is an important part of computer science, which is used in but not limited to automation, pattern recognition, control theory, artificial intelligence, and networking and communication. A signal can be anything, which is measurable and everything that can be measured is also can and inevitably will be processed using analog or digital signal processing methods. It can be a temperature measurement, an image from a camera or a recording of neural brain activity, all of them have to be processed in some way. In signal processing the first step is to measure a physical quantity. After the measurement the quantity can be filtered and amplified. In ideal circumstances a noise-free and proper amplitude quantity can be converted to a digital signal using an analog-digital converter (ADC). This digital signal can be processed by a Central Processing Unit (CPU) / microprocessor, a microcontroller, a Graphics Processing Unit (GPU), a Field Programmable Gate Array (FPGA) or an Application-Specific Integrated Circuit (ASIC).

Changing the hardware in signal processing applications are a complex task and usually requires to modify an optimised software. In many cases only software optimisation offers limited improvements, although the time and energy investment is still significant. In these cases the usage of FPGA makes possible to rapidly change the hardware architecture along with the software and also offers real-time and efficient operation.

The PhD thesis presents FPGA-based real-time signal processing, including the detection and classification of action potential in neurophysiological measurements. A common feature of the presented approaches is real-time implementation using FPGA.

The dissertation consists of three major parts. The first chapter presents real-time detection and synthesis of neurophysiological signals based on FPGA, the second chapter presents real-time classification of neurophysiological signals based on spatial information, while the third chapter presents the application of FPGA-based systems in signal processing and simulation.

2 FPGA-based real-time detection and synthesis of neurophysiological signals

Usually the first method to process the recorded neural activity is spike sorting, which is used in many fields of basic neuroscience research. The identification of spikes of individual neurons is of great importance in some real-time clinical applications, e.g. neuroprosthetic devices or brain-machine interfaces. The first step in a spike sorting algorithm is usually the detection of spikes, which can be computationally intensive.

The performance of current spike detection methods is challenged by multi-channel neural data recorded with high-density, high-channel count silicon probes developed recently. High-density neural probes with closely-packed recording sites can detect the spikes of the same neuron simultaneously on multiple, adjacent sites. In contrast, most spike sorting algorithms are prepared to process data recorded with only a few (usually four) electrodes.

Other challenge is the validation of the spike sorting, because not even the experienced neuroscientists can annotate the recorded neural data with 100% accuracy. Therefore using a hand-annotated ground truth dataset can produce a deceptive validation result.

Neurophysiological signal synthesis can be an important part of spike sorting to produce a reliable ground truth dataset and use it for automatic validation.

In Chapter 2. of the PhD thesis, an FPGA-based multi-channel Non-linear Energy Operator (NEO) based neurophysiological action potential signal detection approach is presented along with an also multi-channel template-based neural signal pattern generator algorithm and its real-time FPGA architecture.

2.1 Overall system architecture

It is necessary to know the structure of the overall system to better understand the detection and synthesis. The overall system information is also relevant in Chapter 3, where the classification part is detailed.

The schematic diagram of the proposed spike sorting system can be seen in Fig. 1, which can be split into three main blocks, the *Recording*, the *Processing* and the *Host PC*. The *Recording* block contains an *Intan RHD2000* electrophysiological recording system. The *Processing* block can be further divided into the *Ethernet* interface, *SPI* interface, Preprocessing *IIR Filter* module, *Spike Generation (FGPA)* (see Section 2.2) module and the *Spike Sorting* module, which contains the *Multi-channel Spike Detect* (see Section 2.4) and the *Multi-channel Online Sorting* (see Section 3.1) cores. The *Host PC* block includes the *Ethernet* interface, the *Visualization* module, the *Spike Generation* module, and the *Validation* module.

The proposed system is able to process in vivo neural measurement recorded with the Intan RHD2000 board, as well as simulated neural recordings. The path of the incoming neural data is handled by the *Source Selector*, therefore stored (offline) data as well as real-time measurements or simulated signals (online) can be used.

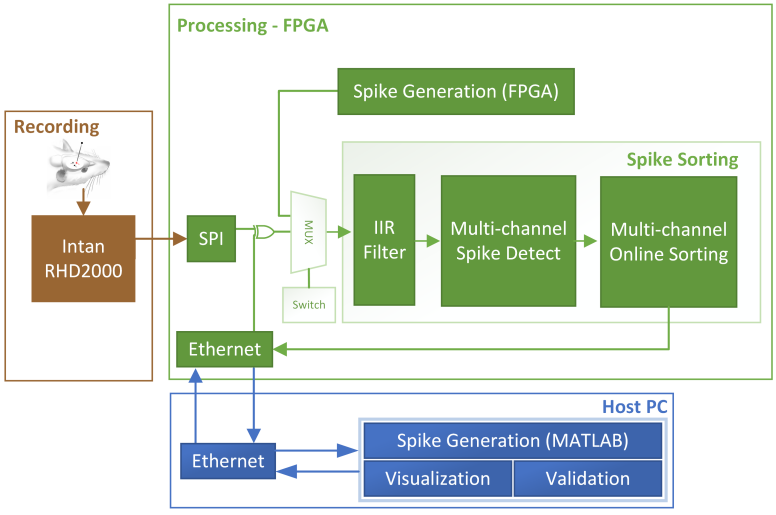

Figure 1: *Schematic diagram of the proposed spike sorting system.*

2.2 Neural Signal Generation

There are several methods to obtain "ground truth" datasets for spike sorting efficiency validation. Simultaneous paired recordings collecting the spikes of the same neuron both extracellularly and intracellularly would be the most optimal solution, however, this method is technically challenging, therefore the availability of such datasets are limited. Simulating the activity of biophysically realistic neural networks is also an option, but the generation of synthesized multi-channel datasets needs high computational power provided only by computer clusters. Finally, "hybrid ground truth" datasets can be generated by using the mean spike waveforms of a subset of well-separated units isolated from real extracellular recordings as templates. Placing these spike templates at random times and positions gives the opportunity to easily simulate neural measurements.

The Spike Generation contains two main parts: the *Mersenne Twister* cores, and the *Neural Dataset Generator* core. The *Mersenne Twister* cores are responsible for the generation of random numbers. Random numbers are required for distances between spikes, for the randomness of positions, which follow a log-normal distribution, and also to obtain the background noise.

The FPGA implementation of the log-normally distributed random number generator is done, however the *ARM Processing System (PS)* has enough computation performance to complete this task, so the calculation does not occupy any FPGA resources.

The *Neural Dataset Generator* core generates the multi-channel neural dataset. The first part of this process is the simulation of the action potential spreading through the neighbouring 6 channels. This is based on a pre-defined normally distributed Gaussian-kernel. The shape of the electrode array is stored in a Look Up Table (LUT), which is used for proper indexing of the weight matrix. For each channel and each neuron, the weight matrix contains the corresponding spreading weight value from the Gaussian-kernel. The electrode array LUT and the weight matrix is pre-calculated on the *ARM PS*.

Therefore, the generation of one output sample is a simple multiplication between the weight coefficient of the actual channel and the value of the actual spike template. Finally, an adder tree summarizes the weighted template values and the result will be the

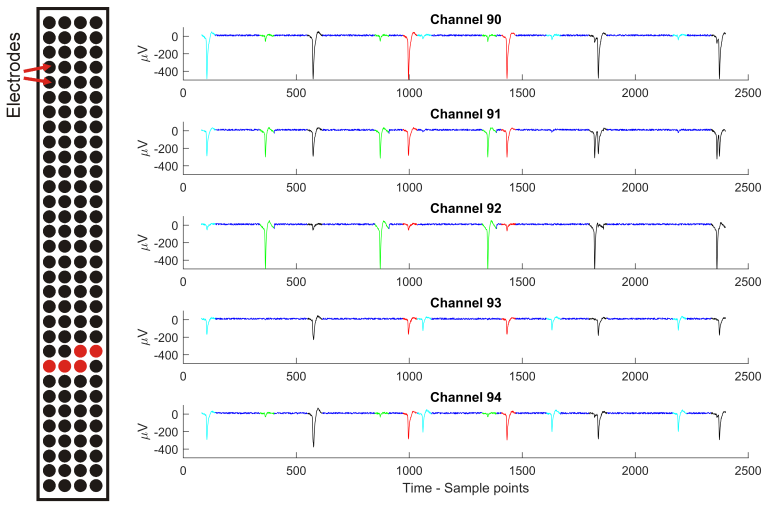

Figure 2: *A* 0.1 *second long segment of the generated neural data on 5 adjacent channels*

output of one channel at one time, so real-time neurophysiological signal generation can be acquired for 128 electrodes with the required 20 kHz sampling frequency. A partial result of the Neural Signal Generator for 128 channels can be seen in Fig. 2.

2.3 Cross-Correlation Based Spike Detection

Using cross-correlation for pattern matching is a proven method, however it is less used for detecting action potentials, because it requires reference templates. Although, if an experienced neuroscientist selects well-distinguishable templates, the cross-correlation method may be effective.

I proposed a cross-correlation-based spike detection method that compares the windowed inbound recording using normalized cross-correlation with the stored templates. If the similarity exceeds a preset threshold value, the signal in the current window is considered a spike.

Since the correlated signal and the minimum point of the spike does not coincide (a so-called lag occurs), a realign step is required. The realign step corrects the lag, so the minimum point of the spike is at the time of detection. Cross-correlation detection can be seen in Fig. 3.

Based on the validation results using simulated neural signals the proposed method achieved an average accuracy of 96.17% for a signal-to-noise ratio of $4 - 10$ dB. Furthermore, the number of false positive cases is the lowest compared to other detection methods.

Without the right quality templates, the proposed method is not suitable for real-time spike detection, but it can be a good candidate for re-detecting and re-classifying an already partially classified recording.

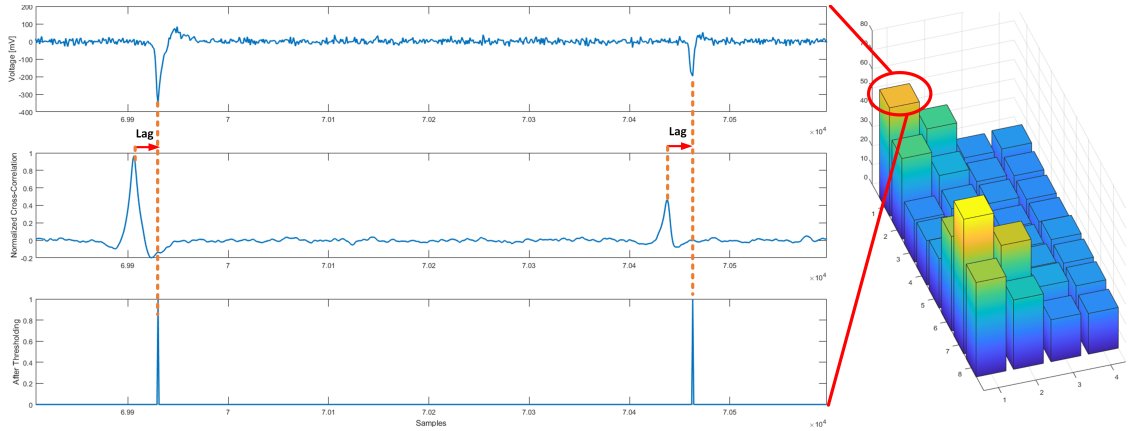

Figure 3: *Cross-correlation based spike detection, visualization on channel 1. From top to bottom: generated neural (3 neurons) signal, normalized cross-correlation, results of thresholding and realign.*

2.4 Multi-channel Spike Detection

Single-channel spike detection is easier, because there are no adjacent electrodes, therefore overlapping spikes are less likely to appear and the number of active cells are small.

On the other hand, using closely packed electrodes the number of cells are proportionally increasing with the number of channels and also the structure of the electrode array is important. A new source of information appears, which is the spatial component. The action potential of a firing neuron propagates through the electrodes with decreasing amplitude in all direction. The spatial information can point towards the source of the action potential to the channel with the highest action potential amplitude.

The *Multi-channel Spike Detect* architecture can be split into the STD, the NEO and the Realign computation parts, which can be seen in Fig. 4. The proposed architecture can process signals sampled at $20\ kHz$ on 128 channels.

The STD computational part is responsible for computing the standard deviation value of the incoming signal using a 5 seconds window, which is read from the off-chip DDR4 memory. The average computation can be done continuously in the AVG block, while the subtraction and power operations are done in the STD block. In the NEO computation part the spike detection with (1) is used.

$$\Psi[x(n)] = x^2(n) - x(n+1)x(n-1) \tag{1}$$

The threshold is calculated using (2).

$$T_{NEO} = C_{\text{NEO}} \frac{1}{N} \sum_{n=1}^{N} \Psi[x(n)] \tag{2}$$

The NEO signal can be calculated parallel for each channel using the NEO BRAM1, BRAM2 and the Serializer.

The Realign part is responsible for the alignment of the detected spikes in the spike window, the determination of the channel containing the maximal amplitude spike in the neighbourhood around the detection, furthermore the selection and transmission of the 3×3 channels on the electrode array. It is required to align the minimum amplitude of the spikes to the center of the spike window, which is critical step for the classification.

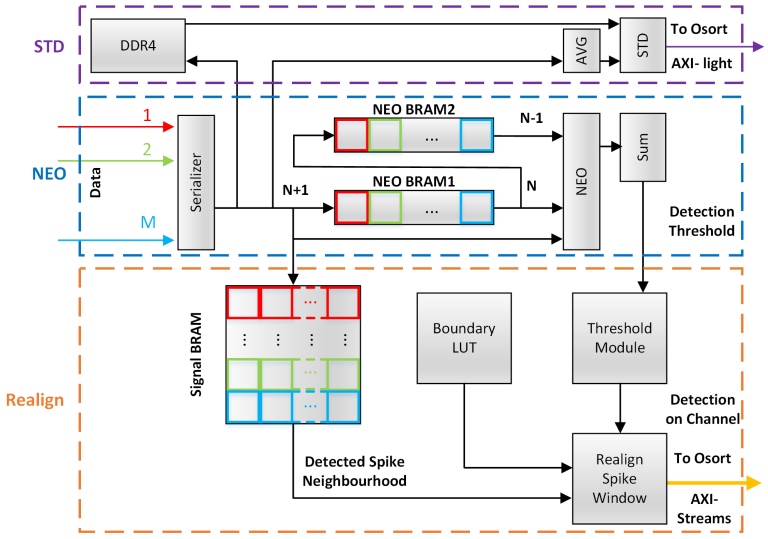

Figure 4: *The architecture diagram of the multi-channel detection*

If selected the channel is at the border of the electrode array, and the spatial window is out of this border, then the 3×3 spatial window is filled out based on the Boundary Look Up Table.

The proposed architecture has a processing performance of $471\ 698$ spike/second, which fulfills the real-time requirements in case of 128 channels and 20 kHz sampling frequency. Based on the validation it can be concluded, that the accuracy of the detection is 100% in case of $10 - 4$ dB Signal-to-Noise Ratios (SNR).

3 FPGA-based real-time classification of neurophysiological signals taking into account spatial information

High-channel-count neural probes comprising over hundred electrodes are able to record the activity of hundreds of neurons from numerous individual brain positions simultaneously. Spike trains of individual neurons can be separated from the detected multi-unit activity in the classification step of spike sorting.

In basic neuroscience research spike classification or clustering is used during the offline analysis of the recorded neural data as well as in real-time clinical applications (e.g. in brain-machine interfaces to control neuroprosthetic devices). However, using a typical spike sorting solution with general-purpose computers the real-time processing of multi-channel neural data is challenging and can greatly reduce the efficiency of clinical applications designed to provide rapid feedback.

In Chapter 3. in the PhD thesis, a real-time multi-channel Online Sorting based algorithm and FPGA architecture is presented, which is capable of using the spatial information of the neural probe.

3.1 Spatial Window-Based Online Sorting

An already available template-matching based unsupervised spike classification method is the Online Sorting (OSort), which - as the name suggests - was designed to operate in real-time, providing rapid feedback. The known disadvantage of the method is that it can only process a single channel at a time and does not take into account the connections between the channels. OSort uses the spikes as templates, so the alignment of each spike is critical. The similarity between the samples and the clusters is determined by square difference, and the required threshold value is determined automatically from the signal according to the standard deviation of an appropriate sized time window.

The efficiency of the OSort algorithm is already proven by various FPGA implementations, which significantly outmatches any CPU based solutions. However, most of the FPGA implementations are only able to process one channel in real-time.

I proposed an extension of the original OSort algorithm with the usage of spatial information from the selected electrode array window and a cluster memory, which makes it more efficient and also still implementable on FPGA.

The data received from the multi-channel spike detect part consists of the spikes of the selected electrode window. By appending the spikes in this window one after the other, they become usable for pattern matching. Thus, the classification algorithm takes into

account the neural activities from multiple channels and the connection between these activities.

A critical parameter of the classification is the value of the clustering and cluster merging thresholds, which are automatically calculated and adaptively changed during the classification based on the incoming standard deviation values, which is calculated and provided continuously by the spike detection part.

When the spike detect part sends the first spike matrix containing 9 spikes from the 3×3 electrode window, it is stored as the first cluster in the cluster memory. The next spike matrix will be compared to the saved cluster mean. If the calculated distance is below the threshold it is assigned to the cluster, otherwise the creation of a new cluster is required. This process is applied to the subsequent incoming spike matrices. After the assignment, the mean of the cluster will be updated, because the composition of the cluster is changed. Furthermore, the cluster mean update changes the distance between cluster means, therefore a distance check between clusters is needed. If a distance is below the threshold, then the updated and the closest cluster will be merged together. In this case the smaller cluster will be removed from the cluster memory.

3.2 FPGA architecture

The architecture of *Multi-channel Online Sorting* core is shown in Fig. 5. The detected spike data is received via AXI-Stream bus and arrive to a 14 spike matrices deep buffer to handle multiple detections. During a 0.1 ms period only one neuron will fire on average in the 3 x 3 window of a recording site. Therefore, on 128 channels 14 neurons can fire in average in a 0.1 ms period, so a system capable of storing 14 spike matrices will cover most of the cases, even when multiple neurons fire in a short time frame.

Partial results of the summed squared difference is computed by the Arithmetic Units. The partial results are summed by the Adder Tree for each sample and an accumulator (ACC) is used to compute the sum over the entire window. The result and the cluster number is saved into the MIN register if it is smaller than the previous minimum squared

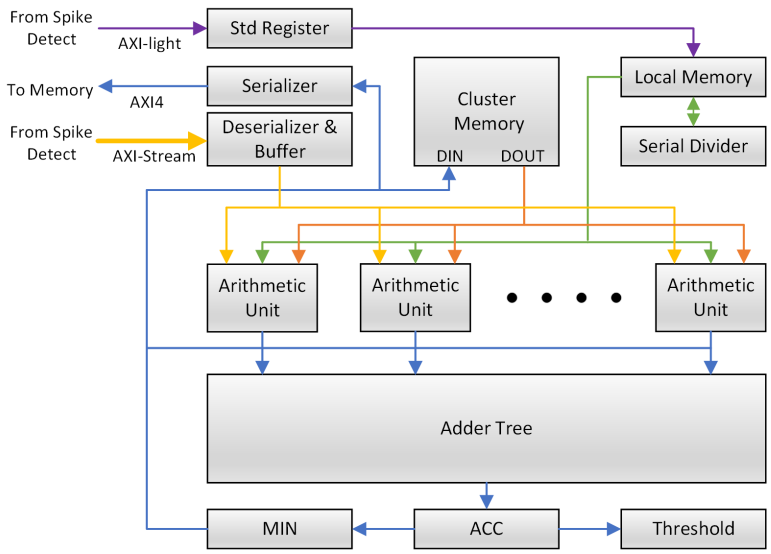

Figure 5: *Architecture of the Multi-channel Online Sorting core*

distance compared to the threshold value. These steps should be executed for each active cluster.

The weights are pre-computed by the serial divider. The new number of spikes and the new weights are stored in the local memory.

The architecture computes the incoming data in 4 stages. In the first stage the data is loaded and compared to the cluster memory. The second stage updates the chosen cluster mean or creates a new cluster. In the third stage the comparison of the cluster means to each other, while in the fourth stage the merge of clusters are done.

The proposed architecture has a classification performance of $11\,741$ spike matrix/second, which fulfills the multi-channel real-time requirements.

Based on the validation results, it can be concluded, that the average classification accuracy is 86% in case of high neuron numbers (16-32) and more than 3 dB SNR. However, the original (single channel, no spatial information) OSort reaches only 74% average accuracy in the same conditions.

Furthermore, the proposed architecture was tested with a 5-minute-long in vivo cortical dataset obtained with a 128-channel silicon probe from an anesthetized rat. After using the offline kiloSort spike sorting algorithm the resulting clusters were manually curated. Running the proposed architecture on this cortical recording, 32 clusters were created, which showed 80% similarity using cross-correlation.

4 Application of FPGA-based systems in the field of signal processing and simulation

Signal processing is an important part of artificial intelligence, control theory, simulation, pattern recognition, which fields nowadays are becoming more and more overlapping. Some of the current signal processing algorithms can be used in embedded systems, which can also be applied in real-time applications, if the requirements are fulfilled. These requirements on the software part are parallel optimization capability, low memory usage and low processing bandwidth. On the hardware requirement part there is low power consumption and small form-factor.

Algorithms that fulfill these software requirements could be accelerated to be used in real-time applications using FPGAs. Furthermore using FPGAs apparently provide small form-factor and low power consumption and the possibility to further improved into ASICs.

In Chapter 4. in the Phd thesis, some signal processing algorithms from different computer science research fields are presented, which can be modified to be accelerated with FPGAs for real-time applications. These applications are from the face recognition, sensor fusion, suspension control and hardware-in-the-loop (HIL) simulation topics.

4.1 Face recognition low-cost real-time FGPA architecture

A widely used method for face recognition is the Eigenfaces approach, which means the usage of Principal Component Analysis (PCA) on a database containing human faces. Due to the dimension reduction of the features the calculation can be faster and the implementation is easier than other face recognition approaches. It can be concluded, that Eigenfaces is tolerant to small rotation, translation and scale changes.

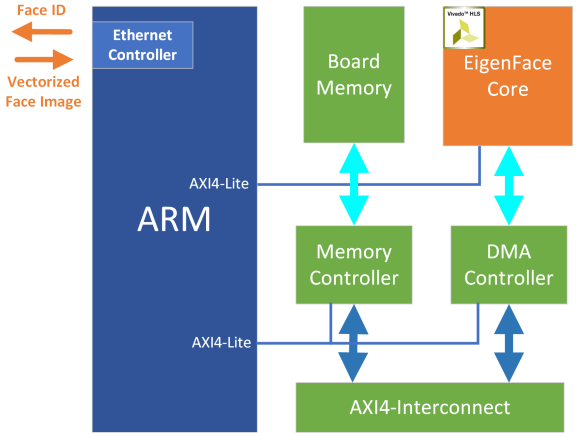

Figure 6: *Architecture of the face recognition system*

The architecture of the proposed system consists of six main parts as can be seen in Fig. 6. These are the *ARM Processor*, the *DMA Controller*, the *AXI-4 Interconnect*, the *Memory Controller*, the *Board Memory* and the *EigenFace Core*.

The *EigenFace Core* computes the algorithmic steps of the EigenFaces method. In the first step it calculates the mean face. In the second step the difference from the mean is computed for each face vector. As the third step the eigenvectors are computed. In the fourth step the projection to the face space is computed.

If the system is in recognition mode, then finally the distances between the projections of the training set and the projection of the input face is calculated. Only the data for the actual step is stored, to minimize the BRAM memory usage.

In mean face calculation only addition and a multiplication is required at the end of each row, when the faces are column vectors. This calculation can be done parallel along on one dimension of the faces. The eigenvector computation consists of three matrix multiplication, which unfortunately can not be done parallel due to data dependency. However, the multiplication of two matrices can use parallelism. After one column of the resulting matrix is computed it can be used immediately for the next matrix multiplication. The projection step includes the multiplication of each face vector with the eigenvector matrix, which can be parallel along the dimension of the face vector. Finally, the euclidean distance calculation is required, which is done serially due to the high DSP requirements of the square roots.

The proposed architecture is capable of processing $13\ 026$ face/second, and the recognition accuracy is 95% on the validation dataset using the low-cost Zybo FPGA board.

4.2 Sensor fusion low-cost real-time FGPA architecture

On a mobile vehicle the calculation time of the estimations is critical, because the environment can change swiftly, therefore the measurements have to be processed in real-time. Taking advantage of the hardware acceleration capabilities of a Field-Programmable Gate Array (FPGA) the processing time can be significantly reduced.

The architecture of the proposed system is built-up from seven main parts as can be seen in Fig. 7. These are the *ARM Processor*, the *DMA Controller*, the *AXI-4 Interconnect*, the *Memory Controller*, the *Board Memory*, the *I2C Controller* and the *Sensor Fusion Core*.

The proposed architecture is capable of handling measurements from various sensor modules, in this case an Inertial Measurement Unit (IMU) and a GPS. The *ARM Processor* receives GPS data via the built-in *UART Controller* and forwards to the *Sensor Fusion Core*. The *I2C Controller* communicates with the IMU module. Since the system can be extended with other sensor boards, an additional *SPI Controller* can be added to the system during the implementation if it is required.

The computation of the Kalman-filter based sensor fusion are implemented in four steps. In the first step the states and the matrices are initialized during the start-up. In the second step only IMU based position and angle estimation is calculated, which is necessary because the sampling rate of the IMU is the highest. In the third step the fusion of the position with a new measurement of a less frequent sensor is computed, if there is an appropriate sensor connected to the FPGA (e.g. GPS). In the fourth step the fusion of the angle with a new measurement of another less frequent sensor is calculated, if there is an appropriate sensor connected to the FPGA (e.g. magnetometer). The ARM processor determines the required step and starts the measurement through the appropriate communication protocol.

The ARM Processor is capable of computing the coordinate transformations and time synchronizations in real-time, so it does not require any FPGA resources. The different measurements of the different sensors are timestamped on the ARM processor and compared to the FPGA timer. Using the time differences the reliability of the measurements can be adjusted.

Two versions of the algorithm were synthesized to the FPGA. The first version of the optimized matrix multiplication were unrolled, while in the second version the multiplications were performed on each FPGA clock cycle. The vectors were stored in LUTs, the covariance matrices were stored in BRAMs.

The pose (position and orientation) estimation was tested on a GPS/IMU recording where the vehicle travels $50meters$ along a curved path. The average error of the sensor fusion was $0.4m \pm 0.18$ both in MATLAB and with the proposed architecture. The proposed architecture on the Zybo FPGA board operates 358 times faster and uses 9.65 times less energy compared to the algorithm running in MATLAB on the PC with the same dataset.

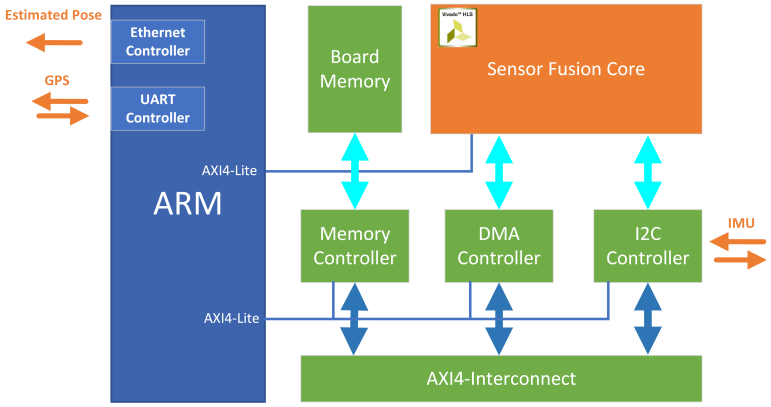

Figure 7: *Architecture of the sensor fusion system*

4.2.1 UWB based indoor localization extension

Localization is an important part of any indoor application which contains object tracking or environment mapping. Many indoor localization techniques (Angle of Arrival - AoA, Time of Flight - ToF, Return Time of Flight - RToF, Received Signal Strength Indicator - RSSI) and technologies (WiFi, Ultra Wideband - UWB,Bluetooth, Radio Frequency Identification Device - RFID) exist, which can be applied to the indoor localization problem.

Based on the measured distances, the position of the object can be estimated using several mathematical methods. The multilateral algorithm is an excellent choice, because its easy implementation. In case of the UWB technology the precision of the estimated position crucially depends on the placement of the anchors (fixed positioned UWB transceiver). The error characteristics of the DWM1001 UWB ranging module were measured and used to find an optimal anchor placement, which improves position estimation accuracy. Using the proposed UWB extension with the proposed sensor fusion architecture indoor localization is possible.

In the proposed architecture the sensors can be changed, and various sensors can be used to measure the same parameter. Therefore the GPS sensor can be replaced with an interface board using the DWM1001 module, which also uses the UART communication protocol. In this case the interpretation of the incoming data packets and the corresponding parameters in the sensor fusion core have to be adjusted.

The reliability parameter can be adaptively changed based on the error characteristics of the UWB device, which is stored in a LUT. The Moore-Penrose pseudo-inverse is an important step in the multilateral algorithm, which can be computed only once, so it can be stored also in a LUT. This is possible, because it depends only on the reference anchor points, which are fixed and have to be known beforehand. The multilateral calculation with the matrix-vector multiplication when each row computed in parallel takes 59 clock cycles. Together with the sensor fusion computation it takes 165 clock cycles, which also makes real-time operation possible.

Unfortunately, together with the sensor fusion architecture the resource consumption exceeds the available DSP resources of the smallest Zybo (Z-7010) FPGA chip. Therefore the Z-7020 Zybo could be used, which has more than double resources.

4.3 Active suspension control low-cost real-time FGPA architecture

In an active suspension system, some state variables can not be measured directly and the usage of sensors is not necessary, due to the cost, accuracy and reliability. Therefore a full state feedback control can not be used, unless a state observer is applied.

The easy reconfiguration of FPGAs gives opportunity for education purposes, because students can analyze the operation of observer based control, and study the effect of parameter changes, without the risk of damaging or endangering the vehicle or the suspension system.

The main equation of the actual observer can be written as follows:

$$\frac{d\hat{x}}{dt} = F\hat{x} + Gy + Hu, F = A - GCA \tag{3}$$

where $\hat{x}$ is the estimated state vector, F, G and H are the observer gains. The dimension of F is 4×4, while G and H are only 4×1 vectors. All of the observer gains can be pre-

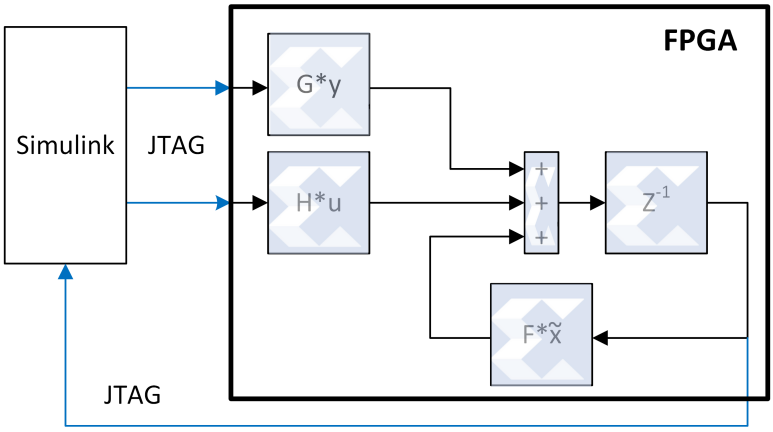

Figure 8: *The active suspension control FPGA architecture system*

computed. The actual observer can be calculated with three matrix multiplication, and a summation. The values of the previous state estimation should be stored in registers and used in the multiplication with the F matrix.

The MATLAB Xilinx System Generator was used for the FPGA implementation, which helps in the design FPGA based state control in Simulink. Using Co-Simulation through JTAG the FPGA architecture can be used together with the Simulink model. The block diagram of the FPGA-based actual observer can be seen in Fig. 8.

The active suspension system was tested both using Simulink-based simulation and real FPGA hardware. The results show that an output of the actual observer can be calculated in $0.8\mu s$, which is 125 times faster than with MATLAB based simulation only. Furthermore, it can be concluded that the control force signal emitted by the FPGA is similar to the Matlab/Simulink model.

The proposed architecture has minimal FPGA utilization, the bottleneck is the number of DSPs. The low-cost Zybo FPGA board is also capable of operating the architecture.

4.4 Wind turbine hardware emulation HIL FPGA architecture

Based on the literature, a dynamically adjustable FPGA-based HIL device has not been developed for horizontal three bladed variable pitch wind turbine modelling. Due to the great flexibility, high computational capacity and boundless reconfiguration possibilities of FPGAs, FPGA-based HIL devices can be used to easily implement various types of wind turbine systems. Furthermore, the HIL approach is well applicable in education of wind turbine based systems, because students can analyze the operation of the wind turbine, and study the effect of different parameter changes, without the risk of damaging or endangering the real physical system.

The Wind Turbine Core is implemented using the Xilinx System Generator Simulink toolbox and it is responsible for the real-time computation of the output torque (T) based on the input parameters. Two type of wind turbine models are implemented, which are based on the normalized and the non-normalized equations. The common parameters are the wind speed (v) and the blade pitch angle (β). In the non-normalized model the blade radius (R), the air density (ρ) is also adjustable.

The *AD* and *DA* cores converts the wind speed and output torque values to analogue or

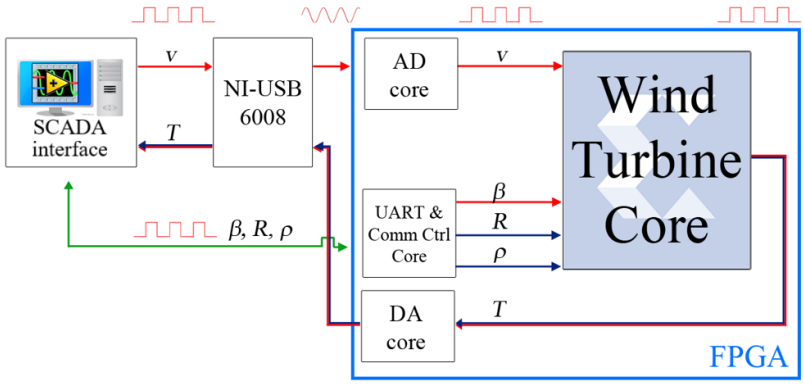

Figure 9: *The FPGA-based architecture in the HIL system.*

digital signals. The *UART & Comm. Ctrl. Core* uses a state machine, which stores the other parameter values into registers, writes/reads their states.

The best way to describe the behavior of a wind turbine is the Cp–TSR characteristic, which plots the extracted power from the wind (Cp) in the function of the ratio between the tangential speed of the tip of a blade and the actual speed of the wind (TSR). The FPGA and the Simulink-based characteristics are very similar. The results showed that a three bladed variable pitch wind turbine can be implemented on an FPGA and used as a real-time dynamically adjustable HIL system.

5 Contributions of the thesis

In **first thesis group**, my contributions are related to FPGA-based real-time detection and synthesis of neurophysiological signals. Detailed discussion can be found in Chapter 2.

I / 1. I gave a Non-Linear Energy Operator (NEO) based multi-channel action potential detecting FPGA architecture, which is also capable of determining the source of the firing neuron in a multi-channel electrode array. I showed that it can operate in real time.

I / 2. I proposed a cross-correlation based action potential detection method, showed its efficiency and pointed out its limitations.

I / 3. I designed a neurophysiological signal generator algorithm, which is capable of placing action potential templates on randomly selected channels with randomly determined frequency and firing crosstalk, taking into account the theory of neuron firing known in neuroscience.

I / 4. I gave an FPGA architecture, which is capable of generating neurophysiological signals using the Mersenne-Twister pseudo-random number generator algorithm. I showed that the architecture can generate real-time neurophysiological signals for an electrode array with 128 channels.

In the **second thesis group**, the contributions are related to the FPGA-based real-time classification of neurophysiological signals taking into account spatial information. Detailed discussion can be found in Chapter 3.

II / 1. I proposed the use of inter-channel spatial information in multi-channel neurophysiological recordings using the Online Sorting algorithm. I compared the efficiency of the original single-channel Online Sorting algorithm to the proposed algorithm on synthetic data. I showed that the proposed algorithm, which uses spatial information has better efficiency.

II / 2. I gave an FPGA architecture based on the Online Sorting algorithm capable of classifying multi-channel neurophysiological recordings in real-time. I showed the effectiveness of the architecture with tests based on synthetic data and real measurements. I showed the FPGA architecture can operate in real time and is suitable for immediate feedback during experiments.

In the **third thesis group**, the contributions include novel FPGA-based systems that make signal processing and rapid prototyping more efficient in some research areas. Detailed description can be seen in Chapter 4.

III / 1. I proposed a new parallel cost-effective FPGA architecture, which capable of operating the Principal Component Analysis (PCA) algorithm in real-time.

III / 2. I gave a new parallel cost-effective FPGA architecture of sensor fusion, which is able to estimate position and orientation in real-time using the measurements of multiple sensors. I proposed an indoor localization solution, which allows the incorporation of information regarding the absolute position into the sensor fusion architecture. Furthermore, I gave an FPGA architecture of the two-dimensional multilateral algorithm.

III / 3. I developed a reliable, cost-effective method, which uses a minimal number of sensors to effectively control the active suspension system of a vehicle. I gave a new parallel FPGA architecture based on an actual observer, which controls the active suspensions force. Furthermore, I showed the real-time functionality of the FPGA architecture.

III / 4. I gave a new parallel FPGA architecture, which is capable of the real-time hardware emulation of a wind turbine in a hardware-in-the-loop (HIL) system. I developed a SCADA interface, which can manipulate the parameters of the system in real-time. Furthermore, I showed the real-time functionality of the FPGA-based hardware emulated wind turbine HIL system.

Table 1 summarizes the relation between the thesis points and the corresponding publications.

Table 1: *Correspondence between the thesis points and my publications.*

Publication	Thesis point									
	I/1	I/2	I/3	I/4	II/1	II/2	III/1	III/2	III/3	III/4
[1]								•		
[2]	•				•	•				
[3]										•
[4]	•				•	•				
[5]							•			
[6]			•	•						
[7]									•	
[8]								•		
[9]		•								

The author's publications on the subjects of the thesis

Journal publications

[1] Á. Kaló, Z. Kincses, **L. Schäffer** and Sz. Pletl Indoor localization simulation framework for optimized sensor placement to increase the position estimation accuracy. *Annales Mathematicae et Informaticae*, 51, 29-39, 2020.

[2] **L. Schäffer**, Z. Nagy, Z. Kincses, R. Fiát and I. Ulbert Spatial Information Based OSort for Real-Time Spike Sorting Using FPGA. *IEEE Transactions on Biomedical Engineering*, 68(1), 99-108, 2021.

Full papers in conference proceedings

[3] **L. Schäffer** and Z. Kincses. Implementation of an FPGA-based wind turbine HIL model. In *Proceedings of the 3rd Mechedu Conference*, Subotica Tech, 159-163, 2015.

[4] **L. Schäffer**, Z. Nagy, Z. Kincses and R. Fiáth. FPGA-based neural probe positioning to improve spike sorting with OSort algorithm. In *Proceedings of the 2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, IEEE, 1375-1378, 2017.

[5] **L. Schäffer**, Z. Kincses and Sz. Pletl. FPGA-based low-cost real-time face recognition. In *Proceedings of the 15th International Symposium on Intelligent Systems and*

Informatics (SISY), IEEE, 35-38, 2017.

[6] **L. Schäffer**, Z. Nagy, Z. Kincses and R. Fiáth. FPGA-based real-time multichannel neural dataset generation. In *Proceedings of the 2017 European Conference on Circuit Theory and Design (ECCTD)*, IEEE, 1-4, 2017.

[7] **L. Schäffer**, Sz. Pletl and Z. Kincses. Implementation of an FPGA-based actual observer for active suspension control. In *Proceedings of the 4th International Conference and Workshop on Mechatronics in Practice and Education*, Subotica Tech, 28-32, 2017.

[8] **L. Schäffer**, Z. Kincses, Sz. Pletl. A Real-Time Pose Estimation Algorithm Based on FPGA and Sensor Fusion. In *Proceedings of the 16th International Symposium on Intelligent Systems and Informatics (SISY)*, IEEE, 149-154, 2018.

[9] **L. Schäffer**, Sz. Pletl and Z. Kincses. Spike Detection Using Cross-Correlation Based Method. In *Proceedings of the 23rd International Conference on Intelligent Engineering Systems (INES)*, IEEE, 175-178, 2019.

Further related publications

[10] **L. Schäffer**, Z. Nagy, Z. Kincses, Z. Vörösházi, R. Fiáth, I. Ulbert, P. Szolgay. FPGA-based clustering of multi-channel neural spike trains. In *The 15th International Workshop on Cellular Nanoscale Networks and their Applications*, VDE, 115-116, 2016.

6 Összefoglalás

Az értekezés FPGA alapú valós idejű jelfeldolgozást ismertet, magába foglalva neurofiziológiai méréseken történő akciós potenciál detektálását és osztályozását, valamint FPGA segítségével valós időben elvégezhető egyéb jelfeldolgozási alkalmazásokat. A bemutatott megközelítések közös vonása az FPGA alkalmazásával történő valós idejű kivitelezés.

A munka három fő témakörből áll. Az első fejezetben a neurofiziológiai jelek FPGA alapú valós idejű detektálása és szintézise, a második fejezetben a neurofiziológiai jelek térbeli információk figyelembevételével történő FPGA alapú valós idejű osztályozása, míg a harmadik fejezetben FPGA alapú rendszerek jelfeldolgozási és szimulációs témakörben való alkalmazása olvasható.

A neurofiziológiai jelek FPGA alapú valós idejű detektálása és szintézise című fejezetben többféle tüske detektálási módszert hasonlítottam össze, majd a valós idejű működés és párhuzamosíthatóság szempontból ideálisat választottam ki és valósítottam meg FPGA-n. A detektálás és osztályozás teszteléséhez egy hibrid sablon alapú neurofiziológiás jelgenerátort fejlesztettem ki és valósítottam meg FPGA-n, amely a tüske sablonokat véletlenszerű tüzelési frekvenciával és tüzelési áthallással helyezi el az elektródatömbön. Így szoftveresen tesztelhetőek a több csatornán működni képes detektáló és osztályozó algoritmusok és hardveresen validálhatóak az FPGA alapú többcsatornás detektáló és osztályozó áramkörök. Továbbá létrehoztam egy valós időben működő Nem-lináris Energia Operátor (NEO) alapú többcsatornás akciós potenciál detektáló FPGA architektúrát, amely képes az elektródák közötti áthallás felhasználásával a tüzelés forrását tartalmazó csatornát meghatározni és validáltam a hatékonyságát a jelgenerátor felhasználásával.

A neurofiziológiai jelek FPGA alapú valós idejű osztályozása térbeli információk figyelembevételével című fejezetben egy FPGA alapú valós idejű tüske osztályozó architektúrát mutatok be, amely figyelembe veszi a térbeli információkat az egymáshoz közeli elektródákról és osztályozza a többcsatornás neurofiziológiás adatokat. Az eredeti Online Sorting algoritmust módosítottam és létrehoztam hozzá egy FPGA architektúrát, amely egyszerre 128 csatornát is képes feldolgozni és az egymáshoz közeli elektródákon való áthallási (térbeli) információt használja fel az osztályozás során.

Az FPGA alapú rendszerek jelfeldolgozási és szimulációs témakörben való alkalmazása című fejezetben különböző kutatási területeken alkalmazott algoritmusok FPGA segítségével való valós idejűvé tételét mutatom be. Az általam választott kutatási területek az arcfelismerés, szenzorfúzió, szabályozástechnika és hardware-in-the-loop (HIL) szimuláció, mely témakörökben egy-egy új FPGA alapú áramkört hoztam létre.

Arcfelismerés témakörben egy PCA alapú valós időben működni képes költséghatékony arcfelismerő FPGA architektúrát hoztam létre. Szenzorfúzió témakörben egy valós időben pozíciót és orientációt becsülni képes szenzorfúziós FPGA architektúrát alkalmazó rendszert fejlesztettem ki. Szabályzástechnika témakörben egy aktív felfüggesztéshez a felhasználni kívánt erő nagyságát szabályzó valós időben működő FPGA architektúrát hoztam létre. HIL szimuláció témakörben szélturbina működését emuláló FPGA architektúrát hoztam létre, amely egy hardware-in-the-loop rendszer részét képezte.

Nyilatkozat

Schäffer László "Implementation of neurobiological and various signal processing algorithms using FPGA circuits" című PhD disszertációjában a következő eredményekben **Schäffer László** hozzájárulása volt a meghatározó:

- Non-Linear Energy Operator (NEO) alapú valós idejű többcsatornás akciós potenciál detektáló és térbeli aktivitás kereső FPGA architektúra – Tézéspont: I/1, Fejezet: 2, Cikkek: [2], [4]
- Keresztkorrelációs akciós potenciál detektáló módszer – Tézispont: I/2, Fejezet: 2, Cikkek: [9]
- Egy neurofiziológiai jelgenerátor algoritmus, amely képes véletlenszerűen kiválasztott csatornákra, véletlenszerűen meghatározott frekvenciával és tüzelési áthallással az akciós potenciál sablonok elhelyezésére – Tézispont: I/3, Fejezet: 2, Cikkek: [6]
- Egy FPGA architektúra, amely képes valós időben neurofiziológiai jelek 128 csatornás elektróda struktúrára történő generálására a Mersenne Twister pszeudo-véletlenszám generáló algoritmus használatával – Tézispont: I/4, Fejezet: 2, Cikkek: [6]
- Többcsatornás neurofiziológiás felvételeken a csatornák közötti térbeli információk felhasználásával az OSort algoritmus hatékonyságának növelése, valamint validációja – Tézispont: II/1, Fejezet: 3, Cikkek: [2], [4]
- Az OSort algoritmuson alapuló többcsatornás neurofiziológiai felvételeket valós időben osztályozni képes FPGA architektúra és validációja – Tézispont: II/2, Fejezet: 3, Cikkek: [2], [4]
- Egy új, párhuzamos, költséghatékony Principal Component Analysis (PCA) megvalósító FPGA architektúra – Tézispont: III/1, Fejezet: 4, Cikkek: [5]
- Egy új, párhuzamos, költséghatékony FPGA architektúra, amely több szenzort és azok méréseit felhasználva képes valós időben szenzorfúzióval pozíciót és orientációt becsülni, valamint annak egy beltéri lokalizációs kiegészítése, amely lehetővé teszi az abszolút pozícióra vonatkozó információk beépítését a kidolgozott szenzorfúziós architektúrába – Tézispont: III/2, Fejezet: 4, Cikkek: [1], [8]
- Egy kétdimenziós multilaterációs FPGA architektúra – Tézispont: III/2, Fejezet: 4, Cikkek: -
- Egy megbízható, költséghatékony módszer, amely minimális számú szenzor felhasználásával hatékonyan szabályozza egy gépjármű aktív felfüggesztő rendszerét és egy új párhuzamos FPGA architektúra, amely az aktuális megfigyelő alapján valós időben szabályozza az aktív felfüggesztés erő nagyságát – Tézispont: III/3, Fejezet: 4, Cikkek: [7]
- Egy szélturbinát emuláló, új, párhuzamos FPGA architektúra, amelynek dinamikusan változtatni lehet a bemeneti paramétereit egy valós idejű hardware-in-the-loop (HIL) rendszerben SCADA interfészen keresztül – Tézispont: III/4, Fejezet: 4, Cikkek: [3]

Ezek az eredmények **Schäffer László** PhD disszertációján kívül más tudományos fokozat megszerzésére nem használhatók fel.

Szeged, 2021.10.27 Schäffer László PLETL SZILVESZTER KINCSES ZOLTÁN

dátum jelölt aláírása témavezető aláírása társ témavezető aláírása

Az Informatika Doktori Iskola vezetője kijelenti, hogy jelen nyilatkozatot minden társszerzőhöz eljuttatta, és azzal szemben egyetlen társszerző sem emelt kifogást.

2021/11/03

dátum DI vezető aláírás