An AER-Based Actuator Interface for Controlling an Anthropomorphic Robotic Hand

A. Linares-Barranco[1], A. Jiménez-Fernandez[1], R. Paz-Vicente[1], S. Varona[2], and G. Jiménez[1]

[1] Departamento de Arquitectura y Tecnología de Computadores. Universidad de Sevilla

[2] Lab. of Neurotechnology, Control and Robotics (NEUROCOR). Technical Univ. of Cartagena. Campus Muralla del Mar

Abstract. Bio-Inspired and Neuro-Inspired systems or circuits are a relatively novel approaches to solve real problems by mimicking the biology in its efficient solutions. Robotic also tries to mimic the biology and more particularly the human body structure and efficiency of the muscles, bones, articulations, etc. Address-Event-Representation (AER) is a communication protocol for transferring asynchronous events between VLSI chips, originally developed for neuro-inspired processing systems (for example, image processing). Such systems may consist of a complicated hierarchical structure with many chips that transmit data among them in real time, while performing some processing (for example, convolutions). The information transmitted is a sequence of spikes coded using high speed digital buses. These multi-layer and multi-chip AER systems perform actually not only image processing, but also audio processing, filtering, learning, locomotion, etc. This paper present an AER interface for controlling an anthropomorphic robotic hand with a neuro-inspired system.

1 Introduction

The Address-Event Representation (AER) was proposed by the Mead lab in 1991 [1] for communicating between neuromorphic chips with spikes (Fig. 1). Each time a cell on a sender device generates a spike, it communicates with the array periphery and a digital word representing a code or address for that pixel is placed on the external inter-chip digital bus (the AER bus). Additional handshaking lines (Acknowledge and Request) are used for completing the asynchronous communication. In the receiver chip the spikes are directed to the pixels whose code or address was on the bus. In this way, cells with the same address in the emitter and receiver chips are virtually connected by streams of spikes.

These spikes can be used to communicate analog information using a rate code, but this is not a requirement. Cells that are more active access the bus more frequently than those less active. Arbitration circuits usually ensure that cells do not simultaneously access the bus. Usually these AER circuits are built using self-timed asynchronous logic by e.g. Boahen [2].

Transmitting the cell addresses allows performing extra operations on the events while they travel from one chip to another. For example the output of a silicon retina can be easily translated, scaled, or rotated by simple mapping operations on the emitted addresses. These mapping can either be lookup-based (using, e.g. an EEPROM) or algorithmic. Furthermore, the events transmitted by one chip can be received by many receiver chips in parallel, by properly handling the asynchronous communication protocol. There is a growing community of AER protocol users for bio-inspired applications in vision, audition systems and robot control, as demonstrated by the success in the last years of the AER group at the Neuromorphic Engineering Workshop series [4]. The goal of this community is to build large multi-chip and multi-layer hierarchically structured systems capable of performing massively-parallel data-driven processing in real time [3].

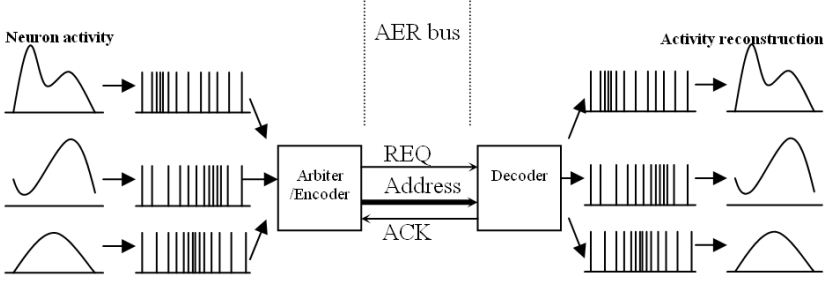

Fig. 1. Rate-coded AER inter-chip communication scheme. REQ is the Request line, ACK is the Acknoledge line a the bus Address hold the neuron code.

The neuromorphic approach of AER can be also applied to actuators, like the muscles in the biology. The success of such systems will strongly depend on the availability of robust and efficient development, debugging and interfacing AER-tools [7][8][9][10][11]. One such tool is a computer interface that allows not only reading a sequence of events with their timestamps, but also reproduces a sequence of events stored in the computer's memory.

From the robotic point of view a very interesting tool will allow the connection between an AER system and a robot. This paper presents, describes and tests the improved AER-Robot [7] interface for this connectivity. This tool is able to interface to actuators and commercial sensors (position, contact, pressure, temperature, ...) to allow movements and feedback allowing a more complex and bio-inspired control of a robot. Depending on the application the interface should be able to: (a) Implement the control algorithm into the interface, using

an FPGA or a Microprocessor, or (b) to transmit the sensor information that is receiving to an AER chip or to a computer in AER format, and the opposite, to receive orders format from an AER chip or from a computer, to the actuators in AER, allowing the computer or the AER chip to implement the control algorithm.

Factorization of Length and Tension (FLETE) is a bio-inspired control mechanism for robots that computes not only the position of the robot, but also the rigidity of it. In this case the visual information is insufficient, and another kind of mechanical sensor is needed.

With these interfaces you can control a robotic platform using an AER system to give it orders and obtain other kind of sensory information (pressure, contact, position, etc) into AER format. Furthermore, the AER interface allows debugging the robotic platform if you connect it to the computer using the PCI-AER interface or the mini-USB-AER [10].

In this paper we present an improved AER Interface to connect the AER system to a set of actuators (motors) and sensors, called AER-Robot. The interface has been used to connect a computer to the TUC[1] anthropomorphic robotic hand in order to enable an AER system to control a complex and bio-inspired robot. The hand is driven by an agonist-antagonist opponent system. In order to measure joint position, velocity, and direction of rotation, hall-effect position sensors were integrated at each joint of the fingers. Force sensors are mounted on the curved surface of the fingertips, and on the palm.

In the following sections we describe the anthropomorphic hand, the AER-Robot interface, the PCI-AER interface, the mini-USB-AER interface to the computer for debugging purpose, and the results.

2 Anthropomorphic Robotic Hand

The TUC anthropomorphic robot design hand is based on the biomechanics modelling of the human hand, as well as on designs by manufacturers of robot hands. The hand has three fingers and an opposing thumb, and four degrees of freedom for each finger. The fingers are mounted on a rigid palm. Fig. 2 shows one photography of the TUC Hand, placed over a industrial robot for grasping, reaching and handling tasks.

The design of the multi-jointed finger presents three joints (metacarpophalangeal (MCP), proximal interphalangeal (PIP), and distal interphalangeal (DIP) joints, respectively), where DIP and PIP are coupled. Both the PIP and DIP joints have flexion and extension, and the MCP joint consist of two joints that allow flexion-extension and adduction-abduction motions. In the finger design, the muscle-like actuators are DC motors. Each joint of the finger is actuated through 2 polystyrene tendons, routed through pulleys and driven by DC motors. The joints are moved by an agonist-antagonist opponent system. In order to measure joint position, velocity, and direction of rotation, hall-effect position sensors were integrated at each joint of the fingers. Tactile sensors based

[1] Technical University of Cartagena.

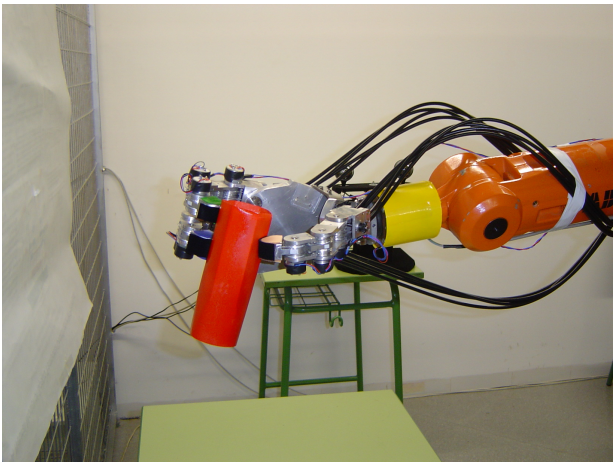

Fig. 2. TUC Robot Hand

on FSR (Force Resistive sensing) technology are mounted on all the joints and on the palm emulating artificial tactile surfaces. The flexibility of these sensors is very suitable for the implementation on the curved surface of the fingertips for precision grasping and manipulations tasks. One two-axis sensor placed on the palm is employed to correct the stability of the gross grasping. It permits the tactile guided for the movement of the wrist of the robot hand-arm [12]. Each one of the fingers that conforms the biomechanical hand is driven by a mechanism constituted by an assembly of pulleys that control the movements of the different phalanges. Each finger is comprised of three articulations with possibility of turn and an additional articulation that permits to reproduce the movement of abduction, besides serving of element of union of the digit with the palm of the hand. The pulleys (on articulations) are driven for a system of cables to way of human tendons. Each articulation possesses two tendons, one flexor and another extensor. The tendon flexor causes the movement of contraction of the articulations while the tendon extensor causes the contrary effect. The mechanical system of actuation arranges of a motor to extend and another to contract the tendon. For control the turn of each articulation, in a synchronized way, the wires remains traction in every instant, and is possible to measure the effort done by the tendons.

3 AER-Robot Interface

This section describes in detail an AER interface to manage actuators and to read analog commercial sensors and convert it to AER format. These actuators are based on DC motors.

The AER-Robot interface can control up to 4 up/down DC motors, each one doted with a two channel encoder. The DC motors are controlled digitally using

Pulse Width Modulation (PWM). AER-Robot can read up to the following sensors: (a) 4 potentiometers for the finger articulations position, (b) 4 contact resistors for the fingertip and the palm object detection, (c) 4 tension sensor for the tendons of the fingers, and (d) 4 current sensor for the power consumption of the motors of the fingers. These sensors information are fundamental for the control algorithms in the hand platform.

The AER-Robot interface has been developed to communicate AER systems with an anthropomorphic robotic hand using two AER buses: one for incoming commands and another for outgoing information of the motors and the sensors. Furthermore, the interface allow the serial connection of several AER-Robot or other AER based components, thanks to other 2 AER busses to (a) pass through the incoming event and (b) to receive events and arbitrate them with the ones produced by itself to send them out. It is based around a Spartan 3 400 FPGA that allows co-processing. This FPGA receives commands through the input AER bus and sends motor and sensor information back. These commands allows to:

- Configure the PWM period that manages all the motors.
- Move a motor attending to PWM intensity and an estimated position through the encoder's information.
- Ask for a motor state.
- Ask for a sensor state.

Fig. 3 shows the block diagram of the circuit of the FPGA, described into VHDL. This circuit is composed by several processes. CMDin receives commands and sends them to the corresponding process. There are 4 independent processes (Motor i) to control de PWM signal to be sent to each motor. There is a process to attend the 16 possible sensors of one finger of the hand (4 potentiometers for articulations positions, 4 contact, 4 hall effect current consumption of the motor and 4 tension tendon sensors). This last process attends to a Cygnal microcontroller that is continuously converting an analog signal to digital from 16 possible analog inputs.

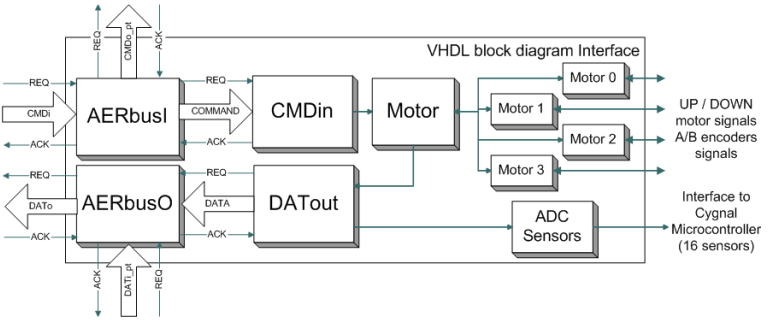

Fig. 3. Circuit block diagram of the FPGA

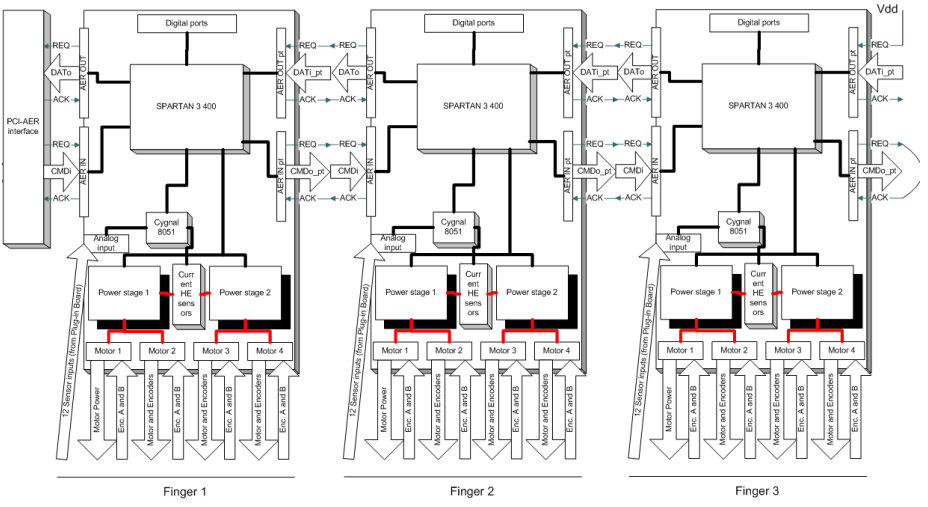

Fig. 4. AER-Robot block diagram interface

These processes work in parallel to allow real-time control of the hand; therefore the interface can receive new commands while it is executing previous ones. There is another process for the AER output bus traffic control (DATout), and another one for the four AER buses control.

For each input command received by the AERin process, the order is sent to the corresponding motor process or sensor process, and then this AERin process is free to attend a new command.

Each motor process is in charge of one motor. If this process receives an order, its motor will go up or down, for a number of encoder pulses and with a programmed intensity.

There is a sensors process that is asked for a value of one the sensors. This process keep updated a 16-address internal RAM memory with the digital value of their sensors, and the digital value is sent to the AERout process when the AERin process received a read-sensor command.

To keep this RAM-table updated, the sensor process communicates with a microcontroller, outside the FPGA, that scan the 16 analog output of the sensors, convert it into 8-bit and send it to the FPGA. The RAM-table is updated every 184μs. Thus, when a command is asking for the value of one sensor, the sensor process doesn't ask it to the microcontroller, but it just has to read it from the internal RAM memory of the FPGA (1 clock cycle).

The microcontroller of the AER-Robot is in charge of 16 sensors of 4 different sets: articulations potentiometers, fingertip and palm contacts, tendon tension and power consumption of the motors.

Fig. 4 shows the block diagram of the interface. The real-time is warranted by the independent process architecture.

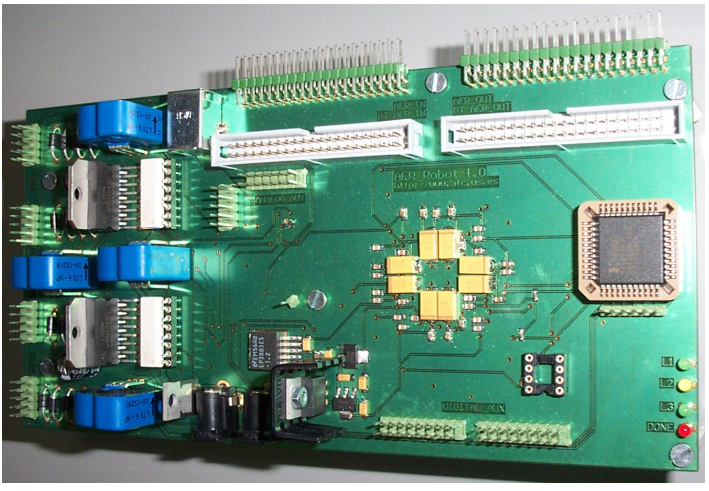

Fig. 5. AER-Robot board photograph

Fig. 5 shows a photograph of the prototype of the AER-Robot Interface PCB. The digital part of the PCB is on the left side. The board has 4 power stages for the 4 motors, 4 Hall Effect sensors for the power consumption measurement of the motors and a Cygnal 80C51F320 microcontroller for the analog to digital conversion (200Ksamples/second and 10-bits) of the sensor measurements, and all the connectors to the Hand and to the AER systems. A small board, called Plug-in board, was connected to the 12 line of the microcontroller to implement the analog adaptation of the output of the different sensor to the voltage needed to convert it into digital by the microcontroller. This plug-in board was composed by 4 x10K amplifiers for the tendon sensors and 8 direct connections to the connectors of the contact sensors and potentiometers sensor installed on the hand,

The AER input bus and AER output bus is connected to a PC using the PCI-AER interface, explained in the next section.

4 PCI-AER Interface

Before the development of our tools the only available PCI-AER interface board was developed by Dante at ISS-Rome (See [4]). This board is very interesting as it embeds all the requirements mentioned above: AER generation, remapping and monitoring. Anyhow its performance is limited to 1Mevent/s approximately. In realistic experiments software overheads reduce this value even further. In many cases these values are acceptable but, currently many address event chips can produce (or accept) much higher spike rates.

As the Computer interfacing elements are mainly a monitoring and testing feature in many address event systems, the instruments used for these proposes

should not delay the neuromorphic chips in the system. Thus, speed requirements are at least 10 times higher than those of the original PCI-AER board. Several alternatives are possible to meet these goals:

- Extended PCI buses.
- Bus mastering.
- Hardware based Frame to AER and AER to Frame conversion.

The previously available PCI-AER board uses polled I/O to transfer data to and from the board. This is possibly the main limiting factor on its performance. To increase PCI bus mastering is the only alternative. The hardware and driver architecture of a bus mastering capable board is significantly different, and more complex, than a polling or interrupt based implementation.

A well designed AER system, which produces events only when meaningful information is available, can be very efficient but, an AER monitoring system should be prepared to support the bandwidth levels that can be found in some real systems. These include systems that have not been designed carefully or that are under adjustment. Currently the available spike rates, even in these cases, are far from the value shown above but, some current AER chips may exceed the 40Mevents/s in extreme conditions.

The theoretical maximum PCI32/33 bandwidth is around 133Mbytes/s. This would allow for approximately 44Mevent/s considering 2 bytes per address and two bytes for timing information. Realistic figures in practice are closer to 32Mbyte/s. Thus, in those cases where the required throughput is higher a possible solution is to transmit the received information by hardware based conversion to/from a frame based representation. Although this solution is adequate in many cases, there are circumstances where the developers want to know precisely the timing of each event, thus both alternatives should be preserved.

Implementing AER to Frame conversion is a relatively simple task as it basically requires counting the events over the frame period. Producing AER from a frame representation is not trivial and several conversion methods have been proposed [8][9].

The physical implementation of all the steps is equal. They differ in the VHDL FPGA code and in the operating system dependant driver. This interface is based on SPARTAN-II family. The board is shown in Fig. 6.

Currently a Windows driver that implements bus mastering is available. An API that is compatible, as much as permitted by the different functionality, with that used in the current PCI-AER board has been implemented. MEX files to control the board from MATLAB have also been developed.

The final goal is to transmit an AER sequence to an AER based system (for example a convolution chip) to perform video processing. An adequate sequence of events can be generated by software for testing an AER based system. This sequence of events needs to be sent to the AER based system. For this purpose it is necessary an interface between the computer and the AER bus. This PCI interface has been developed under the European project CAVIAR. The interface, called CAVIAR PIC-AER, has two operation modes that can work in parallel:

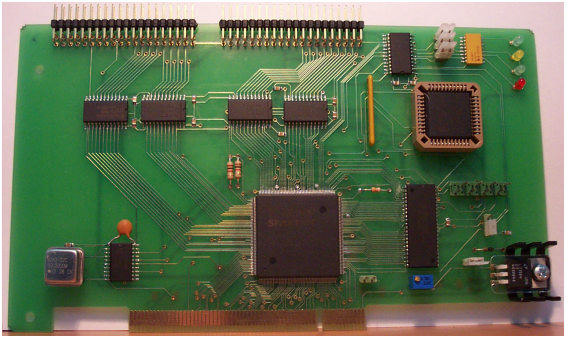

Fig. 6. CAVIAR PCI-AER board

A) From PCI to AER
The AER-stream is stored in the computer memory and then it is sent to the AER system through the PCI bus and the OFIFO. This stream is saved in memory using 32 bits for each address event. The sixteen less significant bits represents the address of the pixel that is emitting the event. The sixteen more significant bits represent a time difference from the previous event in clock cycles. The delay between events is specially treated, so a delay in the ACK reception will not cause a distortion in the time distribution of all the events along the time period.

B) From AER to PCI
The AER sequence arrives to the CAVIAR PCI-AER interface through the input AER port. The AER-IN state machine stores the incoming event (16 bits LSbits) into the IFIFO with temporal information. This temporal information (16 bits MSbits) is the number of clock cycles since the last event.

The connection to the PCI bus is done by a VHDL bridge [13] that attends to the Plug & Play protocol of the PCI bus, decodes the access to the base address by the operating system, allows the bus mastering and the interruptions.

5 Results

Both the hand and the AER interfaces have been connected to debug control algorithms algorithm implemented under Matlab.

The Antrophomorphic robotic hand was developed by NEUROCOR group from TUC (Spain). Under the Spanish grant project SAMANTA the hand has been connected to an AER system. One of the objectives of the project was to control the hand using AER vision systems together with other sensor information. Therefore the hand needs to be controlled under the AER protocol. In that project the visual information comes from an AER retina. Thanks to the PCI-AER interface the visual information is sent to a personal computer. A boundary-contour-system feature-control-system (BCS-FCS) algorithm for

image processing was implemented under Matlab. With the visual processing results and the hand sensors information, the control algorithm gives orders to the hand, for example to catch an object, by computing not only the visual information, but also the rigidity and fatigue of the muscles.

The results shown in table 1 are the ranges of the parameters that can be programmed in the AER-Robot board. These results have been measured under the following scenario: three AER-Robot interfaces have been connected together using AER busses in a chain configuration. The first AER-Robot interface was connected to a PC through the PCI-AER interface. Therefore, commands where sent from Matlab to the PCI-AER interface into AER format, and then they are sent to the first AER-Robot interface, this one retransmit the command to the second, the second to the third and so on. The AER information that produces a board, this is sent through an AER bus. This one can be connected to the data input AER bus of the previous board to the PCI-AER. Each AER-Robot arbitrates the AER information received from the next board with the information that is produces.

Table 1. Measured ranges for DC motor and Sensors

	Freq min	***Freq max***	***High res.***	
PWM	763 Hz	25 MHz	8-bits	
	Dig. Res.	***Range (volts)***	***Kind***	***Sensor range***
Potentiometers	8-bit	0-3,3v	R	
Contact	8-bit	0-3,3v	R	
Tension	8-bit	2-2,5v	R	0,1mv/V
Hall-Effect current	8-bit	2-3,3v	L	
PCI-AER	***Max Th.***	***Pulse width***		
	6 Mev/s	120 ns		
AER-Robot	3 Mev/s	240 ns		

6 Conclusions and Future Work

An AER to anthropomorphic robotic hand interface that can be connected to an AER system or to a PC through a PCI-AER interface have been presented. The AER-Robot interface is based around a Spartan 3 FPGA that allows it to be configurable and easily modified for other robotic applications based on DC motors, encoders and sensors.

The AER neuro-inspired communication channel is connected with the robot. This implies a neuro-inspired control of the robot. This control is based on visual processing using AER retinas and convolution chips, and neuro-inspired FLETE algorithm in software.

The current interface is able to receive and send 16-bit AER data. Coded under these 16-bit is placed the command or the sensor information. Therefore one event is enough to send a command to a motor or ask for sensor information,

then one or two events are sent back with the information required. The future work is focused on the spike based information. In such way, the motor PWM frequency will be sent translating the frequency of one address in the AER bus. Each address corresponds with one motor, and with one sensor in the other way. These VHDL improvements are under development.

Acknowledgements

This work was in part supported by EU project IST-2001-34124 (CAVIAR), Spanish projects TIC-2003-08164-C03-02 (SAMANTA) and SAMANTA II. We also wanted to thank to J.L. Pedreño-Molina, J. Molina-Vilaplana for their contribution in this work with their bio-insipired control algorithms.

References

[1] M. Sivilotti, Wiring Considerations in analog VLSI Systems with Application to Field-Programmable Networks, Ph.D. Thesis, California Institute of Technology, Pasadena CA, 1991.
[2] Kwabena A. Boahen. "Communicating Neuronal Ensembles between Neuromorphic Chips". Neuromorphic Systems. Kluwer Academic Publishers, Boston 1998.
[3] R. Serrano-Gotarredona et al. AER Building Blocks for Multi-Layer Multi-Chip Neuromorphic Vision Systems. NIPS 2005. Vancouver.
[4] A. Cohen, R. Douglas, C. Koch, T. Sejnowski, S. Shamma, T. Horiuchi, and G. Indiveri, "Report to the National Science Foundation: Workshop on Neuromorphic Engineering", Telluride, Colorado, USA, June-July 2001. [www.ini.unizh.ch/telluride]
[5] Avis Cohen, et.al., Report on the 2004 Workshop On Neuromorphic Engineering, Telluride, CO. June - July , 2004 [www.ini.unizh.ch/telluride/previous/report04.pdf]
[6] A. Wilen, J.Schade, R. Thornburg. "Introduction to PCI Express", Intel Press, 2003.
[7] A. Linares-Barranco et al.. "AER Neuro-Inspired interface to Anthropomorphic Robotic Hand". IEEE World Conference on Computational Intelligence. International Join Conference on Cellular Neural Networks. Vancouver, July-2006.
[8] A. Linares-Barranco, G. Jimenez-Moreno, A. Civit-Ballcels, and B. Linares-Barranco. "On Algorithmic Rate-Coded AER Generation". *IEEE Transaction on Neural Networks.* May-2006.
[9] A. Linares-Barranco, M. Oster, D. Cascado, G. Jimenez, A. Civit, B. Linares-Barranco. "Inter-Spike-Intervals analysis of AER Poisson like Generator Hardware". Accepted for publication on Neurocomputing (∼December 2007).
[10] R. Paz, F. Gomez-Rodriguez, M. A. Rodriguez, A. Linares-Barranco, G. Jimenez, A. Civit. Test Infrastructure for Address-Event-Representation Communications. IWANN 2005. LNCS 3512. pp 518-526. Springer Verlag.
[11] Address Event Representation tools. [http://www.atc.us.es/AERtools]
[12] J. López-Coronado, J.L. Pedreño-Molina, A. Guerrero-González, P. Gorce. A neural model for visual-tactile-motor integration in robotic reaching and grasping tasks. Robotica, Volume 20, Issue 01, pp. 23-21. Cambridge Press. (January 2002).
[13] R. Paz. "Análisis del bus PCI. Desarrollo de puentes basados en FPGA para placas PCI". Trabajo de investigación para obtención de suficiencia investigadora. Sevilla, Junio 2003.