An AER to CAN Bridge for Spike-Based Robot Control

[1] Robotic and Technology of Computers Lab. University of Seville, Spain
mdominguez@atc.us.es
[2] NEUROCOR Lab.Technical University of Cartagena, Spain

Abstract. Address-Event-Representation (AER) is a bio-inspired communication protocol between chips. A set of AER sensors (retina and cochleas), processors (convolvers, WTA, mappers, …) and actuators can be found in the literature that have been specifically designed for mimicking the communication principle in the brain: spikes. The problem when developing complex robots based on AER (or spikes) is to command actuators (motors) directly with spikes. Commercial robots are usually based on commercial standards (CAN) that do not allow powering actuators directly with spikes. This paper presents a co-design FPGA and embedded computer system that implements a bridge between these two protocols: CAN and AER. The bridge has been analyzed under the Spanish project VULCANO[1] with an arm robot and a Shadow anthropomorphic hand.

Keywords: CAN, AER, spike-based, neuromorphic engineering, anthropomorphic robot, FPGA, VHDL, embedded computer.

1 Introduction

Controller Area Network (CAN) is an industrial protocol that minimizes the number of unrecovered transmission errors using a set of mechanism for detecting and signaling errors. This protocol is so efficient that it is usually used at automotive, robots and mobile robots for communicating sensors and actuator to a central processor. CAN is nowadays a typical interface for commercial robots, motors or even sensors. On the other hand Address-Event-Representation (AER) is a neuro-inspired communication protocol for transferring information between silicon neurons in different chips. Neuromorphic engineers work actively in developing sensors, processors and actuators that mimic the nervous system behavior.

Neuroinformatic aims to join together several field specialists (biologists, psychologists, engineers, physicists, chemists, and informatics) in order to develop auto-reconfigurable systems that mimic the human body and specially emulate the human brain. Neuromorphic engineers work in the study, design and development of neuro-inspired artifacts developed with artificial mechanisms, like VLSI chips for sensors [1][2][3], neuro-inspired processing, filtering or learning [4][5][6], neuro-inspired control-pattern-generators (CPG) [7], neuro-inspired robotics [8] and so on.

[1] This work has been supported by Spanish goverment grant VULCANO (TEC2009-10639-C04-02).

All these mechanisms share the way of producing, transforming and transferring the information: the spike-based representation of the information, like in a mammal neural system. A neuromorphic VLSI chip is designed in order to gather several thousands of silicon spike-based neurons. A problem arises when those neurons have to communicate outside the chip with other neurons of a different chip. Typically, neuromorphic engineers have adopted the so-called solution Address-Event-Representation (AER). AER was proposed by the Mead lab in 1991 [9] for communicating between neuromorphic chips with spikes (Figure 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). In the receiver chip, 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. Cells that are more active access the bus more frequently than those less active. 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 [7] and the CapoCaccia Cognitive Neuromorphic Engineering Workshop [10]. The goal of this community is to build large multichip and multi-layer hierarchically structured systems capable of performing massively-parallel data-driven processing in real time.

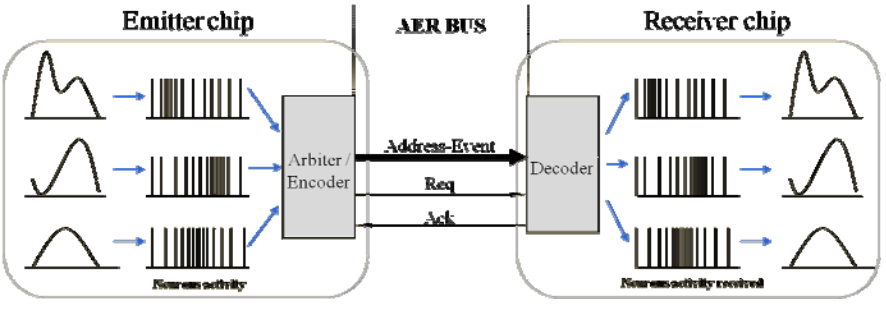


Fig. 1. Rate-coded AER inter-chip communication scheme

Furthermore, the application of these systems to real applications demonstrate the benefits of the spike-based representation, allowing extremely low latencies between spikes produced by a sensor and the first spike arriving to the actuator (in the order of few microseconds), as demonstrated in the EU project CAVIAR [11].

The implementation of spike-based Proportional-Integral-Derivative (PID) controller allows to complete the last stage of a neuromorphic system, by powering the equivalent of a muscle in a robot directly with spikes [8] (a DC motor).

Nevertheless, commercial robots are usually black-boxes consisting in a set of motors or actuators, and sensors with a common interface that allows communicating with a computer that is usually in charge of solving the cinematic and dynamic equations needed to implement tasks like grasping, tracking, manipulating an object, etc. One of these common interfaces is the Controller Area Network (CAN) bus. The CAN bus was developed by Robert Bosch GmbH company for communicating

messages in a distributed environment ensuring real-time capabilities and using only two wires. The number of nodes, the distance between them, the bus bandwidth, the error detection and correction mechanism, the arbitration policy and other aspects of the CAN protocol can be found in [12].

In this paper we present a hardware-software bridge between the AER bus and the CAN bus. The bridge has been implemented using the hardware platforms developed under Spanish Government granted VULCANO project. These platforms have been designed for covering the implementation of more complex spike based operations. VULCANO has the aim of developing and joining together a set of AER chips (retina sensors and filters) into a layer of image filtering. The result of this layer is fussed by a second layer implementing a set of brain-inspired algorithms for sensory fusion and visual-motor coordination. The implementation of this second layer is based on a co-design platform composed by a FPGA and an embedded computer (Toradex Colibrí). This embedded computer provides a CAN interface. Arm and hand robots used in VULCANO offer CAN interfaces for motor control and sensors monitoring.

Section II offers a brief review of the most important characteristics of the CAN protocol. Section III explains the scenario in VULCANO project. Then, in section IV we describe the architecture of the AER-CAN bridge and we present a performance study.

2 Controller Area Network (CAN)

CAN physical layer establishes that the information is represented by the voltage difference between two wires. Logical bits are called dominant (‘0’) when the voltage difference between the two wires is high and recessive (‘1’) when they are similar. The network is composed by a set of nodes sharing a unique pair of wires (CAN bus). When several nodes write at the same time on the bus, if only one node writes a dominant bit, then the CAN bus will conserve the dominant bit, and if all the nodes write recessive bits, the CAN bus conserves the recessive value.

Each node connected to the CAN bus is always monitoring the bus, even when a node is transmitting. If several nodes are transmitting at the same time, a node can lose the CAN bus if it monitors a dominant bit when it is transmitting a recessive bit.

Distributed nodes are free to use the bus when a message is finished. Nodes do not need to have an arbiter selecting which node is able to transmit each time, because the protocol is auto-arbitrated thanks to the characteristic of dominant and recessive bits. All the messages are transmitted starting with a start of frame bit (SOF) and then, the identifier of the message. This identifier is used for arbitrating which node will use the CAN bus. When a node is writing its identifier on the CAN bus, if a recessive bit is written (‘1’) and a dominant bit (‘0’) appears on the bus, this node has lost the competition and it has to wait until the winner node finished its message transmission. Figure 2 bottom shows a competition example. A message includes protocol information, a variable number of data bytes from 0 to 8, followed by a 16-bit Cyclic Redundancy Code (CRC) for error detections (see Figure 2 top).

The medium access and transport protocol layers establish that when a node is transmitting a message, the rest of the nodes are checking the message. Therefore, in the CAN protocol non-transmitter nodes are responsible of the correctness and the

efficiency of message transmissions by monitoring all the bits of a message and collaborating in the ACK bit of a message when an error is found. Even more, the transmitter node is also monitoring the CAN bus during the transmission of its own message and it is also checking if each bit written in the bus is correctly read: if the transmitter node writes a recessive bit in the bus, it has to read a recessive bit during the rest of the time that this bit must appear on the bus.

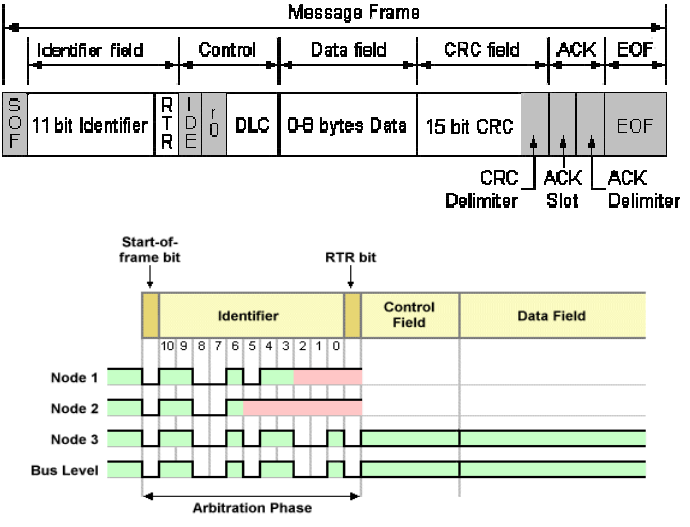


Fig. 2. CAN message (top) and priority arbitration example (bottom)

The maximum speed of the CAN standard is 1 Mega bit per second (Mbps) for up to 40 meters of cable distance between the furthest nodes. This implies one microsecond per bit. Every bit time is divided in several parts: synchronization, propagation (400 ns for 40 meters), segment one and segment two. Synchronization part is very small (usually one clock cycle of the CAN controller clock). The propagation part must be preserved in order to ensure the correct reception of the signal in both sides of a 40 meters cable. Finally segments 1 and 2 complete the bit time up to 1 microsecond. Receivers will check the state of the CAN signal between segment 1 and 2. These segments are equal to a setup and hold time of digital circuits.

CAN standard has several similarities with AER protocol:

- Both are event-based protocols. AER sensors or actuators produce either a stream of events that represents the sensor value, or its change, or the actuator intensity or change. CAN nodes codify in a unique message the sensor value or the command parameters.
- Both use addresses for identifying messages. When using AER, an emitter chip uses a set of addresses depending on the number of emitter neurons. In CAN, each node can work (transmit or receive) a set of messages with different identifiers. An identifier in CAN is not used for identifying neither an emitter nor a receiver; they are used to identify messages produced by sensors or received by actuators.

On the other hand there are several important differences:

- CAN is a serial and synchronous bus, while AER is typically parallel and asynchronous.
- CAN maximum speed is 1 Mbps. This is considerably low respect to AER that can work at peak rates higher than 25 Mega-events-per-second (400 Mbps for 16-bit buses)[4]. Nevertheless there are improvements to the CAN protocol that reach up to 16Mbps (CAN+) [13].
- CAN includes in the message transmission additional information for increasing the robustness. AER transmits the address of a neuron without any other complementary information.
- AER codify the information in frequency or by the number of repetitions of the same address in the bus, while CAN send a unique message with the command and value in up to 8 bytes.

For those applications with low speed requirements there is a reduced number of neurons or identifiers and scenarios with strong noise interferences, so it could be adequate to inherit the properties of the CAN bus in order to implement a serial and robust AER protocol. For example, the spike motor actuation and the motor sensor monitoring. An AER system is able to process visual information in a very fast way and with a low latency. The AER system can produce a stream of spikes for actuating into a set of muscles (or motors in robotic). Since the motor is a mechanical object, it has a huge delay in implementing the position or velocity orders, when compared to the AER processing system. Due to this limitation of the motors, it is justified to include an AER-CAN interface without losing performance.

3 Actuators and Sensors through CAN

VULCANO focuses on the AER for developing a sensory-motor system completely based on spikes. From sensing and filtering the visual information, developing AER retina and AER convolvers, to the adaptation and development of anthropomorphic robots (hand and arm) and their interfaces, through the development of high level algorithms for sensory fusion, visual-motor coordination and the cinematic and dynamic of the robots.

The robotic arm consists into a set of articulations based on commercial motors. Each of these motors is mounted with a controller that receives CAN messages with commands. These commands can be either a) new positions or degree for the articulation, with a fixed intensity and speed, or b) a request for the value of a sensor, like the position of the articulation, the intensity of the motor, the speed, …

Existing AER based controllers for DC motors [8] requires to access directly to the motor and to receive directly the information of the sensors in order to adapt the frequency of spikes to be sent to the motor for modifying the position, speed or power of the motor. This kind of AER controller cannot be used in such a commercial platform because the robot will lose all the robustness and efficiency that the CAN is providing (apart from the warranty of the product). Therefore, in this case it is very

important and necessary to develop an AER-CAN interface able to receive streams of spikes from the last layer of the AER systems and convert it into CAN commands, and vice versa, to convert CAN sensory messages into a stream of spikes for the corresponding layer of the AER system.

4 CAN Bridge Architecture and Performance

Figure 5 shows a block diagram of a co-design prototype for implementing the sensory fusion, visual-motor coordination and kinematic/dynamic algorithms of robots in VULCANO. It consists in a Xilinx Spartan 3 400 FPGA connected to an embedded computer (Toradex Colibri with a Intel Xscale PXA270 under Windows CE) through Direct Memory Access interface (DMA). The FPGA is responsible for receiving, in the AER communication, all the sensing information coming from the cortex layer (AER retina + convolver) and spinal cord sensing (the robot sensors in AER format). This sensor information can be either 1) fussed in the FPGA and stored, through DMA, in the computer DRAM, or 2) packed into a sequence of time-stamped events in the DRAM computer memory, through DMA. Then, the embedded computer will use this information for executing the sensory fusion and visual-motor coordination. In the first case, the computer program can work in a completely digital way, being the spike-based approach only resident in the FPGA. The computer receives digital values of the sensors, so a spike to digital conversion is done. In the second case, both the FPGA and the embedded computer are working with spikes, so the system takes all the advantages of a spike-based system, having very low latencies from sensors to actuators. DMA has a maximum bandwidth of 101'72 MBps between the Spartan 3 and the Toradex computer. This bandwidth allows transmitting packetized AER 16-bit events and their 16-bit timestamps at a regular rate of 25 Mevps that is the maximum throughput of the USB-AER hardware interface [14].

Since the sensory fusion algorithm is executed in the embedded computer and the CAN protocol is accessible through the Intel XScale board, we have decided to access the CAN bus through the embedded computer and offer the additional sensor information directly to the sensory fusion algorithm. On the other hand, the visual-motor coordination is also executed in the embedded computer, so it is faster and more adequate to send the CAN motor commands directly from the embedded computer.

Each node of the arm-robot consists in a Schunk motor and a PowerCube communication interface in CAN mode. There are six nodes connected through CAN to the Toradex embedded computer. For each message received by the PowerCube interface of a motor node, a command is executed in the node. During the execution of that command, an acknowledgment message (ACK) is sent back. If a new command is received while the previous one is being executed, the old one is aborted and the new one is executed. These events allow the system to correct trajectories at the same time they are improved.

We have analyzed the CAN traffic in order to extract the typical and maximum bandwidth of commands that is supported. As can be seen in figure 3, there is a

minimum Inter-Message-Interval (IMI) of 320µs that represents a very fast update time for motor commands compared to the response time of such a mechanical actuator. This time has been obtained when the commands are sent sequentially (a new command is not sent until the reception of the ACK of the previous one). A motor command of 6 data bytes requires 110µs to be transmitted and 120µs to receive and process it through PowerCube. Then it responds with an ACK message (2 data bytes) that requires 78 µs. Finally there is a 12 µs pause before the next motor command appears in the CAN bus. With these data the performance of the system is a bandwidth of 3,125 K commands per second (Kcps). In Figure 4 (right), it is shown a performance graph considering different command lengths (from 1 to 6 bytes) and two different communication mechanisms: 1) Sequentially and 2) Interlaced (the sparse time between the command CAN message and the ACK is used to send a second command message to another motor, as represented in figure 4 left). The maximum bandwidth that can be extracted from the AER-CAN interface is 5Kcps.

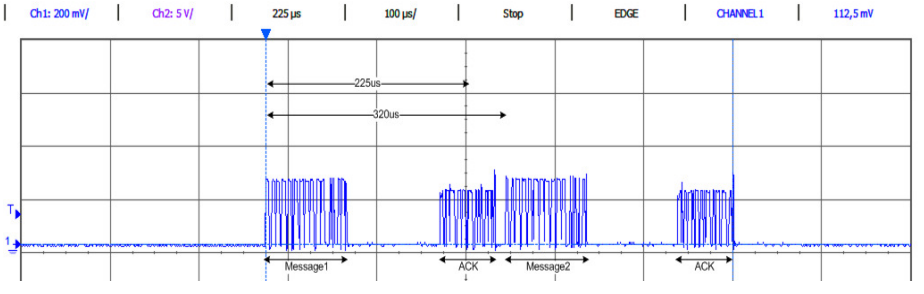


Fig. 3. Oscilloscope capture of CANRX over time during two CAN message transmissions (6 data bytes) for different motors. Acknowledge CAN messages are sent back (2 data bytes).

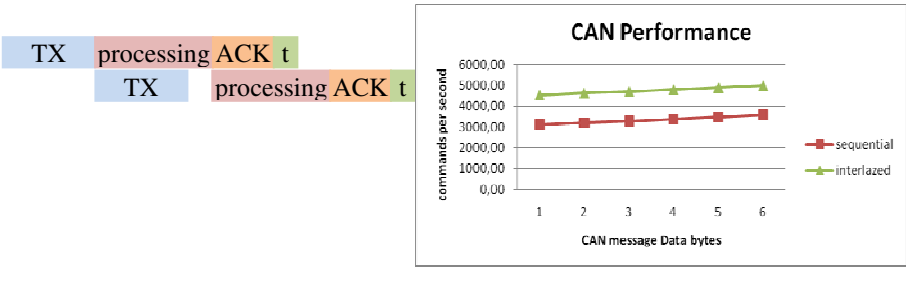
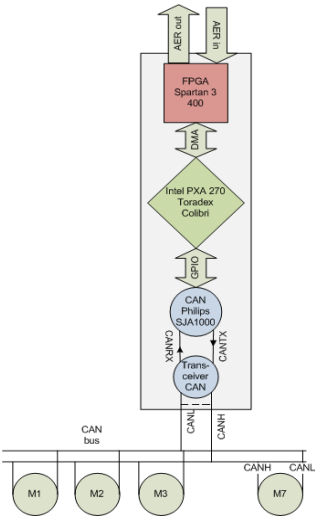
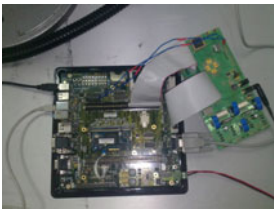


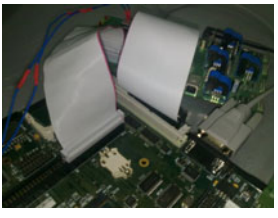
Fig. 4. Performance in Commands per second for sequential and interlaced CAN messages

If we suppose the 1) case with 320 µs of IMI, as it is shown in figure 3, then it means that the FPGA can receive a considerable set of AER events (i.e., 320 events if we consider an Inter-Spike-Interval of 1µs or higher) in order to calculate the new updated position of each node of the arm before sending a new CAN message.





Toradex board with CAN interface and Colobri PXA270 connected to AER-Robot board.



Robotic arm used in VULCANO

Fig. 5. AER-CAN bridge block diagram and prototype boards used under VULCANO project

5 Conclusions

In this paper we present a bridge between AER and CAN buses that allow attaching commercial closed robots to neuro-inspired and spike-based sensory-processing systems developed under AER. The bridge is not able to transmit spikes directly to motors, but it is able to translate a stream of spikes from the AER system into an equivalent CAN message for modifying the degree, intensity and power of a fixed articulation in a robot without bandwidth problems. Furthermore, this AER-CAN bridge allows reading robot sensors and translating the information received into a stream of spikes for AER sensory fusion respect to visual information.

References

[1] Lichtsteiner, P., Posh, C., Delbruck, T.: A 128×128 120dB 15 us Asynchronous Temporal Contrast Vision Sensor. IEEE Journal on Solid-State Circuits 43(2), 566–576 (2008)

[2] Chan, V., Liu, S.C., van Schaik, A.: AER EAR: A Matched Silicon Cochlea Pair with Address-Event-Representation Interface. IEEE Trans. Circuits and Systems-I 54(1), 48–59 (2007)

[3] Costas-Santos, J., Serrano-Gotarredona, T., Serrano-Gotarredona, R., Linares-Barranco, B.: A Spatial Contrast Retina with On-chip Calibration for Neuromorphic Spike-Based AER Vision Systems. IEEE Trans. Circuits and Systems-I 54(7), 1444–1458 (2007)

[4] Serrano-Gotarredona, R., et al.: A Neuromorphic Cortical-Layer Microchip for Spike-Based Event Processing Vision Systems. IEEE T. Circuits Systems-I 53(12), 2548–2566 (2006)

[5] Hafliger, P.: Adaptive WTA with an Analog VLSI Neuromorphic Learning Chip. IEEE Transactions on Neural Networks 18(2), 551–572 (2007)

[6] Indiveri, G., Chicca, E., Douglas, R.: A VLSI Array of Low-Power Spiking Neurons and Bistables Synapses with Spike-Timig Dependant Plasticity. IEEE Transactions on Neural Networks 17(1), 211–221 (2006)
[7] Cohen, A., et al.: Report to the National Science Foundation: Workshop on Neuromorphic Engineering, Telluride, Colorado, USA (June-July 2004), http://www.ini.unizh.ch/telluride
[8] CAN specification, http://www.semiconductors.bosch.de/pdf/can2spec.pdf
[9] Sivilotti, M.: Wiring Considerations in analog VLSI Systems with Application to Field-Programmable Networks. Ph.D. Thesis, California Institute of Technology, Pasadena CA (1991)
[10] The 2011 Cognitive Neuromorphic Engineering Workshop, http://capocaccia.ethz.ch/capo/wiki/2011
[11] Jiménez-Fernández, A., Linares-Barranco, A., Paz-Vicente, R., Jimenez-Moreno, G., Berner, R.: Spike-Based Control Monitoring and Analysis With Address Event Representation. In: Proceeding of the AICCSA-2009, Rabat, Morocco, vol. 7, pp. 900–906 (2009)
[12] Ziermann, T., Wildermann, S., Teich, J.: CAN+: A new backward-compatible Controller Area Network (CAN) protocol with up to 16× higher data rates. In: DATE 2009, pp. 1088–1093 (2009)
[13] Linares-Barranco, A., Jiménez-Fernandez, A., Paz-Vicente, R., Varona, S., Jiménez, G.: An AER-based actuator interface for controlling an anthropomorphic robotic hand. In: Mira, J., Álvarez, J.R. (eds.) IWINAC 2007. LNCS, vol. 4528, pp. 479–489. Springer, Heidelberg (2007)
[14] Serrano-Gotarredona, R., Oster, M., Lichtsteiner, P., Linares-Barranco, A., Paz-Vicente, R., Gómez-Rodríguez, F., et al.: CAVIAR: A 45k-neuron, 5M-synapse AER Hardware Sensory-Processing-Learning-Actuating System for High-Speed Visual Object Recognition and Tracking. IEEE Trans. on Neural Networks 20(9), 1417–1438 (2009)