A low power architecture for AER event-processing microcontroller

A Low Power Architecture for AER Event-Processing Microcontroller

Simone Aiassa*, Paolo Motto Ros†, Guido Masera*, Maurizio Martina*
*Dipartimento di Elettronica e Telecomunicazioni (DET), Politecnico di Torino, Torino, Italy
†Electronic Design Laboratory (EDL), Istituto Italiano di Tecnologia (IIT), Genova, Italy
simone.aiassa@studenti.polito.it, paolo.mottoros@iit.it, {guido.masera | maurizio.martina}@polito.it

***Abstract*—This paper presents a custom MSP430™-compatible microcontroller, specifically tailored for quasi-digital processing Address Event Representation (AER) events. Main target applications are fully reprogrammable sensory systems where events pre-processing has to be carried out by means of easily-tunable elaboration algorithms; a microcontroller-based design could provide the right trade-off between flexibility and performance. Key features are good time resolution, high reactivity, on-demand only processing and power consumption reduction. The proposed architecture has been analyzed and compared with an open source MSP430™-compliant microcontroller (openMSP430) in terms of performance and power consumption. Accurate and wide cases-spectrum simulations (targeting ASIC technology) show an average power consumption reduction ranging from 50 % (same operating frequency) up to 79 % (same maximum event rate); equivalently, with the same power budget, an average improvement of either resolution of 84 % or maximum event rate of 1020 % is obtained.**

I. Introduction

Event-driven stands for an engineering approach for the design of electronic hardware by trying to learn from the behavior of biological nervous systems. The key point is that the activity is triggered only when a significant information occurs. Thus, computation and communication subsystems are employed only on-demand, leading to efficient resource utilization and power reduction without compromising performance. As a consequence, this approach is well suited for robotic applications e.g., [1].

Even if it is possible to map event-driven applications onto clock-driven architectures based on commercial off-the-shelf components [2], [3], inherently fully asynchronous event-driven architectures are needed [4], [5] in order to minimize power consumption. At the same time, other solutions [6]–[12] show that it is possible to develop effective event-driven architectures on synchronous hardware (FPGA or ASIC), with all the advantages of relying on well known and tested development tools (e.g., standard HDL compilers and synthesizers) and platforms (e.g., programmable hardware), leading to potentially technology-independent solutions.

The aim of this paper is to propose an example of power efficient address-event microcontroller architecture, which can be used in several applications where an event-based approach can provide significant advantages, e.g., tactile sensing [13]. The idea is on one hand to promote flexibility and effectiveness (with respect to any specific hardware solution), on the other hand to improve time-domain data processing performance and power consumption (w.r.t. common commercially available MCUs, Micro-Controller-Units).

The application chosen as a reference use case (to define significant performance tests) is the processing of Quasi-Digital Address-Event Representation events (QD-AER) into NeuroMorphic ones (NM-AER), in a similar scenario as in [12], where an hardwired application-specific architecture was designed. The proposed solution aims to improve the flexibility of [12] by resorting to a programmable core.

AER is an event-based communication approach, where the only transmitted data is the source identifier/address [14]; it is an efficient way of interconnecting multi-chip systems or to manage inter-systems communication (e.g., [15]). The fundamental concept of the neuromorphic paradigm (applied to sensors) is to send an event only when a change in the sensed physical quantity occurs. On the other hand, the quasi-digital approach, aims to continuously represent time-based analog information over a digital channel, allowing to greatly reduce the complexity of the core of read-out circuits down to only few digital inverter gates [16].

The processing of QD-AER events into NM-AER as a sample application has been chosen for three reasons: first, it is mostly an (event-based) I/O-bound test case, so to stress the real limits of standard MCUs in event-driven applications; second, to exploit the opportunities and performance of a fully firmware reprogrammable device in pre-processing of events; third, to enable the research and development of hybrid QD/NM AER systems [12], where the advantages of quasi-digital read-out circuits (very low power consumption, small size, simplicity) can be complemented by those of a communication infrastructure based on neuromorphic concepts (bandwidth and latency minimization).

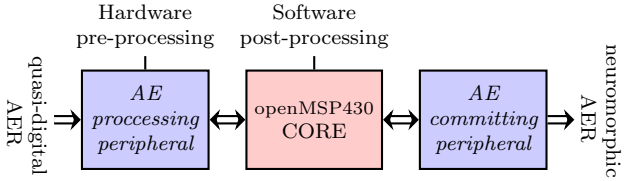


Fig. 1: AER custom microcontroller block diagram: a quasi-digital signal is received and pre-processed by the *AE processing peripheral*. Then, a neuromorphic output is generated by the *AE committing peripheral*.

II. Design and Development

The proposed Address Event microcontroller relies on an open source MSP430™ compatible core, the openMSP430 [17], which is largely adopted into low power applications. It can execute the code generated by Texas Instrument™ tools allowing easy firmware upgrades. The core is connected to two fully compatible firmware programmable custom peripherals. Moreover, all the features of a common microcontroller (like GPIO, Timer, UART, I^2C and so on) are still available. The whole system is described in Verilog HDL to be both FPGA and ASIC ready. Figure 1 shows the corresponding functional block diagram.

The first module is the *AE processing peripheral* devoted to receive and pre-process an input stream of quasi-digital events. Namely, the aim of this peripheral is to manage in an optimized way the time-domain information and to reduce the core workload. Figure 2 shows the details of the *AE processing peripheral* which contains: a timer unit and a processing unit. Both elements include a Data-Path (DP) and a control unit, based on a Finite-State-Machine (FSM). This duplex architecture allows to handle the input with a different frequency than the microcontroller core, in order to tune sampling resolution, without affecting core performance and power consumption. The timer FSM acquires events through a source-ready/destination-ready handshake [12]. The timer unit produces 16-bit timestamps with a free running counter and sends them to the processing unit through a four-phase handshake protocol. The processing unit is able to manage an event with a fixed maximum number of addresses (256 as default). For each address the time distance between contiguous events is computed and stored in an internal memory to keep track of the last four events. Thanks to this strategy, a kind of event-history is recorded in order to get time-correlation between the events. The peripheral can be configured via firmware to generate an interrupt whenever the time difference between two consecutive events overcomes a threshold. It is possible to select which couple of events has to be considered and the threshold can be either absolute or relative. In this last case the bound is a fraction of the time distance between the two preceding events. All the processing is performed inside the DP, which is composed of two adders, one comparator and two shifters. The DP works at the main core clock frequency in order to read/store data from/to registers and memories, which the microcontroller core can directly access without any timing violation. The *AE processing peripheral* is highly firmware configurable providing the following features:

- Selectable input clock source with a configurable prescaler to allow for a wide range of use cases depending on average/maximum event rate and needed time resolution;
- Recent history recording to provide the possibility of implementing time-correlation based algorithms;
- Event-threshold setting on a per address-event basis to tune the sensitivity, also *on the run*, for every different input source;

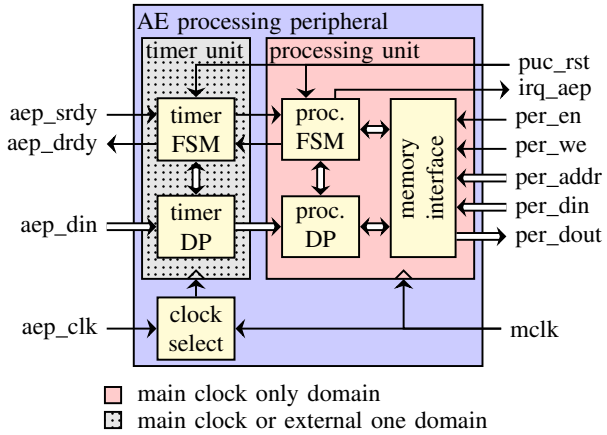


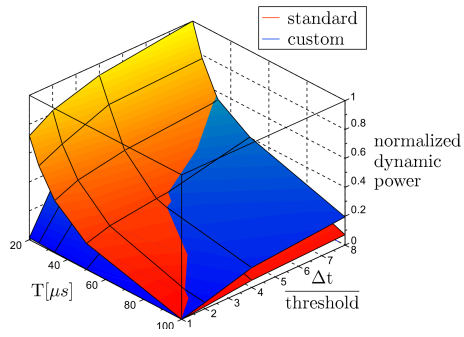
Fig. 2: Address Event processing peripheral functional block diagram with connections to the outside AER signals (on the left side) and to the openMSP430 core (on the right side).

- Absolute or adaptive/relative event-threshold, with polarity values to expand hardware pre-processing capabilities;
- Saturation arithmetic to avoid overflow errors.

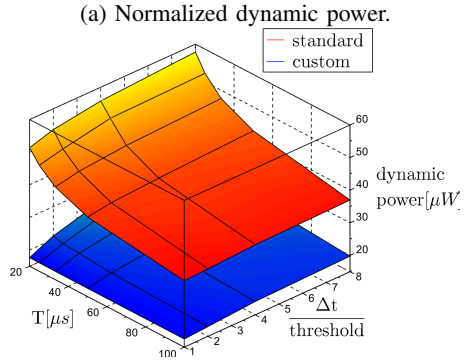
The second peripheral integrated in the proposed MCU is the *AE committing peripheral*, which has been designed to receive events from the openMSP430 core and to commit them out as AER-events as soon as possible. This peripheral can work with two different clock frequencies thanks to a FIFO buffer.

The proposed architecture has been implemented on an FPGA board based on a Xilinx ® Spartan3 XC3S1400A. The whole developed system occupies 41% of available slices and 17 BRAMs (required by the main memory). Moreover, both the proposed solution and the original openMSP430 model have been synthesized on ASIC technology in order to actually analyze the advantages offered by the proposed solution in term of power consumption. In the following we will refer to the original openMSP430 MCU as *standard* microcontroller and to the proposed solution as *custom*.

In the *standard* microcontroller input/output and time information are managed by a GPIO and a timer, respectively. All the computation is performed by the core through firmware/software code. The *custom* solution differs from the original one just for the presence of the custom peripherals described in the previous paragraphs, which are exploited to leverage the CPU from computation. The synthesis is performed through Synopsys Design Compiler®, on a 32 nm technology. The *standard* solution occupies an area of 8973 GE (18257 μm^2), with a total static power consumption of 24.27 mW and achieving a maximum main clock frequency of 230 MHz. Meanwhile, the *custom* MCU reaches an area of 9537 GE (19391 μm^2), a static power consumption of 25.48 mW and a maximum speed of 170 MHz. These results show that the proposed solution does not cause relevant overhead in terms of occupation, static power consumption and peak core performance.



(a) Normalized dynamic power.



(b) Absolute dynamic power.

Fig. 3: Total dynamic power (on z axis) varying period (T on x axis) and signal/threshold ratio ($\Delta t/threshold$ on y axis).

III. Methods

In this section the method used for validation and result generation is discussed. In the simulation the real MCU is emulated by the original openMSP430. Clearly, the two implementations are not equivalent in terms of technology optimization; however, this analysis aims to show the advantages offered by the proposed approach. For this reason only dynamic power consumption and synchronous timing performance are analyzed. In order to build a realistic environment a deterministic quasi-digital signal has been applied to both the original openMSP430 and the proposed microcontroller. Its Pulse Position Modulation (PPM) model is as follows:

$$t_2 - t_1 = T + k \cdot \frac{\Delta t}{2} \qquad k \in [-1, 1], \tag{1}$$

where:

- T ranges from 20 µs up to 100 µs that means a carrier frequency in the range 10÷50 kHz, with five equally spaced frequency steps.
- Δt ranges from 1 µs up to 40 µs, in four threshold/signal ratio constant steps; this corresponds to a spike-based signal triggered in a range 0÷80 % of input events.
- k is a pseudo-random number, in an uniform distribution.

Thanks to the firmware programmable structure, users can develop different algorithms. In this work six algorithms are considered as benchmarks. They differ in terms of pre-processing thresholding method (absolute or relative) and post-processing complexity. Indeed, the four history values can be ignored (no filtering) or exploited to implement time-correlation algorithms, namely, average or median filtering.

IV. Results

Fig. 3a shows the power consumption normalized w.r.t. the corresponding maximum value of the proposed architecture in different configurations. In this analysis, we define $\Delta t/threshold$ as the best parameter to represent the amount of significant information in an event-based signal. Indeed, the signal/threshold ratio reflects the number of quasi-digital events which trigger the neuromorphic output. The red upper plane describes the *standard* architecture, while the blue lower one the *custom* architecture. The event frequency increases moving on the left side axis, the signal/threshold ratio is presented on right side axis, instead. As it can be observed, the power consumption in the *custom* microcontroller remains approximately stable as long as there is not a significant variation on the input signal (at $\Delta t/threshold$ equal to 1), as opposite in the *standard* architecture the power consumption increases exponentially. This behavior can be explained considering that the original microcontroller always performs the same number of operations every time a new event arises, so the energy consumption depends primarily on the number of events per unit of time (T). Differently, the *AE processing peripheral* in the proposed architecture wakes the core only when a significant input occurs, so keeping low the power consumption when no relevant information is present.

It is worth noting that the total dynamic power consumption in the proposed solution is lower than the one in the original openMSP430 as highlighted in Fig. 3b. The factor that leads to energy reduction is the presence of a dedicated hardware I/O management unit with computational capability, embedded into the custom peripherals. Since the better overall performance of the *custom* microcontroller with respect to the *standard* one is linked to its double clock structure, a deep analysis on this aspect has been performed. The two input clock sources can be varied according to the imposed power budget. Indeed, the processor main clock affects the dynamic power consumption with a factor of 0.64 µW/MHz and changes linearly the maximum sustained event rate, i.e., the maximum number of events per time unit. The external AE clock tunes the resolution and contributes with a factor

TABLE I: Performance of different solutions, in term of main clock frequency (F_{mclk}), maximum event rate (ER_{max}), minimum time resolution (Δt_{min}), average dynamic power consumption ($\overline{P}_{dyn}$), and average energy per event ($\overline{EpE}$).

	solution	F_{mclk} [MHz]	ER_{max} [kevent/s]	Δt_{min} [ns]	$\overline{P}_{dyn}$ [µW]	$\overline{EpE}$ [nJ/event]
	openMSP430	20	116	50	45.5	1.90
proposed	**nominal**	20	571	50	20.9	0.95
	min. pow.	4.1	117	50	8.3	0.40
	max. res.	10	284	8	38.0	1.70
	balanced	28	793	14	37.5	1.67
	max. perf.	45	1300	50	37.2	1.66

TABLE II: Comparison with programmable synchronous event-acquiring/processing solutions.

	[6]	[7]	[8]	[9]	[10]*	[11]	[17]†	proposed
Core technology	FPGA	FPGA	CPLD	CMOS-180nm	CMOS-130nm	FPGA	FPGA/CMOS-32nm	
Event processing	yes	no	no	yes	yes	yes	yes	yes
Gate count [kGates]	n.a.	n.a.	n.a.	19.2	n.a.	n.a.	8.9	9.5
System clock [MHz]	100	20	30	100	200	75	20	4÷45‡
Max time resolution [ns]	50000	1000	33	10	n.a.	n.a.	50	8
Max sustained event rate [Mevent/s]	n.a.	1	5	5.1	0.16	0.58	0.12	1.3
Power consumption [mW]	n.a.	n.a.	n.a.	300	1000	1500	24.3§	25.5§

* Considering a single node (core)
† Simulated on the given application
‡ Test setup
§ Total power from simulation test setup (static plus dynamic contributions)

of 0.22 µW/MHz. Therefore, improving the resolution is less energy expensive than increasing the maximum event rate.

Table I highlights good results achieved by the proposed solution. The first row contains the *standard* microcontroller working at 20MHz; the others present different *custom* solutions, which can be implemented by varying the system clock frequency. At nominal frequency the *custom* MCU shows a peak event processing-rate five times higher than the *standard* MCU. Furthermore, it features less than 20 % of the power consumed by the *standard* MCU to achieve the same performance. If both solutions are constrained by an equal energy budget either the input event-rate can be raised of one order of magnitude or the resolution improved six times. With the system reactivity approximated to the reciprocal of the maximum event rate, the *custom* MCU shows also a response-time improvement (up to 1020 % w.r.t. the *standard* one).

Table II shows a comparison between the proposed solution and [6]–[11], [17]. It has been restricted to event-driven acquiring/processing systems based on synchronous hardware. As expected, every FPGA or CPLD solution can not be fairly compared in terms of power. Considering the ASIC solutions, the ratio between the power consumption and the maximum event rate can be defined as a metric under which evaluating the advantages of the proposed solution. The presented microcontroller can handle a given maximum input event rate consuming a total power three times lower than what required by other solutions to reach similar performance.

V. Conclusion

The presented event-processing MCU, thanks to the inclusion of custom peripherals, promotes energy saving (the power consumption is more than halved), ensuring performance (the peak processing rate can be increased tenfold), without losing any advantage of a fully firmware-programmable commercial architecture. Full source is available [18].

References

[1] F. Rea, G. Metta, and C. Bartolozzi, "Event-driven visual attention for the humanoid robot iCub," *Frontiers in Neuroscience*, vol. 7, no. 234, 2013.

[2] J. Conradt, M. Cook, R. Berner, P. Lichtsteiner, R. Douglas, and T. Delbruck, "A pencil balancing robot using a pair of AER dynamic vision sensors," in *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on*, pp. 781–784, May 2009.

[3] T. Delbruck and M. Lang, "Robotic goalie with 3 ms reaction time at 4% CPU load using event-based dynamic vision sensor," *Frontiers in Neuroscience*, vol. 7, Nov 2013.

[4] N. Qiao, H. Mostafa, F. Corradi, M. Osswald, F. Stefanini, D. Sumislawska, and G. Indiveri, "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128k synapses," *Frontiers in Neuroscience*, vol. 9, p. 141, 2015.

[5] C. T. O. Otero, J. Tse, R. Karmazin, B. Hill, and R. Manohar, "Ulsnap: An ultra-low power event-driven microcontroller for sensor network nodes," in *Fifteenth International Symposium on Quality Electronic Design*, pp. 667–674, March 2014.

[6] R. George, C. Mayr, G. Indiveri, and S. Vassanelli, "Event-based softcore processor in a biohybrid setup applied to structural plasticity," in *2015 International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP)*, pp. 1–4, June 2015.

[7] E. Chicca, A. M. Whatley, P. Lichtsteiner, V. Dante, T. Delbruck, P. D. Giudice, R. J. Douglas, and G. Indiveri, "A multichip pulse-based neuromorphic infrastructure and its application to a model of orientation selectivity," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, pp. 981–993, May 2007.

[8] R. Berner, T. Delbruck, A. Civit-Balcells, and A. Linares-Barranco, "A 5 Meps $100 USB2.0 address-event monitor-sequencer interface," in *2007 IEEE International Symposium on Circuits and Systems*, pp. 2451–2454, May 2007.

[9] M. Hofstatter, P. Schn, and C. Posch, "A SPARC-compatible general purpose address-event processor with 20-bit 10ns-resolution asynchronous sensor data interface in 0.18µm cmos," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pp. 4229–4232, May 2010.

[10] S. Furber, D. Lester, L. Plana, J. Garside, E. Painkras, S. Temple, and A. Brown, "Overview of the SpiNNaker system architecture," *Computers, IEEE Transactions on*, vol. 62, pp. 2454–2467, Dec 2013.

[11] D. Neil and S.-C. Liu, "Minitaur, an event-driven FPGA-based spiking network accelerator," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 22, pp. 2621–2628, Dec 2014.

[12] P. Motto Ros, M. Crepaldi, and D. Demarchi, "A hybrid quasi-digital/neuromorphic architecture for tactile sensing in humanoid robots," in *2015 6th International Workshop on Advances in Sensors and Interfaces (IWASI)*, pp. 126–130, June 2015.

[13] C. Bartolozzi, P. Motto Ros, F. Diotalevi, N. Jamali, L. Natale, M. Crepaldi, and D. Demarchi, "Event-driven encoding of off-the-shelf tactile sensors for compression and latency optimisation for robotic skin," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep 2017.

[14] K. Boahen, "Point-to-point connectivity between neuromorphic chips using address events," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, vol. 47, pp. 416–434, May 2000.

[15] P. Motto Ros, M. Crepaldi, C. Bartolozzi, and D. Demarchi, "Asynchronous DC-free serial protocol for event-based AER systems," in *2015 IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, pp. 248–251, Dec 2015.

[16] A. Damilano, P. Motto Ros, A. Sanginario, A. Chiolerio, S. Bocchini, I. Roppolo, C. F. Pirri, S. Carrara, D. Demarchi, and M. Crepaldi, "A robust capacitive digital read-out circuit for a scalable tactile skin," *IEEE Sensors Journal*, vol. 17, pp. 2682–2695, May 2017.

[17] O. Girard, *openmsp430*, 1.16 ed., December 2007.

[18] S. Aiassa, P. Motto Ros, G. Masera, and M. Martina, "A Low Power Architecture for AER Event-Processing Microcontroller: full source (both HDL and code). [Online]. Available: http://personal.det.polito.it/maurizio.martina/event.html," June 2017.