Low-Latency Image Acquisition and Processing with a Programmable Vision-System-on-Chip

Jens Döge, Christoph Hoppe, Andreas Reichel, Peter Reichel, Nico Peter and Holger Priwitzer
Fraunhofer Institute for Integrated Circuits IIS, Germany
E-mail: jens.doege@eas.iis.fraunhofer.de

***Abstract*—This work aims to demonstrate the benefits of using a Vision-System-on-Chip for image processing tasks with very high latency demands between image acquisition and processing. By leveraging a column-parallel, mixed-signal data path, which is entirely software-defined by three application-specific instruction-set processors (ASIPs), image data within multiple regions of interest can be analyzed at a frame rate of 5 kHz. Thus, with a delay of 0.44 ms, the trajectory of a moving object is analyzed and the object is precisely deflected using a solenoid.**

***Index Terms*—image sensor, Vision-System-on-Chip, VSoC, ASIP, low latency, actuator control, trajectory prediction**

I. Introduction

Döge et al. [1] have presented a novel Vision-System-on-Chip (VSoC) with column-parallel, mixed-signal data processing and a flexible digital interface. With three dedicated, stack-based application-specific instruction-set processors (ASIPs), this system allows for executing entirely software-defined, complex image processing algorithms at very high image rates. Due to the transfer function of the VSoC's charge-based pixel cells exhibiting linear-logarithmic characteristics, its dynamic range is above 120 dB.

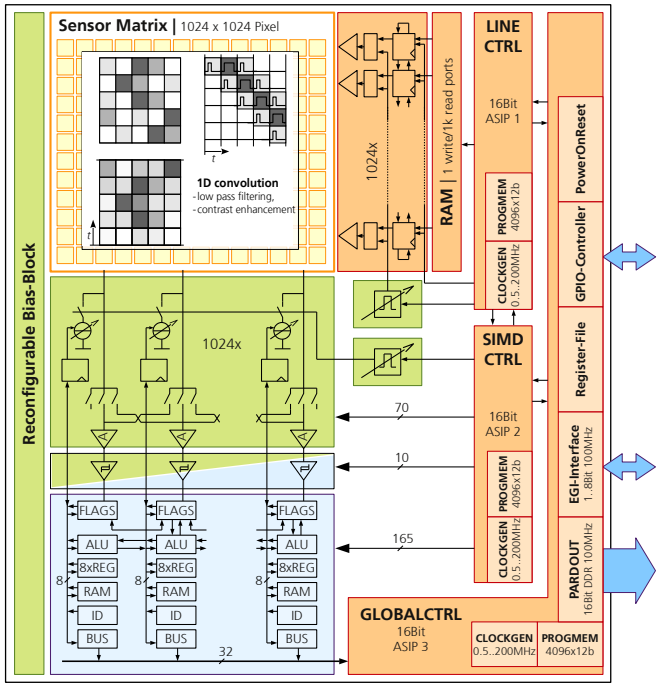


Figure 1. Architecture of the Vision-System-on-Chip[1].

The architecture of the VSoC is depicted in Figure 1. Communication with the outside world is handled by a dedicated ASIP (`Global Control`). The sensor matrix consists of an array of 1024 × 1024 charge-based pixel cells with 8.75 µm pitch. Pixel readout can be controlled on a row-by-row basis using a dedicated ASIP (`Line Ctrl`), e. g. to readout multiple regions of interest with varying exposure times or to perform column-wise filtering (high or low pass) across multiple rows. The ASIPs' control flow, as well as inter-ASIP synchronization, is defined using a subset of the high-level programming language Python, as well as bindings to embed processor-specific assembler instructions [2].

The resulting charges in each column are converted to digital values, ranging from 1 to 10 bit, using a SIMD array of 1024 processor elements (PE). This conversion process, which is also entirely software-defined, is managed by another dedicated ASIP (`SIMD Control`). With 55 instructions and operating at a clock frequency of up to 100 MHz, the PEs allow for implementing a wide range of algorithms, such as sheet of light (SoL), Features from Accelerated Segment Test (FAST, proposed by Rosten and Drummond [3], [4]) or Local Binary Patterns (LBP)[5]. The software development process is simplified with the aid of a simulation environment and a library providing basic functionality and hooks to insert one's own instructions at well-defined points [6].

II. Low-Latency Image Acquisition and Processing

In contrast to conventional image processing systems consisting of an image sensor and a dedicated processing unit (PC and/or FPGA), this approach allows for implementing complex vision-based control tasks using solely the VSoC and its general-purpose interfaces, without having to rely on external hardware. Such an image processing task is schematically depicted in Figure 2. It can be roughly divided into five steps: feature acquisition (1), feature-based control (2), precise data analysis (3), actuation (4) and image acquisition (5).

In the first step, after exposing the pixel cells' photo diodes, image data are readout from the sensor and initial column-parallel information processing and feature extraction (1) takes place with very high speed, e. g. by reducing the spatial and/or value resolution (region of interest or binarized image data). The results of this step are immediately used to decide whether or not feature image acquisition has to be repeated (2). If not, either the digitized data are used directly for the following data analysis (3), or an additional, more precise digitization step

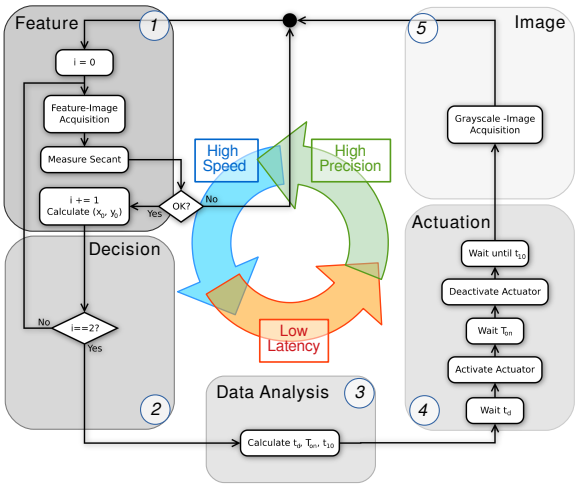


Figure 2. VSoC-Integrated Sensor and Actuator Control.

(other region of interest, higher spatial and value resolution) is performed with the existing pixel values. This analysis step is often relatively complex, i. e. quickly deciding if it should take place, based on the available data, has a significant impact on the latency of the overall system. Controlling the (e. g. mechanical or optical) actuators based on the available data can then be performed with minimal latency, as well (4). In the next and final step, additional readout and/or exposure steps can be performed in high resolution and with high accuracy or in collaboration with the actuators (e. g. lock-in measurement with mechanical stimuli or switching wavelength range) (5).

It is not always necessary to include all of the above steps. However, with the following demonstration, an exemplary implementation of this general procedure shall be illustrated.

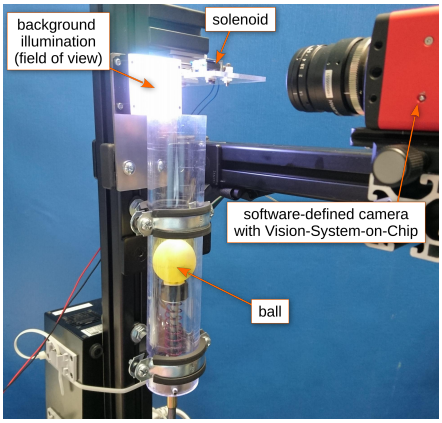
III. Measurement Task

A. General Specification

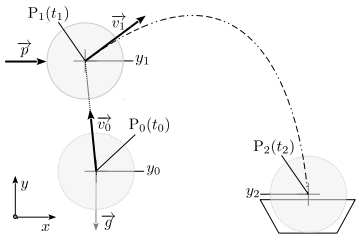
Figure 3 illustrates the task to be solved by this demonstration. A solenoid shall very precisely deflect the trajectory of a table tennis ball, such that it lands within a vessel. The ball is put in motion within a tube by means of a spring with unknown preload determined by a human. At time t_0, the ball has reached height y_0 at point P_0. The velocity vector at this point, which is denoted by $\overrightarrow{v_0}$, is used to estimate the time t_1 that it reaches the point P_1 at height y_1. The solenoid is triggered to transmit an impulse $\overrightarrow{p}$ onto the ball, such that its trajectory ends at height y_2 within point P_2. Using time t_1, point P_1 and impulse $\overrightarrow{p}$, as well as the actuator's dynamics, the VSoC determines the point in time and current flow pulse width to control the actuator. Additionally, a gray-value image is acquired at the estimated time t_1, capturing the exact moment when the ball is hit by the plunger.

B. Determining Position and Velocity

Deducing the ball's velocity from the point in time that it passes two well-defined points in the image would be the most straightforward measurement method for the task



(a) Photography



(b) Schematic Diagram

Figure 3. Experimental Setup.

at hand. However, taking the actuator's inherent latency of approximately 10 ms and a possible maximum velocity of $4\,m/s$ into account, even a sampling rate of 20 kHz would inevitably result in either the position measurement resolution being too coarse or the remaining prediction time being too short. This is why an alternative method was chosen, as depicted in Figure 4.

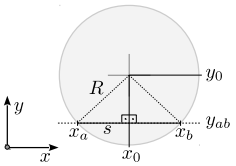


Figure 4. Measurement of Position P_i.

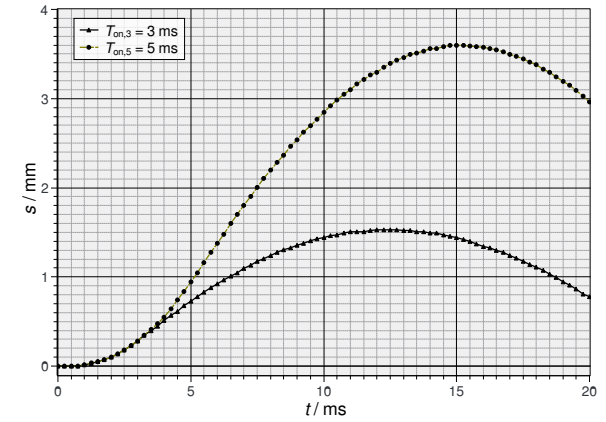
The specification of the demonstration implies that, in terms of the image coordinate system, the ball moves predominantly along the y axis. Therefore, when capturing and calculating a smoothed first derivative of a reference image row y_{ab} at a high sample rate ($f > 5$ kHz) in the SIMD unit, the secant $s = \overline{x_a x_b}$ will be found by a maximum search. Assuming that the ball's radius R is known, the image coordinates of the center (x_0, y_0) can be obtained from Equations 1 and 2.

$$x_0 = \frac{x_a + x_b}{2} \tag{1}$$

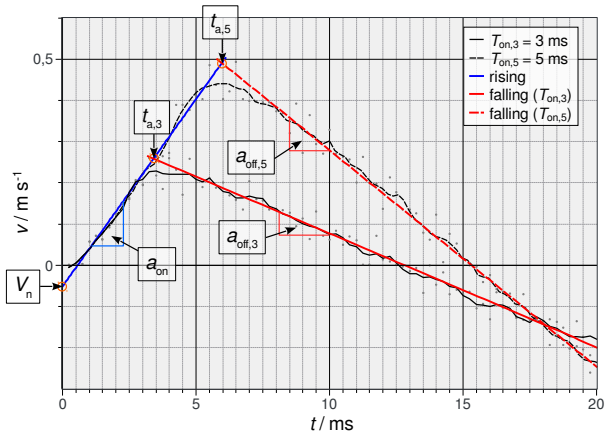
$$y_0 = y_{ab} + \sqrt{R^2 - \frac{(x_b - x_a)^2}{4}} \tag{2}$$

Parallel to analyzing the secant s, the reference row y_{ab} is scanned again after a defined time T_1 in order to check whether the coordinates (x_0, y_0) of the center have changed sufficiently, i. e. whether or not it is possible to measure the ball's velocity. If no significant change was detected before a timeout T_0 was reached, the ball is apparently moving too slow to reach the vessel and the process is aborted.

C. Actuator Dynamics



(a) Plunger Position s vs. Time t



(b) Speed v vs. Time t

Figure 5. Results of the Actuator Dynamics Measurements.

The solenoid consists of a coil with movable iron core and a return spring. Upon energizing the coil, the iron core moves into it and, once the coil is no longer supplied with a voltage, is pulled back to its initial position by the return spring. This kind of actuator is typically used for static switching operations, such as opening or closing a valve. This is why there is no information about dynamic behavior in their data sheets. Instead, a preliminary experiment was conducted to study the actuator dynamics. For this purpose, the actuator was triggered with varying pulse lengths T_{on} and the plunger's movement was observed by a high-speed camera at 4000 frames per second. Figure 5a depicts the measured displacement/time characteristics for varying T_{on} (3 ms and 5 ms). From this, velocity/time characteristics can be deduced, which are depicted in Figure 5b. At first, velocity increases approximately linearly, i. e. with constant acceleration, and, after a delay depending on the initial velocity, decreases linearly once the voltage supply is turned off. Furthermore, the delay does not start immediately after turning off the voltage. Thus, the actuator can be modeled according to the following equations.

$$v_{\mathrm{m}}(t) = \begin{cases} a_{\mathrm{on}} \cdot t + v_{\mathrm{n}} & t \le t_a \\ v_{\mathrm{m}}(t_a) + C_{\mathrm{off}} \cdot v_{\mathrm{m}}(t_a) \cdot (t - t_a) & t > t_a \end{cases} \tag{3}$$

$$s_{\mathrm{m}}(t) = \begin{cases} \frac{a_{\mathrm{on}}}{2} \cdot t^2 + v_{\mathrm{n}} \cdot t & t \le t_a \\ \frac{v_{\mathrm{m}}(t) + v_{\mathrm{m}}(t_a)}{2} \cdot (t - t_a) + s_{\mathrm{m}}(t_a) & t > t_a \end{cases} \tag{4}$$

$$t_a = c_1 T_{\mathrm{on}}^2 + c_2 T_{\mathrm{on}} + c_3 \tag{5}$$

The model parameters a_{on}, v_{n}, C_{off}, c_1, c_2 and c_3 can be determined from the measurement data via regression calculation. Slope and intercept of the regression line $v(t)$, $t \in [0, T_{\mathrm{on}}]$ determine a_{on} and v_{n}, respectively. The parameter C_{off} is derived from one of the regression lines $v(t)$, $t > t_a$ by dividing its slope by the initial velocity $v(t_a)$. Finally, c_{i}, $\mathrm{i} \in \{1, 2, 3\}$ are determined from fitting a parabola to $t_a(T_{\mathrm{on}})$. After changes to the setup the model parameters can be automatically updated in a calibration cycle based on image data from the VSoC at 5 kHz.

D. Trajectory Prediction and Actuator Parameter Calculation

Assuming frictionless motion, the ball follows a parabolic trajectory, given by Equation 6.

$$\overrightarrow{s(t)} = \begin{bmatrix} v_{1x} \cdot (t - t_1) \\ v_{1y} \cdot (t - t_1) - \frac{g}{2}(t - t_1)^2 + y_1 \end{bmatrix} \tag{6}$$

If v_{1y}, P_0, P_1 and P_2 are known, v_{1x} can be determined such that P_2 is reached. To calculate $\overrightarrow{v_1}$, the measured $\overrightarrow{v_0}$ is used to determine $(t_1 - t_0)$, v_{1y} and P_{1x}.

$$\triangle t_{10} = t_1 - t_0 = \frac{v_{0y} - \sqrt{v_{0y}^2 - 2g(y_1 - y_0)}}{g} \tag{7}$$

$$v_{1y} = v_{0y} - g \cdot \triangle t_{10} \qquad P_{1x} = P_{0x} + v_{0x} \cdot \triangle t_{10} \tag{8}$$

Assuming that the actuator hits the ball exactly perpendicularly, the actuator does not have any influence on the trajectory's y component. By rearranging Equation 6 and inserting all known parameters, v_{1x} can be determined.

$$v_{1x} = \frac{g \cdot (P_{2x} - P_{1x})}{v_{1y} - \sqrt{v_{1y}^2 + 2g(y_1 - y_2)}} \tag{9}$$

If P_{1x} and v_{1x} as well as the actuator's preload are known, the required solutions to the actuator model v_{m} and s_{m} can be derived based on the elastic momentum transfer from the actuator core (m_{m}) to the ball (m_{b}, $v_{0x} \approx 0$).

$$v_{\mathrm{m}} = 2\frac{v_{1x}(m_{\mathrm{m}} + m_{\mathrm{b}})}{m_{\mathrm{m}}}. \tag{10}$$

From this, T_{on} and the delay time t_{d} are calculated, such that at time t_1 the desired momentum is transferred from the actuator onto the ball. This is done by solving the the non-linear equation system consisting of Equations 3 and 4 for t_a and t. Then, T_{on} is determined from Equation 5. Finally, the delay time t_{d} is given by Equation 11, with $t_{\mathrm{hit}} = t$ from the solution to the actuator model.

$$t_d = \triangle t_{10} - t_{\mathrm{hit}} \tag{11}$$

IV. Software Implementation

All calculations and processes are to be realized directly by the VSoC. Because its ASIPs lack support for floating point arithmetic, all operations have to be performed with fixed point arithmetic, instead. The required accuracy of the calculation thereby determines the bit width of the utilized variables. The processors offer hardware support for addition, subtraction and multiplication of 16-bit numbers. Division and square root are not natively supported and have to be determined algorithmically. Table I depicts the time required for executing certain operations with varying accuracy.

Operation	Input Bits	Output Bits	min. #Clocks	max. #Clocks
`add/sub`	16	16	1	7
`add/sub`	32	32	3	15
`mul`	16	32	10	19
`shift`	16	16	1	7
`shift`	32	32	3	15
`div`	16	16	466	472
`div`	32	32	2300	2500
`sqrt`	16	8	244	250
`sqrt`	32	16	1250	1450

Table I
Maximum Time Required for Arithmetic Operations.

A. Determining Position and Velocity

The points x_a and x_b are determined with pixel accuracy by finding the first and last covered pixel within row y_{ab}. Equations 1 and 2 are solved for these values, resulting in the x_0 coordinate of the ball's center point being determined at a maximum accuracy of half a pixel. Using 16-bit square root, y_0 can only be determined with two pixel accuracy. However, by implementing this operation for 32-bit input values, $1/16$ pixel accuracy for the y_0 coordinate was achieved. At a sample rate of $5\,\mathrm{kHz}$ and magnification of $50\,{}^{\mu m}/_{\mathrm{pixel}}$, $\overrightarrow{v_0}$ can be determined from two sequentially calculated positions $(x_0,\, y_0)$ with the following uncertainties.

$$\triangle\, v_{0x} = 0.25\,{}^{m}/_{s} \;\; \triangle\, v_{0y} = 0.03\,{}^{m}/_{s} \tag{12}$$

Calculating position and velocity takes approximately between 1600 and 1800 clock cycles or 16 µs to 18 µs, comfortably allowing for a refresh rate of $10\,\mathrm{kHz}$ and above.

B. Trajectory Prediction and Actuator Parameter Calculation

Equation 7 can now be solved for the measured velocity $\overrightarrow{v_0}$, which takes approximately 2000 clock cycles. Calculating the parameters v_{1y} and P_{1x} according to Equation 8 takes about 30 clock cycles. Then, v_{1x} can be calculated within about 2000 clock cycles according to Equation 9.

The equation system of Equations 3 and 4 has been solved using Newton's method. The remaining parameters t_d and T_{on} are determined by solving Equation 11 and by transforming Equation 5 to Equation 13, respectively. Based on this results during an calibration step two look-up tables are generated to reduce the processing effort from 11000 (live calculations on ASIP) to approximately 20 cycles.

$$T_{on} = \frac{1}{2c_1} \cdot \left(c_2 + \sqrt{c_2^2 - 4c_1\,(c_3 - t_a)} \right) \tag{13}$$

In total, approximately 4000 clock cycles or 40 µs are required to calculate all the parameters needed to control the actuator. Therefore, the maximum ball velocity is not limited by the calculations on the VSoC, but by the actuator's response time of multiple milliseconds. To demonstrate the fusion of data processing and image acquisition based on calculated trigger information on the same VSoC, an image of the actuator hitting the ball is captured. Such a low-latency feedback loop greatly benefits from the VSoC: Not only would the required operations in the feature extraction and measurement phase exceed the processing capabilities of similarly low-power microcontrollers, but the highly adaptive sensor readout scheme also rules out the use of conventional imaging systems.

V. Conclusion

A methodology for general tasks of low-latency image acquisition and processing that particularly benefits from the use of a novel, programmable VSoC, was presented and exemplarily realized as a demonstration to illustrate its practical use. Features are thereby acquired at a frame rate of 5 kHz and with latency much smaller than the corresponding frame period. Calculating the actuator parameters, which involves parameter extraction and solving a non-linear equation system, takes place with latency below two frame periods, i. e. only $1/50$ the latency of the mechanical system. Thus, the following image acquisition can be performed at exactly the predicted point in time.

VI. Acknowledgement

The development was supported by the German Federal Ministry of Education and Research (BMBF) within the information and communication technology (IKT) research initiative for SME (KMU Innovativ), joint project "Smart-FusionCam", grant number 01IS15030B. The authors of this paper are solely responsible for its content.

REFERENCES

[1] J. Döge, C. Hoppe, P. Reichel, and N. Peter, "Megapixel HDR Image Sensor SoC with Highly Parallel Mixed-Signal Processing," in *International Image Sensor Workshop (IISW)*, 2015.

[2] P. Reichel, J. Döge, N. Peter, C. Hoppe, and A. Reichel, "An asip-based control system for vision chips with highly parallel signal processing," in *2015 IEEE 24th International Symposium on Industrial Electronics (ISIE)*. IEEE, 2015, pp. 932–937.

[3] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking." in *IEEE International Conference on Computer Vision*, vol. 2, October 2005, pp. 1508–1511. [Online]. Available: http://www.coxphysics.com/work/rosten_2005_tracking.pdf

[4] ——, "Machine learning for high-speed corner detection," in *European Conference on Computer Vision*, vol. 1, May 2006, pp. 430–443. [Online]. Available: http://www.coxphysics.com/work/rosten_2006_machine.pdf

[5] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, vol. 1. IEEE, 1994, pp. 582–585.

[6] P. Reichel, J. Döge, C. Hoppe, N. Peter, A. Reichel, and P. Schneider, "Simulation platform for application development on a vision-system-on-chip with integrated signal processing," *Journal of Electronic Imaging*, vol. 25, no. 4, pp. 041 004–041 004, 2016.