26th IMEKO TC4 International Symposium
24th International Workshop on ADC and DAC Modelling and Testing
IMEKO TC-4 2023
Pordenone, Italy / September 20-21, 2023

Custom Synthesizable VHDL Processor for Embedded Capacitive Angle Sensor Data Processing

Milos Drutarovsky[1], Ondrej Benedik[2], Miroslav Sokol[1], Pavol Galajda[1], Jan Saliga[1], Jan Ligus[2], Cristian D Stratyinski[2]

[1]*Technical University of Kosice, Letna 1/9, 042 00 Kosice, Slovakia, Milos.Drutarovsky@tuke.sk*
[2]*CTRL Ltd., Omska 14, 040 01 Kosice, Slovakia, Cristian.Stratyinski@ctrl1.eu*

***Abstract* – We describe custom architecture of a small synthesizable soft processor for the next generation of proprietary capacitive angle sensor (CAPSE) developed by CTRL company for space applications. We process data streams from ADCs by custom developed 16-bit processor. The processor is written in platform independent VHDL code. It uses a small Leros (soft processor) control unit and several custom coprocessors including CORDIC, fixed-point fractional multiplier, and adder with barrel shifter optimized for fractional fixed-point arithmetic. We describe architecture of the proposed processor and present the results of developed custom processor mapping to the target FPGA circuit. The complete processor occupies only ∼2100 Logic Elements in target FPGA and complete firmware has less than 220 instructions. For a 10 kHz sampling rate, it requires less than 3 MHz system clock frequency.**

Fig. 1. CAPSE sensor prototype with embedded RadHard microcontroller.

I. INTRODUCTION

The robust and reliable measurement of the absolute angle of rotation has an important role in positioning systems in space mechanisms in order to check and control the position of the mechanism moving parts. Position sensors have a significant impact on the performance, functionality, and reliability of space mechanisms and space missions. Capacitive encoders are quite attractive because they can be easily manufactured as small units with simple construction and low power consumption [1]. The proprietary capacitive sensor CAPSE shown in Fig. 1 was developed by CTRL company [2], within ESA supported activity.

CAPSE is a contactless angular sensor, that measures the rotation angle between its rotor mechanical part and its stator mechanical part. The uniqueness of the design is a measurement of absolute angle values, low susceptibility to vibration, temperature error canceling, working in vacuum/air or lubricant conditions, small particles do not affect the performance, easy and fast design scalability, manufacturing and its integration.

The ultimate future goal of CAPSE development is to replace the currently used RadHard microcontroller (MCU) used in CAPSE Engineering Qualification Model (EQM) by a custom digital Application Specific Integrated Circuit (ASIC). In this paper we describe the first step in this effort, the developed custom synthesizable VHDL processor with specialized peripherals implemented for efficient processing of signals acquired by Analog to Digital Converters (ADCs) from CAPSE sensor. We mapped our custom processor into FPGA as a proof of concept in the first stage of future ASIC development in the actual CAPASIC project.

II. EXPERIMENTAL CAPASIC HARDWARE

The CAPASIC project moves main processing functions (data preprocessing of 3 separate digitized analog input channels, math for angle computation, angle linearization, interfaces and status of computation monitoring) into the custom digital Hardware (HW). We describe proposed custom HW in synthesizable VHDL and map it currently into the off-the-shelf FPGA circuit. This allows us to map, test and optimize the main CAPASIC functionality (referred to

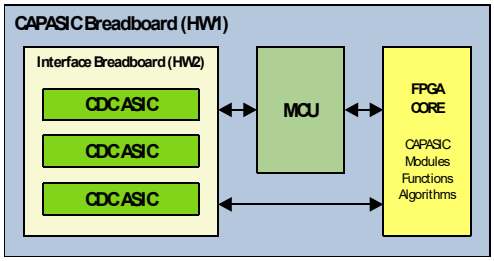

Fig. 2. Experimental CAPASIC hardware.

as FPGA CORE in further text and shown in Fig. 2) already in current project development stage with real capacitive analog sensor prototype.

A. *Analog interface board*

The analog Interface Breadboad (HW2) uses 3 separate ADCs for data acquisition. The ESS214D [3] rad-hard tolerant capacitive sensor signal conditioner integrated circuit (CDC) realized as ASIC includes also sigma-delta ADC. The inputs of CDC ASICs connect directly to the analog capacities of absolute CAPSE rotary encoder. The HW2 provides digital data representing actual values of 3 monitored capacities in real-time. Developed and implemented FPGA CORE algorithms are not limited only to the used CDC ASIC, and, in principle, we can use also other sigma-delta ADCs in future designs.

B. *MCU based development interface*

The MCU provides flexible development interface for transformation of CDCs data to the format required by FPGA CORE development and testing of the interface. For development we use off-the-shelf development board with ARM based MCU. It provides us the following main functionality:

- I2C (optionally SPI) for interface to CDC ASICs.
- Parallel I/O (PIO) for communication with FPGA CORE.
- UART interface for debugging and development.

III. ARCHITECTURE OF PROPOSED FPGA CORE

The main function of the FPGA CORE (shown in Fig. 3) is real-time processing of 3 acquired parallel data streams provided by CDCs. The FPGA CORE performs main signal processing functionality required for computation of actual angle α from acquired CDC data streams in real-time. The main time-critical tasks executed by FPGA CORE are:

- Sequentially read all input data streams from PIO interface with required resolution.

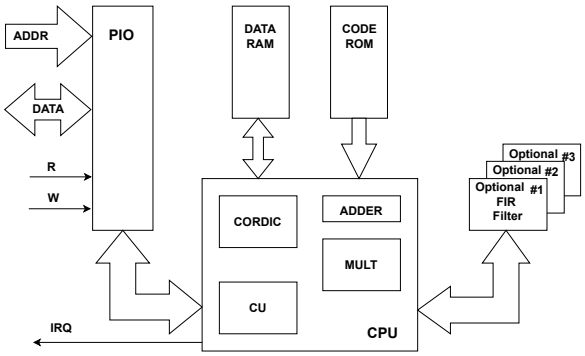

Fig. 3. Block diagram of FPGA CORE hardware.

- Optionally filter input data by decimation FIR filter. Former CAPSE MCU implementation uses a simple averaging FIR filter so we added this block as an option to the FPGA CORE. This block is currently not used.
- Preprocessing (DC offset removing, gain compensation, component transformation) of acquired data from ADCs.
- Compute angle of rotation α by using trigonometric $\arctan()$ function.
- Compensate non-idealities of analog part by piecewise linear correction of computed angle.
- Post-process output data (output data formatting, providing status information about quality of input data).

The FPGA CORE consists of several hardware blocks:

A. *Control Unit (CU)*

A simple CU is required for the control of datapaths implemented as separate external HW blocks. We selected a simple 16-bit Leros processor available as VHDL based soft/processor [4]. It has Harvard architecture with separate Program Memory (PM) and Data Memory (DM), small number ($\sim$20) of really used instructions and can be easily extended. The small and fully customizable Leros design is relatively slow but it is not a bottleneck for overall performance as main data processing tasks are executed by the external datapaths.

B. *Fixed-point 16-bit hardware multiplier*

We use a standard Booth's algorithm based fractional signed 16-bit fixed-point multiplier (MULT) as an external HW coprocessor for implemented CU. It uses input data scaled into fractional (-1,1) interval similarly as is done in a typical fixed-point digital signal processor. The MULT is connected to the external peripheral address and data buses of CU and provides 16-bit signed result (the high significant word) in fractional format. The CU has access also

to the low significant word of multiplication result, but due to proper scaling of data in implemented algorithms, it is not required in the final data processing algorithms. The MULT is implemented as a pure combinatorial circuit and provides result with latency of 1 CU clock cycle.

C. *Fixed-point 32-bit adder*

A 32-bit fixed-point adder (ADDER) is used as an additional external HW coprocessor for implemented CU. It contains also parallel barrel shifter for fast extraction of properly scaled 16-bit output words out of the input 32-bit ADC ones. This coprocessor enables fast removing of DC offsets from input data streams and selection of 16 most significant data bits after DC offset removing. The CDC ASIC uses a sigma delta ADC with up to 32-bit configurable output word-length, but in principle, any ADC with compatible word-lengths can be used.

D. *CORDIC coprocessor*

Implemented signal processing algorithm requires computation of trigonometric function

$$\alpha = \arctan\left(\frac{Y}{X}\right) \tag{1}$$

for evaluation of angle of rotation α in all 4 quadrants, where Y and X are represented as fractional fixed point numbers. A sequential CORDIC (COordinate Rotation DIgital Computer) algorithm [5] based fixed-point coprocessor with 16-bit input/output interface uses a modification of VHDL code from [6]. The CORDIC interfaces to the external peripheral address and data buses of CU and provides 16-bit resolution in full 2π radians angle range. The internal precision of computation and number of iterations of CORDIC are configurable before synthesis. Actually used values are 20 bits for the width of internal CORDIC datapath and 16 iterations (clock periods). These values ensure lower overall errors than required by the actual specification. The CORDIC coprocessor uses the same clock as CU, but in principle, we can use also higher separate clock signal.

E. *Parallel interface with interrupt handling (PIO)*

The PIO interface to an external MCU is implemented as a set of two separate small Dual-Port Memories (DPMs). DPM_1 is used for CU $\rightarrow$ MCU direction (CU can write only, MCU can read only) and DPM_2 is used for MCU $\rightarrow$ CU one (MCU can write only, CU can read only). Additionally, the CU has read only access to the PIO STATUS register. The DPM_1 has only four 16-bit words $DATA_{0-3}$ addressed by address bits ADDR(1:0) and mapped to the specific address locations. The DPM_2 has eight 16-bit words $DATA_{0-7}$ in order to support efficient transfer of 3 data channels from MCU to CU. The CU signalizes by interrupt line (sets IRQ) to the MCU automatically by writing to the specific DATA register in DPM_1. The MCU clears automatically pending IRQ by writing to the specific DATA register in DPM_2. The external MCU interface uses a set of standard signals: 16-bit ADDRES bus, 16-bit DATA bus, RD, WR and IRQ. The PIO block is mapped to the I/O space of CU. The CU can read actual status of IRQ (properly synchronized to the CU clock domain) that is mapped to the STATUS register.

We assume the MCU clock is not synchronous with the CU clock. For reliable cross domain clock crossing we use "Flancter" based hardware synchronization circuit [7] and HW controlled interrupt line.

F. *Optional decimation FIR filter*

A FIR filter can be used for additional filtration of input data streams in order to suppress input noise and increase input data resolution. Filtration is currently not yet implemented and optional simple averaging is computed by CU.

G. *Optional UART*

A simple Universal Asynchronous Receiver-Transmitter (UART) is actually connected to the CU. We use UART for communication with an external computer during development and debugging of complete HW and Firmware (FW) design. The UART will be removed from the final stable design.

IV. FIRMWARE OF THE PROPOSED FPGA CORE

The FPGA CORE functionality follows developed mathematical expressions already tested in the former experimental CAPSE MCU implementation. The experimental MCU implementation used floating-point implementation.

The proposed CAPASIC implementation maps these algorithms into digital FPGA CORE HW blocks with the aim to decrease overall HW complexity and maintain required precision of computation with small HW resource requirements. Hence, we use fractional fixed-point arithmetic in all implemented signal processing algorithm steps mapped into the FPGA CORE shown in Fig. 4.

The implemented algorithms are controlled by CU firmware stored in the PM of implemented CU. The PM contains less than 256 CU instructions that are encoded as a simple loop even without subprogram calls (although subroutine calls are also supported in Leros CU). The firmware is written in assembler of Leros CU. The instructions are used for mapping of intermediate data to/from described hardware coprocessors and reading (and waiting) of corresponding status information.

The implemented firmware supports also a "boot phase" during which all relevant parameters are downloaded from an external MCU via PIO interface to the DM of CU. These parameters include:

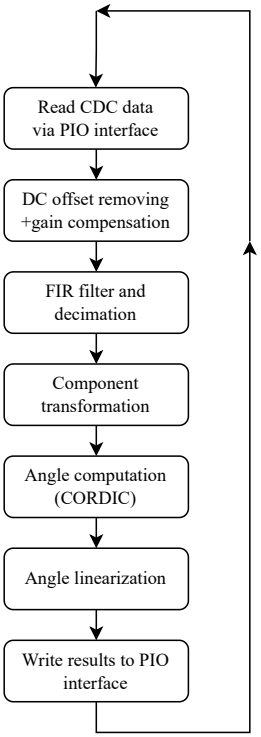

Fig. 4. Flowchart of implementd signal processing algorithms steps.

- Offsets and gains for compensation of errors of input analog sensor parts.
- System parameters used for monitoring status of input data. The firmware also monitors and reports in real-time (via status register) some fault conditions caused by fault input data.
- Calibration constants for simple piece-wise linearization. The firmware supports N_{seg}=8, 16, 32 and 64-point linearization.

Currently used Leros HW implementation supports up to 256 words of DM with memory words that are directly mapped as CU registers. The complete FW uses only 20 DM words for all implemented signal processing steps except for linearization routine. We implemented a simple linearization routine [8] with correction curve based on N_{seg} piece-wise linear segments. We adapted correction coefficients for usage of fractional fixed-point multiplier. In contrast to [8], we use calibration coefficients that minimize Minimum Min Square Error (MMSE) in each of N_{seg} segments. The linearization routine requires 2 additional DM words per one linearization segment. This limits support for max $N_{seg} = 64$-point linearization in available DM with 256 words.

V. EXPERIMENTAL RESULTS

We implemented all described blocks as parametrized VHDL code and we do not use specific FPGA HW blocks. E.g., usage of embedded FPGA multipliers was intentionally disabled in order to enable porting of developed VHDL code to the future ASIC design. We prepared VHDL testbench files for independent testing of all developed HW blocks in Modelsim simulator. Functional and timing simulations (after mapping to the low-cost MAX 10 FPGA) were performed in Modelsim to confirm proper functionality of implemented blocks. Mapping and placement & route to the specific FPGA HW were done by Intel Quartus CAE tool. The expected functionality in selected FPGA was confirmed. Prepared testbenches can be used for testing critical paths of implemented HW blocks in the target FPGA.

As the next step, all developed HW blocks were connected and mapped to the specified peripheral memory locations of Leros CU. The complete FPGA CORE with debug UART occupies 2200 Logic Elements (LEs) in the target MAX 10 FPGA hardware. This number includes also LEs used for storage of firmware that is mapped to the synthetized HW and not stored in embedded FPGA Block Memories (BMs). Resources required for implementation of specific coprocessors are shown in Table 1 (ADDER includes also barrel shifter, CU includes also stored FW).

Table 1. FPGA resources required for implementation of specific parts of developed FPGA CORE

MULT	CORDIC	ADDER	CU
425 LEs	444 LEs	300 LEs	512 LEs
0 BM	0 BM	0 BM	2 BMs

We used data acquired from analog sensor testbench developed in previous CAPSE project for extensive testing of developed custom processor digital hardware. The raw input data for angle rotation $\alpha \in \langle 0, 2\pi \rangle$ acquired by three 20-bit ADCs are shown in Fig. 5. The input raw data have slightly different offsets and require also different gains compensation for each of 3 processed ADC channels. The differences are caused by imperfections of analog sensor sections and they are decreased by algorithms implemented in CU firmware. The final computed angle errors for $N_{sec} = 8$ and $N_{sec} = 32$ are shown in Fig. 6.

We tested our complete custom processor also on DE10-Lite FPGA development board with CU clock generated by FPGA PLL clock block. The extensive hardware tests confirmed proper functionality of developed custom processor and its long-term reliable and stable operation.

Complete data processing (without decimation filtration) shown in Fig. 4 takes 212 CU clock cycles (actual FW has 238 instructions). For an expected sampling frequency F_s = 10 kHz the CU system clock frequency of

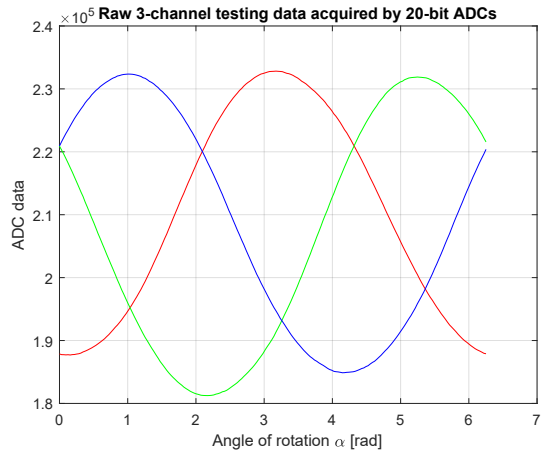

Fig. 5. Three channel testing data acquired from analog testbench by 20-bit ADCs during complete 2π radian rotation.

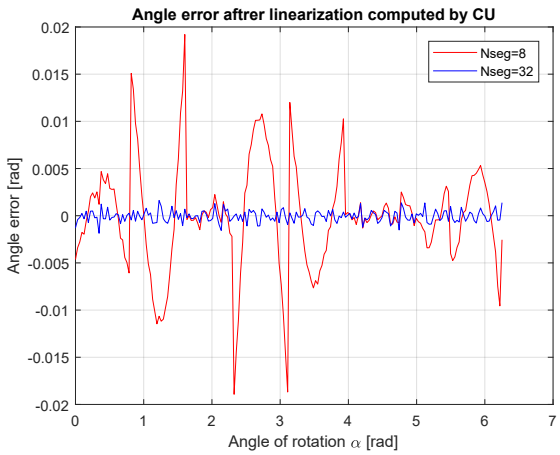

Fig. 6. Angle error of actual angle position α computed by developed processor in fixed-point arithmetic.

only $\sim$2.2 MHz is required. This is significantly lower frequency than the one required for previous MCU implementation. The overall precision of computed angle α by digital part is better than 13 bits that was required according to the specified design requirements.

VI. CONCLUSIONS

We described architecture of a small custom soft processor for proprietary capacitive angle sensor written in synthesizable VHDL. The final custom processor occupies $\sim$2100 LEs (without debug UART) and uses only 2 BMs in the target FPGA. The complete firmware has less than 250 CU instructions. Developed processor HW and FW can replace standard MCU used in former CAPSE prototype with significantly lower clock frequency in comparison with former MCU implementation. We mapped current version of developed processor to the specific Intel FPGA and fully tested in Intel Quartus and Modelsim CAE tools as well as in the target off-the-shelf FPGA development board. We can use prepared set of VHDL testbenches for efficient testing of developed processor HW and FW with acquired real data records in further stages of development. The complete design provides better than required 13-bit precision in 2π radians angle range. By changing internal parameters of implemented coprocessors (before synthesis) we can increase overall precision at the expense of additional HW resources. We will use our stable and tested VHDL code of developed custom processor as the main signal processing block in further CAPSE sensor development in the actual CAPASIC project.

ACKNOWLEDGMENTS

This work was supported also by the Science Grant Agency of the Slovak Republic (project No. 1/0413/22) and by the Slovak Research and Development Agency under Contract No. APVV-22-0400.

REFERENCES

[1] X. Fan, Z. Yu, K. Peng, and Z. Chen, "A compact and high-precision capacitive absolute angular displacement sensor," *IEEE Sensors Journal*, vol. 20, no. 19, pp. 11 173–11 182, 2020.

[2] "CTRL Ltd." http://www.ctrl1.eu.

[3] *ESS214D Radiation tolerant Capacitive Sensor Signal Conditioning IC*, ES Systems, 2020, rev. 1.

[4] M. Schoeberl, "Leros: A tiny microcontroller for FPGAs," in *Proc. of 21st International Conference on Field Programmable Logic and Applications*, ser. FPL, Sep 5–7, 2011, pp. 10–14.

[5] P. K. Meher, J. Valls, T. Juang, K. Sridharan, and K. Maharatna, "50 years of cordic: Algorithms, architectures, and applications," *IEEE Trans. on Circuits and Systems*, vol. 56, no. 9, pp. 1893–1907, 2009.

[6] "VHDL-extras library," https://github.com/kevinpt/vhdl-extras.

[7] R. Weinstein, *Application Note: The "Flancter"*, http://fpgacpu.ca/fpga/Flancter_App_Note.pdf, Memec Design, July 2000.

[8] *ZMID4200 Calibration and Linearization Manual – Analog Output*, https://www.renesas.com/us/en/document/mas/zmid4200-manual-cal-linearization-analog-out, Renesas, October 2020, rev 5.0-1.