Ročník 2012 electroscope Číslo VI

Serial IIR Filter Structure Generator for ASICs

M. Pristach, L. Fujcik
Department of Microelectronics, FEEC, BUT Brno, Technická 10, 616 00 Czech Republic
E-mail: xprist00@stud.feec.vutbr.cz, fujcik@feec.vutbr.cz

Abstract:

The paper presents generator of an infinite impulse response (IIR) digital filter structure for implementation in application specific integration circuits (ASICs). The paper describes the filter architecture with serial calculation. The serial architecture utilizes one shared multiply and accumulate (MAC) unit in order to achieve minimal area on chip. Software in C++ language was written for automatic filter generation. The software generates fully synthesizable VHDL description of filter, batch file for simulator and test-bench file for automatic filter verification from the filter specification file.

INTRODUCTION

Infinite impulse response (IIR) filters are different from finite impulse response (FIR) that are often using in digital signal processing. The IIR filters used feedback in contrast to FIR – the output sample depends on previous input and output samples. The IIR filters are very efficient. Nevertheless, it is compensated by the complicated structure and theirs design is harder. The IIR filters require noticeably fewer multiplications per output sample to achieve same frequency response. It means that IIR filters can be very fast and can operate over much higher sample rates then FIR filters. [1]
The aim of this project was to create a universal architecture of IIR filter with serial processing of the input data. Currently, one filter structure is supported – Direct-Form II, Second-Order Sections [2]. This structure uses a cascade of second order filters (often called as biquad) [3] as is shown in figure 1.

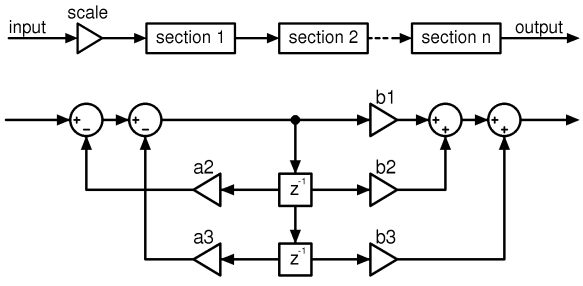

Fig. 1: Structure of Direct-Form II, Second-Order Sections filter and detail of one section

The main advantage of this structure is small number of registers. The second-order section filters are easier to design, have better frequency stability than higher-order filters and have lower sensitivity on coefficient quantization due to usage of the cascade structure. The second-order section filters also support data word scaling to reduce the possibility of the overflow effects.

STATE OF THE ART

Hardware representation of various filters can be directly generated by the Matlab, but for IIR filters only parallel architecture which contains a large number of adders and multipliers is supported. This architecture can process input data and calculate output sample in one clock period but it is not suitable for usage on a chip because it takes a lot of area. On the other hand, serial architecture uses only one adder and one multiplier. This architecture has longer calculation time then parallel architecture. This is not critical for use filters on chip because the chip usually works on higher frequency then the processed frequency band.

ARCHITECTURE OF THE FILTER

The filter architecture is based on a finite state machine with data path (FSMD) [4]. The filter consists of three parts – three blocks of memory, one multiply and accumulate unit and a controller. The memory blocks consist of two banks of read-write memory for data and one bank of read-only memory (ROM) with coefficients. The controller provides generation of control signals, e.g. write enable signals, memory addresses and multiplexer select signals.
The calculation is executed sequentially; every clock cycle the multiplication is executed and the result is stored in the accumulator. Calculation of one second-order section response takes six clock periods. Calculation of the whole filter response is given by the number of sections. In some cases when certain coefficients are zero, the calculation can be optimized and the response time is shorter. The block diagram of designed filter architecture is shown in figure 2.

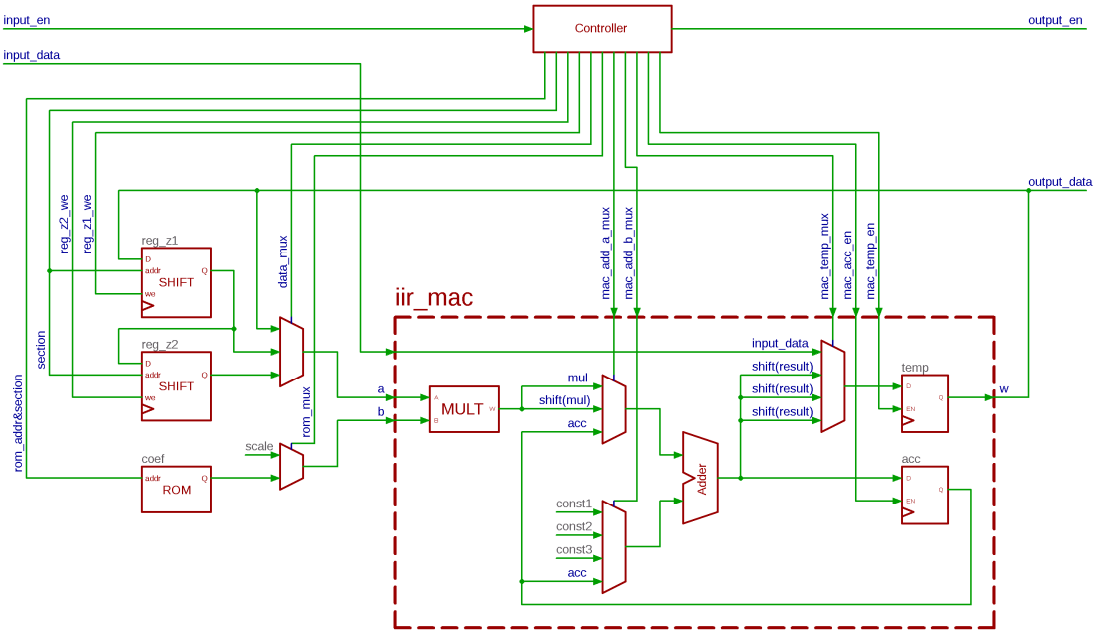

Fig. 2: Block diagram of filter architecture with shared multiply and accumulate unit

SOFTWARE

Software for automatic generation of filter hardware representation was written. The software uses a file with filter description generated by Matlab FDATool. By this tool, users can design a filter complying with their requirements, e.g. filter type (low pass, high pas, band stop, ...), bandwidth, attenuation, ripple or internal parameters of the filter. Then Matlab designs this filter, calculates its coefficients and the user can save this information in text format into a FCF file. Developed software reads this file and generates optimized and fully synthesizable VHDL description of filter.

The software was written in C++ language and provides two interfaces – command line and graphical (GUI). Graphical interface was created by the QT framework. Screen of the main window is shown in figure 3. Current version of the software supports filters with up to 128 sections, unlimited word length, with round modes *nearest*, *floor* and *ceil* and with both overflow modes *wrap* and *saturate*.

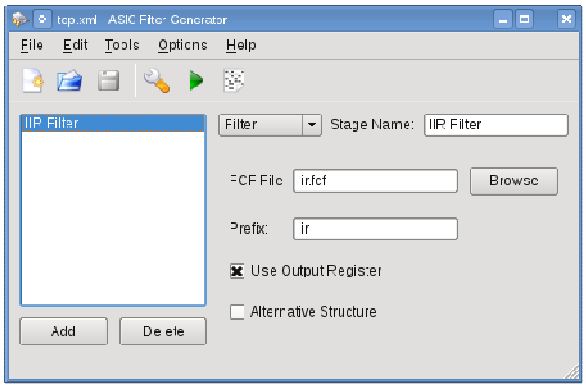

Fig. 3: Main window of the developed software

IMPLEMENTATION AND RESULTS

The hardware description of various filters was synthesized by the Cadence Encounter RTL Compiler RC9.1.203. The results of synthesis for technology AMIS 0.35µm (I3T25 at 3.3V; 25°C) are shown in table 1 and table 2 where the utilization of area, maximum clock frequency and power consumption are compared. Table 1 displays results of parallel IIR filter architecture available from Matlab HDL Coder; table 2 displays results of serial IIR filter architecture from developed software. All filters have width of input, output and coefficient of 16 bits, width of accumulator of 34 bits, round mode nearest and overflow mode wrap. Number of gates is computed with respect to area of 2-input gate which performs the logical NAND function ($38.88\mu m^2$) [5].

A frequency constraint of 1MHz was used for timing and power consumption analyses. Synthesis was optimized to area minimization with medium optimization effort. It can be observed on timing analysis results that the synthesizer did not optimize the design for timing constraint because of sufficient timing slack.

The results of power consumption analysis are only approximate, real value are based on the processing input data rate (on the input samples frequency). When no data are processing, the power consumption is almost zero because CMOS technology is used. Presented values of power consumption are for total power (dynamic + leakage power). These values are almost same as dynamic power consumption; leakage power is almost zero for this technology.

Tab. 1: Synthesis results of parallel IIR filter architecture (from Matlab)

Filter order	Total area		Logic area		Sequential area		Frequency	Power
[-]	[µm^2]	[gates]	[µm^2]	[gates]	[µm^2]	[registers]	[MHz]	[mW]
2	146098.08	3758	132697.44	3413	13400.64	47	25.030	37.707
4	234174.24	6023	211649.76	5444	22524.48	79	16.460	68.995
6	364720.32	9381	333046.08	8566	31674.24	111	11.804	96.555
8	504221.76	12969	463449.60	11920	40772.16	143	9.161	139.267
10	605348.64	15570	555452.64	14286	49896.00	175	7.260	152.487
12	639563.04	16450	580335.84	14926	59227.20	207	6.615	150.083
14	885375.36	22772	817231.68	21019	68143.68	239	5.531	212.015
16	958197.60	24645	880904.16	22657	77293.44	271	5.011	187.242

Tab. 2: Synthesis results of serial IIR filter architecture (from developed software)

Filter order	Total area		Logic area		Sequential area		Frequency	Power
[-]	[µm^2]	[gates]	[µm^2]	[gates]	[µm^2]	[registers]	[MHz]	[mW]
2	109758.24	2823	85795.20	2207	23963.04	86	31.525	7.517
4	126178.56	3246	92910.24	2390	33268.32	119	31.093	6.631
6	140097.60	3604	96526.08	2483	43571.52	152	31.095	10.14
8	150634.08	3875	97925.76	2519	52708.32	184	31.353	9.123
10	162933.12	4191	100764.00	2592	62169.12	217	30.018	11.66
12	176502.24	4540	105196.32	2706	71305.92	249	28.544	7.450
14	185159.52	4763	104729.76	2694	80429.76	281	30.069	9.195
16	197769.60	5087	108216.00	2783	89553.60	313	29.707	12.07

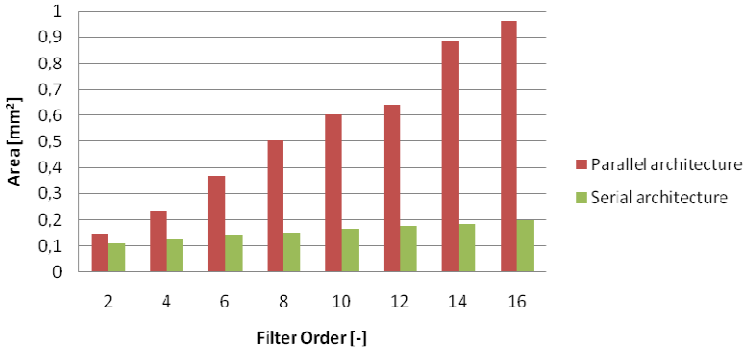

Fig. 4: Comparison of area for serial and parallel filter architecture

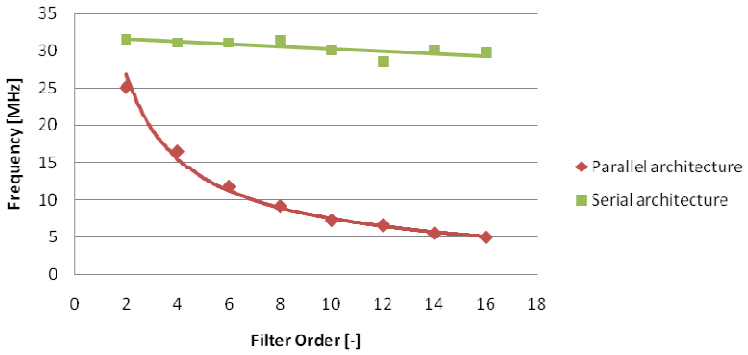

Fig. 5: Comparison maximum clock frequency for serial and parallel filter architecture

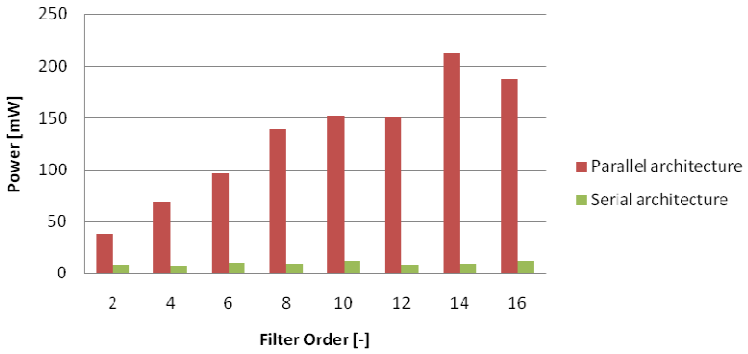

Fig. 6: Comparison power consumption for serial and parallel filter architecture for frequency 1MHz

Comparison of both architectures with respect to area is shown in figure 4. The graph shows a significant increase of the area in the parallel architecture with increasing order of the filter. Serial architecture has by the assumption significantly smaller area. Area of combination logic in the serial architecture varies with filter order only minimally. The number of registers in the serial architecture increases linear with filter order and registers represents the main amount of chip area in filters with higher order.

Comparison of both architectures with respect to maximum operating frequency is shown in figure 5. Serial architecture has approximately constant operating frequency that varies with filter order only minimally. This is the result of that the architecture of filters does not change – order of the filter affects only the memory size and width of the address bus.

The maximum operating frequency of the parallel architecture decreases exponentially with order of the filter. This decrease is determined by the architecture where all second-order sections are connected in series – critical path passes through all stages of the filter. The frequency of the parallel architecture can be increased by using pipelining techniques, which will have an impact on the increase of chip area.

Comparison of both architectures with respect to power consumption is shown in figure 6. Serial architecture has approximately constant power consumption; these values slightly fluctuate around 10mW. The power consumption of parallel architecture increases with order of filter. This increase of power is given by the larger chip area where most of the area is occupied by combinational logic (approximately 90% of total area). The power comparison does not take into account that the parallel architecture within a specified period process much more input samples than serial architecture. The suitable test case based on really processing data rate must be used for better comparison. The VCD file from simulation should be generated and used in power consumption analysis.

CONCLUSION

The paper presents developed architecture of serial IIR filter and software for automatic generation of filter description in VHDL. Developed architecture is optimized for area and is mainly suitable for use in ASIC design that does not require high speed of input samples processing. Developed software is suitable for fast design of filters while time-consuming hand-writing of HDL code is omitted. User can focus on better design of the filter in Matlab and gets hardware parameters (speed, area) of the designed filter very quickly.

ACKNOWLEDGMENTS

The research has been supported by project Prospective applications of new sensor technologies and circuits for processing of sensor signals No. FEKT-S-11-16 and by the Czech Science Foundation as the project No. GA102/11/1379.

REFERENCES

[1] Lyons, R. G., *Understanding Digital Signal Processing, 2nd Edition*, Prentice Hall, 2004, ISBN 978-0-13-108989-1.

[2] Stearns, S. D., Hush, D. R., Digital Signal Processing with Examples in MATLAB, 2nd Edition. CRC Press, 2011, p. 516, ISBN 978-1-439-83782-5.

[3] Matlab Documentation, DSP System Toolbox, Fixed-Point Filter Properties [online]. 2011 [cit. 2011-12-10]. Available from WWW: <http://www.mathworks.com/help/toolbox/dsp/ref/f8-109180.html>.

[4] Chu, Pong P., FPGA prototyping by VHDL examples. Wiley-Interscience, 2008, p. 468, ISBN 978-0-470-18531-5.

[5] Standard cell libraries: amis350ucascc_Rev7.9_Mar_18_2010. ON Semiconductor (formerly AMIS), 2010, p. 994.