UTILIZATION OF MATLAB FOR THE LOGARITHMIC PROCESSOR DEVELOPMENT

M.Licko, R.Matousek, Z.Pohl
UTIA Prague, Czech Republic

Introduction

Today's microprocessors are using the floating-point units for the real data type manipulation. Basic research has indicated that the logarithmic number system can offer new possibilities of implementation for some algorithms. Especially for the algorithms widely using multiplication, division and square root operations.

Consequently an European project based on the idea of the floating point numbers was started [1]. The goal is logarithmic microprocessor ASIC. One of the method used for the evaluation is FPGA prototype and C-to-hardware synthesis design based on Handel-C [2] suite.

C-like syntax of Handel-C language allowed us to use Matlab's rapid prototyping too. All the functions of the logarithmic microprocessor core (high speed logarithmic arithmetic unit - HSLA) are bit exact emulated in Matlab.

We are using two hardware platforms for the FPGA prototyping. Strong, powerful and expensive ADC-RC1000 [3] and cheaper and less strong XSV-800 [4]. We have designed the first prototype version of the logarithmic microprocessor on XSV-800. The development, evaluation and demonstration was realized with the utilization of Matlab and this article provides short summary of it.

System configuration

The development was realized especially under the Windows98/NT. Matlab environment was used for the DOS/Win32 processes control, development and debugging of algorithms, result's evaluation and rapid graphical user interface development. There is a system configuration and three main development tools on the Fig. 1.

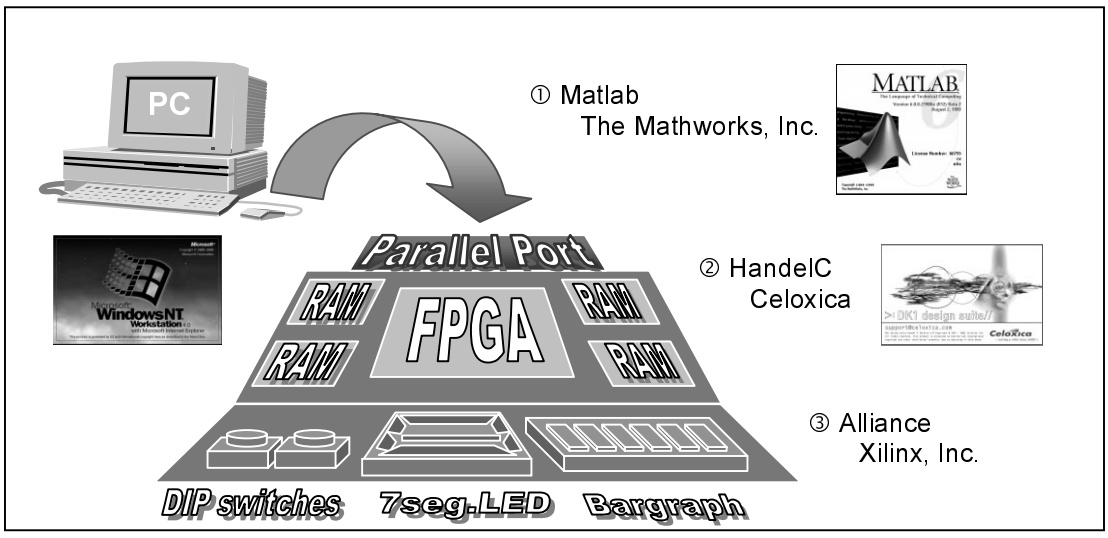

Fig. 1 System configuration and used

The XSV-800 board is based on the Xilinx Virtex FPGA and it is connected to the PC by the parallel port. Board is capable to process video and audio signals, has two independent banks of the 512Kx16 15ns SRAM, DIP switches, LED digits, LED bar-graph and variety of interfaces for the communication, including USB port and PS/2 port. But the peripheral protocols are not implemented directly on the board. Everything what is necessary must be implemented directly in the FPGA. At this moment we have implemented memory interfaces, VGA, PS/2 keyboard and parallel port interface.

The main development tools are stated on the Fig. 1. Complete development path needs complementary tools. They are necessary for the FPGA design. HandelC serves for the translation from the C-like code to the meta-file for the FPGA. HandelC is based on the theory of communicating sequential processes CSP and historically is built on OCCAM language. Alliance tool serves for the target code generation, it is so called 'bit stream'. All tools are summarized in the next table Tab. 1.

Name	Corporation	Purpose / Utilization
① Matlab 6 R12	Mathworks	Rapid prototyping and project management role
② HandelC v2.1	Celoxica	Simulator and compiler of the meta file for FPGA
③ Alliance 3.3.06i	Xilinx	Generate the target code format for FPGA from HandelC metafile

Tab. 1 The main software development tools

Logarithmic processor overview

Under the project [1] a basic software tools for the European Logarithmic Microprocessor ELM were used. They are specified in the following table Tab. 2.

Name	Version	Purpose / Utilization
ELMASM	2.0	ELM assembler
ELMLNK	2.1	ELM linkage editor
ELMSIM	2.3	ELM simulator

Tab. 2 The main software development tools

The processor is built round the 32bit high-speed logarithmic arithmetic (HSLA). This is our logarithmic equivalent for the FP unit. A special instruction set was designed and implemented.

The prototype in HandelC uses an external memory – SRAM for the ELM's program and for auxiliary tables.

Flowchart of the cycle of development

An assembler, linker and the simulator of the ELM processor were used. There is a build and boot process of ELM on the Fig. 2. Each step is controlled from Matlab.

Firstly the auxiliary tables for HSLA are downloaded to SRAM through FPGA on XSV-800 board. Secondly the program for ELM processor can be modified and downloaded too. The whole process is shown on the right column where an assembler ASM and the linker LNK are called. Than Matlab is used to generate a header file (*.h) for HandelC. There is the code for ELM processor. In next steps the applications from Fig. 1 are called (② HandelC and ③Alliance) to translate the code to the bit file for FPGA. In the end the ELM processor is downloaded and starts execution on target platform.

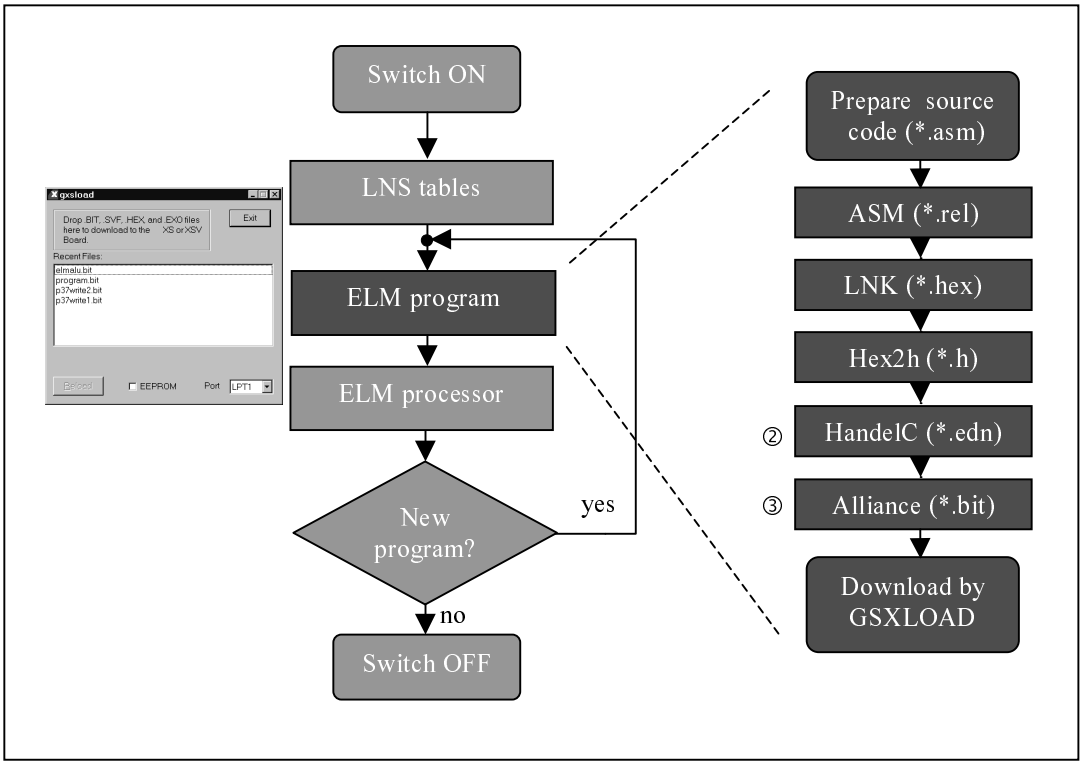

Fig. 2 Flowchart of the cycle of development

Demonstration of function of ELM

There is a demo presenting the ELM at work on logarithmic numerical application. Microprocessor gets the integer index value and gives back the integer value of the sinus function. Sinus is computed in logarithm. There are steps for conversion to and from the logarithmic domain. Each time a button on the XSV-800 board is pressed a new value for the next index in the order is computed. Microprocessor communicates with Matlab via the parallel port and the appropriate index is marked on Matlab's figure (Fig. 3).

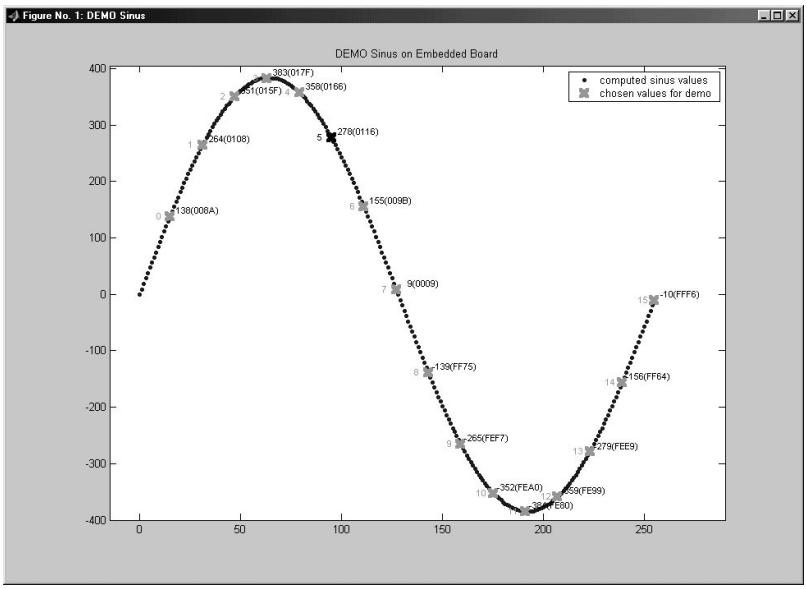

Fig. 3 Shown computed sinus values

There is a Simulink's model running on the PC's side. It supports parallel port communication via our interface and Matlab's figure handling. The value of the index five is pointed at this moment.

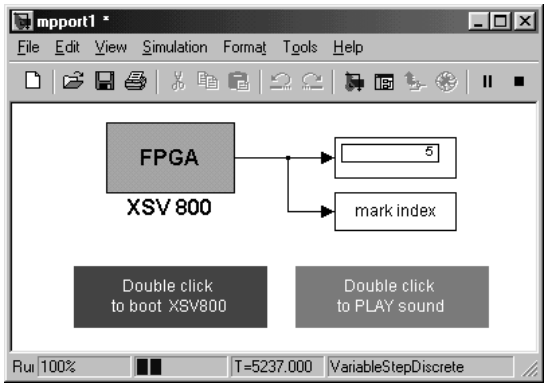

Fig. 4 Simulink's model

There are many possibilities how to extend the functionality of ELM. Suitable application areas include signal processing, digital control, graphics processing and numerical analysis. For example an audio interface is implemented in this sinus demo too. One part of the FPGA HW works on the audio processing while another counts the sinus values. This way ELM processor utilizes FPGA's parallel processing strengths.

Parallel port demonstration function

There is another simple demo to present and test the parallel port interface and Simulink's possibilities. There are PC's inputs as segments in Simulink's block or slider in Matlab's figure to send a value on the parallel port. The appropriate segment on the XSV-800 board will shine. And similarly the pressed switch on the XSV-800 board is indicated in the Simulink's model.

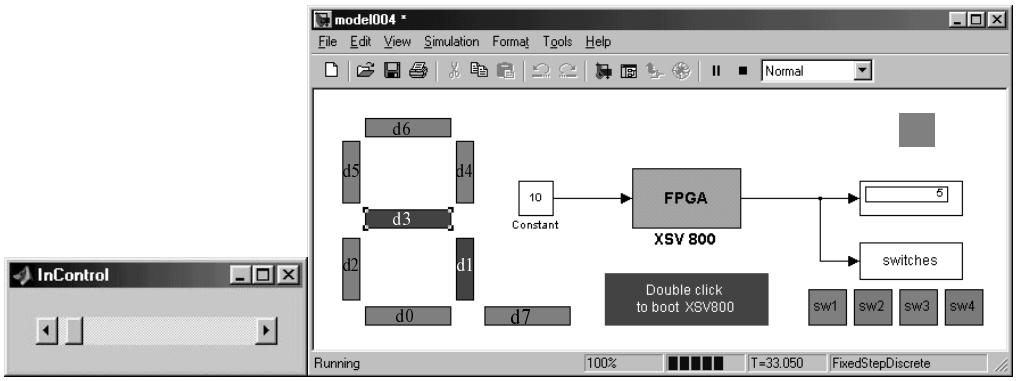

Fig. 5 Simulink's and parallel port's functions presentation

Conclusion

Matlab helped us to shorten the debugging process of the first version of the logarithmic microprocessor design. All the source codes exist as MEX files, too. It's useful for us, because we have a bit exact hardware emulation by this way.

We are familiar with Matlab figure's and Simulik model's objects. Matlab helped us to get powerful presentation and user interface for our embedded FPGA design.

References

[1] http://napier.ncl.ac.uk/HSLA

[2] http://www.celoxica.com

[3] http://www.alphadata.co.uk

[4] http://www.xess.com

Acknowledgements

This work was supported by the Ministry of Education of the Czech Republic under Project LN00B096.

Contact

Centre for Applied Cybernetics
Department of Signal Processing, Institute of Theory and Automation
Academy of Sciences of the Czech Republic (UTIA AV CR)
Pod vodarenskou vezi 4, 182 08, Prague 8, Czech Republic
Email: Licko@utia.cas.cz
www: www.utia.cas.cz/ZS
dce.felk.cvut.cz/cak
www.c-a-k.cz