Andrei-Sorin GHEORGHE

Address:
Date of birth: 31/12/1985
Phone: +40 752 029 892
E-mail: ghiga_andrei@yahoo.com

Desired position:
Digital, analog or mixed signal integrated circuits designer

Work experience

07/2009 – 02/2010	**AVITECH – Electronics Engineer – Service Division** • Component level repair of Audio-Video equipment: Plasma displays, LCDs, Video projectors, Audio amplifiers, Home Cinema Receivers, Studio mixers, Microphones, LED or laser light projectors. • Video projectors maintenance.
07/2007 – 07/2009	**AVITECH – Technician – Service Division** • Component level repair of Audio-Video equipment. • Video projectors maintenance.

Education

Since 10/2009	Master in **"**Advanced Microelectronics**",** at the Faculty of Electronics, Telecommunications and Information Technology, Bucharest
10/2005 – 07/2009	Faculty of Electronics, Telecommunications and Information Technology, Electronics engineering and Telecommunications domain, majoring in Microelectronics, Optoelectronics and Nanotechnologies, Bucharest Graduation average: 9.18
09/2001 – 06/2005	National College of Informatics "Tudor Vianu", Bucharest Graduation average: 9.75

Projects

work in progress	**Dissertation project**: Digital oscilloscope with data logger for I2C, SPI, RS232, Wiegand and other common buses.
07/2009	**Diploma project**: Savage16 - 16bit RISC Microprocessor
work in progress	Team project: Intelligent Autonomous Slot Car for the Freescale Race Challenge 2010 international contest, using a ColdFire family microcontroller
11/2009	Wav file audio player from SD card, using the Savage16 microprocessor, designed in its assembly language
10/2009	Assembler and programmer for the Savage16 microprocessor, developed in Visual C++
10/2009	DC/DC Converter, in unisolated Boost topology
06/2009	Blog theme based on the Wordpress platform, developed in HTML and PHP
02/2009	Precision millivoltmeter with digital display, using the C8051 microcontroller from SiLabs

10/2008	Digital integrated circuit layout: MMC 4518, using Ledit
07/2008	SmartCard reader for room access control using the ATmega32 microcontroller, from Atmel
11/2007	Electronic control for the integration of a miniPC in a piece of furniture, using the ATmega32 microcontroller from Atmel
05/2007	Text-mode VGA terminal with keyboard input and monitor output, designed in FPGA
04/2007	Pong game with multiplayer support, designed with two interconnected FPGAs
07/2004	Computer assisted teaching module, made in Flash
08/2002	Various DOS games (Tetris, Worm, Cars), programmed in Pascal

Technical knowledge

Programming languages C, C++, Pascal, HTML, PHP, ActionScript

HDL languages Verilog

Known software Xilinx IDE, ModelSim, Matlab, Labview, Visual C++, LEdit, Virtuoso, Orcad Capture, Pspice, LTspice, CodeWarrior IDE, Silicon Laboratories IDE, CodeVision AVR, Macromedia Flash

Technical knowledge Data protocols: RS232, I2C, SPI, WIEGAND, RC
Theoretical knowledge which covers the following areas:

- Microelectronics (design, simulation, testing, layout)
- Optoelectronics
- Object oriented programming (C++)
- Microcontrollers and Embedded Software
- System modeling (Matlab / Labview)
- Microsensors and MEMS systems
- Power electronics (Switching mode power supplies and Switching devices)
- Digital signal processing (Filters)

Personal skills and competences

Languages English - advanced level
French - medium level
Spanish - beginner lever

Computer skills Windows, Office (Word, Excel and PowerPoint), Hardware

Abilities Strong ambition, logical thinking, very good memory, great capacity to understand and constant desire to learn

Qualities Punctuality, sense of humor, dynamism

Driving license B category, since 2005

References

Ing. Laurențiu Teodorescu, Service Manager, AVITECH
e-mail: laurentiu.teodorescu@avitech.ro
phone: +40 740 109 666

Appendix 1: Projects Descriptions

1. **Dissertation project**: Digital oscilloscope with data logger for I2C, SPI, RS232, Wiegand and other common buses.

The project consists in implementing a Savage16 microprocessor based digital oscilloscope. The microprocessor will be implemented using a Virtex FPGA development board from Xilinx, which provides a working speed of 300 MHz required to obtain the appropriate sampling rates based on the bus speeds that will need to be investigated.

The oscilloscope will display the captured signals on a mini LCD monitor and will have a simple on screen display menu and input from a keyboard consisting in a few push-buttons. All adjustments will be made digitally.

Data Logging option allows common data bus monitoring, automatic level conversion (5V <-> 3.3 V <-> 1.8 V) and automatic logic transitions interpretation. Thus, the monitor will directly display the corresponding hexadecimal codes of the captured digital signals, decoded according to the implemented protocols.

The user will have the option of saving the captured data on a SD card, for him to be able to download it to a PC and perform various other operations with them.

The code describing the structure of the Savage16 microprocessor is written in Verilog HDL and the microprocessor programming will be done in assembly language.

2. **Diploma project**: Savage16 – 16bit RISC Microprocessor.

The project involved the creation of a fully functional microprocessor, starting from point zero. For implementation the modified Harvard architecture and an internal structure of a five-level pipeline were chosen.

The first step was to design the instruction set. A total of 28 arithmetic and logical instructions, 14 data transfer instructions, 15 jump instructions 3 control instructions were implemented. Most instructions are executed in one clock cycle, except jump instructions, which require two clock cycle and multiplication and division instructions, which are implemented using serial algorithms, and require 11, respectively 19 clock cycles to complete.

The second step was to describe the elements of microprocessor architecture. It has separate hardware implemented program and data stacks, a program memory map of 256 kB (organized as 64K x 32b), a data memory map of 128 kB (organized as 64K x 16b), four 16bit input ports, four 16bit output ports, prioritized interrupt system (12 external interrupt requests on dedicated terminals + 4 internally generated interrupt requests) and a set of 16 general purpose registers, each 16bit wide.

The next step was to design the internal structure of the microprocessor, at basic modules and logic gates level. Five pipeline stages (Fetch, Decode, Read, Execute, WriteBack) and hazard detection mechanisms have been implemented. Data forwarding to avoid hazards and simplify the programmer's work by avoiding the need for manual interleaving of NOP instructions between dependent instructions have also been implemented.

The previous designed blocks were described in Verilog language and simulated in ModelSim environment. Then, the code was synthesized and implemented in a Spartan3 FPGA, produced by Xilinx company. The maximum clock frequency at which the microprocessor operated correctly in this FPGA is around 70 MHz, and is strongly influenced by the modest performances of the Spartan3 family FPGAs.

The microprocessor was given the name Savage16 and is now fully functional, currently being used in my other projects.

This diploma project received the highest mark from the evaluation committee.

3. Team project: Intelligent Autonomous Slot Car for the Freescale Race Challenge 2010 international contest, using a ColdFire family microcontroller.

The project consists in designing a software that provides the intelligence necessary for a slot car so that it obtains the best time for the completion of 10 laps of a competition track, unknown until the day of the competition.

The car has only one sensor, a three axis analog accelerometer, which provides the information necessary to build the circuit map. In the first lap the car drives with a properly chosen speed so that the detection threshold is above the sensitivity of the accelerometer. The data from the accelerometer are sampled by an ADC, filtered and then processed using appropriate DSP techniques. At this point the Matlab program came very handy, making it easy to design all the necessary digital filters. The final results are written on a miniSD card.

Once the circuit map is built, for the following 10 laps the car will have to accelerate and brake in accordance with the information stored on the SD card. These 10 laps are timed and the team with the lowest time wins.

The electronics of the car were offered by the Freescale company as a kit, as parts and an unpopulated PCB. Soldering all those parts was a difficult task of the contest, many of the parts being SMD components and having very tiny capsules with many pins. Also, the task to mount the PCB in the car and make the appropriate changes in the chassis was also given to each team.

The first eliminatory stage was held on January 11th, 2010, with my team advancing further in the competition. The next stage will take place on April 1st, 2010, after which only the best 5 teams in the faculty will advanced further in the competition.

For this project we had to familiarize with the ColdFire microcontrollers architecture and the Freescale CodeWarrior IDE development environment.

4. Wav file audio player from SD card, using the Savage16 microprocessor, designed in its assembly language.

This project's main purpose was the thorough testing of the Savage16 microprocessor and the assembler developed for it.

In addition to the microprocessor, I have implemented in the FPGA the following peripherals: a 16bit SPI controller to communicate with a DAC, an 8bit SPI controller to interface the SD card and a RC modem for decoding the data received via infrared from a remote control.

The assembly language code decodes the FAT32 file system, according to the technical specifications published by Microsoft. Subroutines are implemented to initialize the SD card, browse the directory and file structure, decode the allocation tables and read the wave files.

For the audio output stage I designed a PCB that features a DAC with a resolution of 12 bit, a 4th order Sallen-Key topology Butterworth low-pass filter and a 6W audio amplifier with analog volume control. The data decoded by the microprocessor is sent to the DAC, at a sampling frequency of 44.1 kHz, generated using the interrupt system at which I have connected a timer described in Verilog code.

Tracks can be changed via a remote control, its decoder being also connected to the interrupt system. When a button is pressed, the RC decoder generates an IRQ, and a proper interrupt service routine changes the song, stops the playing or resumes playing.

During and due to the implementation of this project I have corrected a few major bugs in the microprocessor, most of them related to the interrupt system and its synchronization with the Fetch mechanism.

The resulting audio quality is very good, the DAC and the 4th order smoothing filter generating a very warm sound and no disturbing sound artifacts.

5. Assembler and programmer for the Savage16 microprocessor, developed in Visual C++.

This project allows easy programming of the Savage16 microprocessor. It converts a text file with mnemonics and operands into a binary file consisting of machine code, and then sends this code on the RS232 interface to the programming module implemented in the microprocessor.

The assembler is done using the new powerful string classes available in Visual C++ 2008. It:

- identifies lines starting with or containing the characters "//" and ignores everything after them (comments); if there's something before the characters then it does not completely remove the line
- has defined an "alias" directive that can assign user names to the internal components or define numeric constants
- verifies that the user didn't define two identical aliases
- deletes blank lines or lines that consist solely of spaces and tabs; it also eliminates any illegal characters
- allows the use of a single mnemonic for multiple instructions with the same role but with operands of different types (overloading)
- finds all the jump labels in the program and calculates the corresponding jump addresses
- checks that no two jump labels are defined twice
- announces the user when it detects an unknown instruction mnemonic

- checks the syntax for each type of instruction in part and generates an error when the operands are too many, too few, when used a register as an operand instead of a port or vice versa, when an immediate data exceeds 16bits etc
- automatically detects whether your numbers are in decimal or hexadecimal
- with each error generated it shows exactly the line that contains the error (the line in the original file, with spaces and blank lines for rapid identification in the text editor)
- it opens the COM1 port for microprocessor programming
- sends all the resulted machine code through the RS232 interface to the microprocessor's program memory
- downloads the data memory back through the RS232 interface and writes a log file (memory dump)
- notes any step taken in a log file that will contain extracted aliases, jump labels, instructions processed (possibly error code) and the memory dump.

6. DC/DC Convertor, in unisolated Boost topology.

This converter is a switching mode power supply very useful if we have to supply an electronic device that requires a higher voltage than is available (e.g. from a 12V car battery).

The scheme implemented is very simple, with a rectangular signal generator using an operational amplifier with positive feedback, and control without an opto-coupler, using a second operational amplifier that will command the first. At the non-inverting input of the second operational amplifier we apply a fraction of the output voltage and on the inverting input we apply a reference voltage.

The TL431 integrated circuit was chosen to generate the reference voltage. It is set up to generate an output voltage of 2.5V fixed.

The output of the converter consists of conventional Boost topology, with a coil, a bipolar transistor as a switching element, a very fast recovery switching diode and an electrolytic capacitor for filtering.

The minimum supply voltage is 10V and the maximum output voltage is approx. 55V.

7. Blog theme based on the Wordpress platform, developed in HTML and PHP.

This project consisted in setting up a Wordpress blog platform theme.

For the graphical design of this project I had to learn the basics of HTML programming and CSS file structure. The basic architecture was inspired from a theme taken from the Yahoo 360 blog platform, service closed by Yahoo on July 13th, 2009.

For the user interface I had to learn basic PHP programming and the Wordpress platform internal set of functions. Tutorials available on the Internet have been very helpful.

Finally, the PHP code was integrated into the HTML code and linked to an SQL database.

8. Precision millivoltmeter with digital display, using the C8051 microcontroller from SiLabs.

This project was basically aimed at familiarizing me with the C8051 microcontroller from SiLabs, the development board provided by them and the Silicon Laboratories IDE development environment.

Sampling is done using the high-performance digital to analog converter within the C8051 microcontroller. This is a 24-bit ADC, preceded by a PGA amplifier with adjustable gain up to 128. The input is differential and the output is decimated and filtered using a Sinc3 type filter. Equivalent noise is almost nonexistent.

To display the measured voltage, I used a 7 segments and 4 digits display, produced by Vishay. This is a common-anode display type, to minimize the need for microcontroller ports. I used a time multiplexing method controlled by timers and the interrupt system.

9. Digital integrated circuit layout: MMC 4518, using Ledit.

For the BTM laboratory we had to realize a complete digital integrated circuit layout. I chose the MMC 4518 integrated circuit, a product of Microelectronica, which is a four-bit decade counter with reset.

First, I had to get familiar with the Ledit environment, its DRC rules and technological bases. Then, I realized the basic transistor structures, the CMOS inverter and the elementary logic gates (NAND and NOR). Combining these gates I made a D-type flip-flop. The final counter consisted of four D-type flip-flop and some tens of logic gates. For the interconnection of the gates I had two layers of metal at my disposal. All paths have been routed manually.

The area of the final layout was square, which was one of the project objectives. I drew the pads and then I finished the layout.

For verification I used the device extraction feature of Ledit, generated the netlist according to the circuit drawn (more than 200 transistors) and I imported the generated netlist from Ledit to the PSpice program. Using a 2um technology model from Mosis, I simulated the resulted circuit. The circuit worked correctly.

10. SmartCard reader for room access control using the ATmega32 microcontroller, from Atmel.

For this project I had available a device for reading smartcards, manufactured by Sankyo Company. This device is controlled on the RS232 interface using a set of instructions and a protocol designed by Sankyo, described in its datasheet.

The reader should obtain the unique identifier of the smartcard and send it further to a central door access security console using the Wiegand 26 protocol. For this I had to study the set of Wiegand protocols.

The code written in the ATmega32 microcontroller initializes the Sankyo device, implements control algorithms for the reader and sends the code obtained to the security console by Wiegand interface. In this purpose I used the ATmega32 UART modem and system timers to generate precise pulses necessary for the Wiegand protocol.

11. Electronic control for the integration of a miniPC in a piece of furniture, using the ATmega32 microcontroller, from Atmel.

The mechanical structure of the integration system consists of two motors to be controlled in both directions, one to control the keyboard and the other to control the monitor. Two push-buttons are used to command the system to compact into the furniture or expand out of it. Four Hall sensors detect the keyboard and monitor positions.

I designed a PCB that has an ATmega32 microcontroller in the center. The input ports were connected to the two buttons and the four sensors. Mechanisms have been implemented for hardware debounce, consisting in basic low-pass filters.

To control the two motors I used two DPDT relays. The first relay selects one of two motors and the second relay is used as an H-bridge, reversing the motor direction.

Engine speed is regulated by PWM modulation, achieved with a power NMOS transistor. All commands to the relays and MOS transistor are generated by the code written in the microcontroller. This code changes the duty factor of the motors so that the movement is smooth and detects a situation of a motor block, when the appropriate sensor does not signal correctly in a specified period of time.

The code implemented in the microcontroller is of medium complexity and is mostly based on the ATmega32 interrupt system, timers and counters.

12. Text-mode VGA terminal with keyboard input and monitor output, designed in FPGA

This project emulates an input / output console terminal with keyboard input and it displays the text on a VGA monitor.

To generate the necessary timing signals for the monitor, HS and VS, I have used the timings in the tables from the VESA VGA standards. The screen is divided into 80 columns and 24 lines.

Appropriate to store the ASCII character set, I used a ROM memory in which I wrote the bit matrices exported from a Microsoft DOS font.

The input device used is a PS/2 keyboard. A decoder that interprets the PS/2 protocol and exports the codes for the keys pressed is implemented in Verilog code. Also, the decoder generates control signals for when the user presses the Shift, Ctrl or F1-F6 keys.

Translation codes for PS/2 to ASCII codes were stored in a corresponding ROM table. The translation is implemented also in the Verilog code.

The terminal can display characters typed at the keyboard. The user can change the color to display the text or the background color and can delete characters using the Backspace key. The user can also move the cursor on the screen using the arrow keys.

Appendix 2: Engineer certificate

MINISTERUL EDUCAŢIEI, CERCETĂRII ŞI INOVĂRII
Universitatea POLITEHNICA din Bucureşti
Facultatea de Electronică,Telecomunicaţii şi Tehnologia Informaţiei

Înregistrat ca operator de date cu caracter personal sub nr. 3291

Nr.1127 / 17 07 2009

ADEVERINŢĂ

Prin prezenta se adevereşte că

GHEORGHE T. ANDREI-SORIN

(numele, iniţiala prenumelui tatălui şi prenumele)

născut(ă) la data de 31 decembrie 1985 , în localitatea Bucureşti, judeţul -, fiul (fiica) lui Tudor şi al Iulia a urmat şi a absolvit cursurile de zi în perioada 2005 -2009.

A promovat examenul de licenţă în sesiunea IULIE 2009
(luna şi anul)

cu media 10 (zece) .
(în cifre şi litere)

A obţinut titlul de INGINER în domeniul Inginerie electronică şi telecomunicaţii,specializarea Microelectronică, optoelectronică şi nanotehnologii.

Prezenta adeverinţă se eliberează la cererea titularului şi are valabilitate până la data de 31 ianuarie 2010.

RECTOR,
Prof.dr.ing. Şerban RAICU
L.S.

SECRETAR ŞEF UNIVERSITATE,
Prof.dr.ing. Gabriel IACOBESCU

DECAN,
Prof.dr.ing. Teodor PETRESCU

Secretar şef facultate,
Prof.Elena MLADIN

Appendix 3: Baccalaureate diploma

R O M Â N I A
MINISTERUL EDUCAȚIEI ȘI CERCETĂRII

T.S.

Seria U
Nr. 0005836

DIPLOMĂ DE BACALAUREAT

Domnului GHEORGHE T. ANDREI - SORIN,
absolvent... a l COLEGIULUI NAȚIONAL DE INFORMATICĂ "TUDOR VIANU",
cod SIRUES 2 0 0 2 0 6 4, *localitatea* BUCUREȘTI
, județul SECTOR 1,
filiera TEORETICA, *profilul* REAL,
specializarea MATEMATICA INFORMATICA, *în anul* 2005,
C.N.P. 1 8 5 1 2 3 1 4 4 0 0 1 7, *i se acordă*

DIPLOMA DE BACALAUREAT

în urma rezultatelor obținute la examenul din sesiunea IUNIE-IULIE 2005,
organizat la COLEGIUL NAȚIONAL DE INFORMATICĂ "TUDOR VIANU",
din localitatea BUCUREȘTI, *județul* SECTOR 1

PREȘEDINTELE COMISIEI,
BADEA MARIN

L.S.

SECRETAR,
BERCEA DANIELA

Nr. 54
Data eliberării: anul 2005, luna IULIE, ziua 15 Semnătura titularului

17 - 6 - 1/a