IMPLEMENTATION OF A BASIC MICROCONTROLLER FOR TEACHING EMBEDDED SYSTEMS DESIGN

Maicon Carlos Pereira, Cesar Albenes Zeferino

Universidade do Vale do Itajaí – CTTMar – GSED
Rua Uruguai, 458 – C.P. 360
CEP 88302-202, Itajaí, SC, Brazil
{maicon, zeferino}@univali.br

ABSTRACT

In this paper, it is presented a basic microcontroller, named µBIP, developed to be used in the teaching of embedded systems design. Its architecture was specified by using ArchC, and a synthesizable model was implemented in VHDL and validated in FPGA.

1. INTRODUCTION

The demand for professionals able to deal with the design of digital systems for embedded systems has increased in the last years. Courses covering issues related with this topic lack of basic architectures which could easily be used to teach concepts related to the design of processors and microcontrollers.

In this sense, a basic microcontroller architecture named µBIP (read *micro bip*) was proposed and implemented in order to be used in the teaching of concepts on the design of microcontrollers. A first organization was specified and a soft core was model in VHDL. In the design flow, architectural specification was performed by using ArchC, an Architecture Description Language – ADL developed University of Campinas [1]. ArchC automatically generated the assembler, the linker and a Instruction-Set Simulator. After that, µBIP was specified and a synthesizable soft core was modeled in VHDL and synthesized in FPGA for validation.

This paper presents µBIP architecture and organization and issues related to its implementation.

2. µBIP ARCHITECTURE

In µBIP microcontroller, instructions and data are 16-bit wide. There is only one instruction format, shown in Fig. 1, and three addressing modes. The instruction format has one implicit operand, the ACC (Accumulator) register, and one explicit operand (the operand field). Depending on the addressing mode, this explicit operand can be: a constant in immediate addressing mode); a variable in direct addressing mode; or a vector index in indirect addressing mode. The instruction set includes 29 instructions organized in nine classes (shown in Table 1).

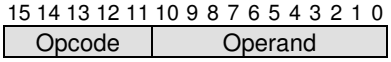

Fig. 1. µBIP instruction format.

Table 1. µBIP ISA.

Class	Instructions
Arithmetic	ADD, ADDI, SUB, SUBI
Logical	AND, OR, XOR, ANDI, ORI, XORI, NOT
Shift	SLL, SRL
Load	LD, LDI
Store	STO
Control	HLT
Branch	BEQ, BNE, BGT, BGE, BLT, BLE, JMP
Vector access	STOV, LDV
Procedures	CALL , RET, RETI

µBIP separated memories for instruction and data, and addressing space is organized in a 2 Kword program space and a 2-Kword data space. I/O is memory mapped.

At the architecture level, µBIP has five registers: PC (Program Counter), ACC (Accumulator), STATUS, SP (Stack Pointer), and INDR (Index Register), used in array-based operations.

For procedure calls and interrupts, µBIP uses a stack to save the current context, which is only the value of PC+1. This approach is based on the one used in Microchip's PIC16 architecture.

As one can see, µBIP architecture was specified to be "as simple as possible". Since it uses a single format for all the instructions of its small instruction set, students can easily learn how to program it. Also, its architecture was specified based in the goal of make easier the design of its organization, as is shown in the following section.

2. µBIP ORGANIZATION

Fig. 2 depicts the µBIP organization. It is based on a Harvard mono-cycle approach like the one used in the monocycle version of MIPS [2]. It includes the instruction and data memories, the CPU (composed by the Control Unit and the Datapath) and the peripherals.

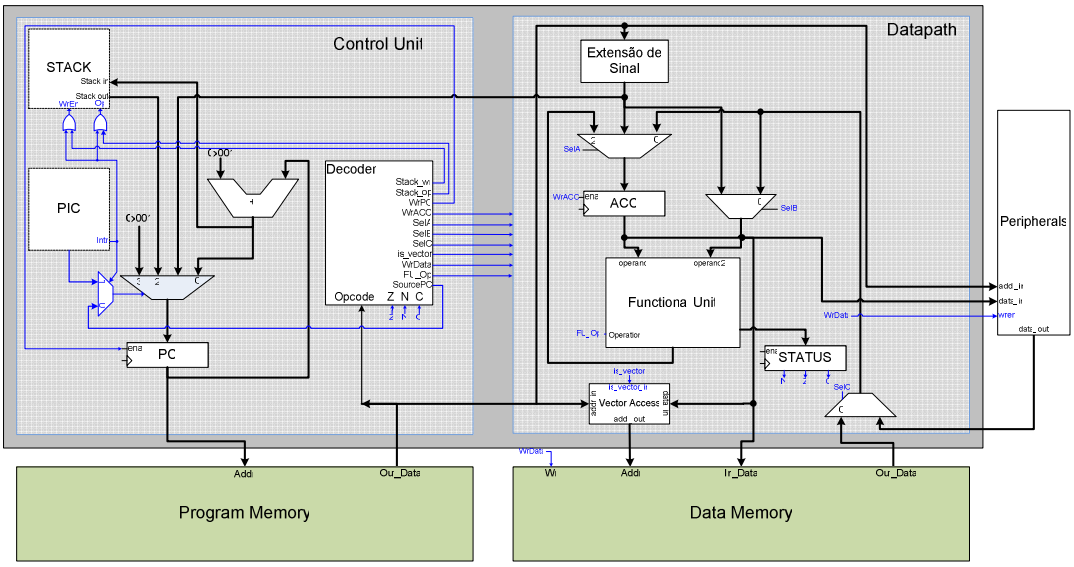

Fig. 2. µBIP organization.

The Control Unit fetches a instruction from the Program Memory, decodes it and sends command signals to the Datapath, which is responsible to process data. The first version of µBIP includes the following peripheral and hardware features: (a) two 16-bit I/O ports with individual direction control for each pin; (b) a 16-bit timer; (c) an interrupt controller; and (d) hardware support for procedure calls.

3. µBIP DESIGN

In design of µBIP, ArchC was used in the architectural specification phase. The tools generated automatically by ArchC were used in the validation of the proposed architecture. All the instructions were validated by using unitary tests and 79% of the ISA was also validated by simulating the execution of applications based on the benchmarks of Dalton Project [3].

µBIP organization was described in VHDL and synthesized in Altera FPGA EPF10K70RC240-4. Silicon costs depends on the target application.

Table 2 summarizes the ISA coverage of some of the Dalton Project applications, their costs and maximum operation frequency.

Table 2. Test coverage and silicon results of Dalton Project applications.

Application	Program size (instructions)	ISA test coverage	#Logic Cells	*fclk* (MHz)
cast	9	20%	492	7.44
fib	50	48%	992	6.95
gcd	22	34%	805	6.75
sort	97	44%	1050	6.58
sqroot	59	44%	996	6.51

Fig. 3 illustrates a simulation of execution of *cast* application, which takes a 16-bit word and separates it into two 8-bit words. Firstly, the two I/O ports are configured as output ports (1 and 2). After, a 16 bit word, 0x1234, is built in ACC register (3). In 4, the less significant byte (0x34) is written into *port0_data*. ACC register is shifted right (5) and the most significant byte is written into *port1_data* (6).

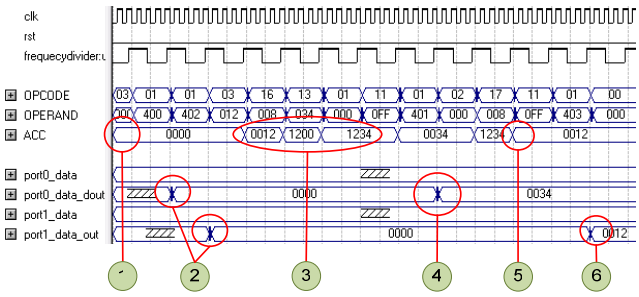

Fig. 3. Simulation of *cast* application.

Validation was also performed by running application on Altera UP-2 developing board.

4. CONCLUSIONS

µBIP was developed as a final work of a Computer Science student. Its simplicity and the use of CAD tools make ease its implementation in about 5 months. Futures works on µBIP includes the development of a compiler and of a visual IDE.

5. REFERENCES

[1] The ArchC Team. The ArchC architecture description language v2.0. Campinas: The ArchC Team, 2007.

[2] Patterson, D. A., Hennessy, J. L. Computer Organization and Design. San Francisco: Morgan Kaufmann, 1998.

[3] The Dalton Project Team. The UCR Dalton Project. University of California – Riverside, 2001.