Design And Implementation Of Sharc Processor

T. Chandrasekhar
Associate Professor
Ciet Engineering College,
Rajahmundry (Ap).

J. Suneel Chakravarthi
Assistant Professor
Ciet Engineering College,
Rajahmundry (Ap).

ABSTRACT

This paper deals with the design and implementation of the 32-bit floating point Digital signal processor with MIPS (microcomputer with out interlocked pipeline stages).The designed DSP has 32 floating point MIPS instructions, instruction sets suitable for processing digital signals and consists of super Harvard architecture, 40-bit ALU, 5 level pipelines, 17-bit X 17-bit parallel multiplier for single-cycle MAC operation, 8 addressing modes, 8 auxiliary registers, 2 auxiliary register arithmetic units, two 40-bit accumulators and 2 address generators. The VHSIC HDL coded synthesizable RTL code of the DSP core has a complexity of 80,670 in the two input NAND gates. We verified the functions of the DSP by a simulation with a single instruction test as the first step and then implemented the DSP with the FPGA. The test vectors have a single instruction test, combination of single instructions and algorithm applications, ADPCM vocoder and the MP3 decoder. After FPGA verification, the DSP core carried out three test vector sets which are tested at FPGA at the 106 MHz clock rates.

General Terms

Digital signal processor that can execute millions of instructions per second

Keywords: VLSI, DSP, MIPS, FPGA

1. INTRODUCTION

As per the growing technology, the size of processor is smaller and so many portable devices are manufactured. Digital Signal Processing is distinguished from other areas in computer science by the unique type of data it uses signals. In most cases, these signals originate as sensory data from the real world: seismic vibrations, visual images, sound waves, etc. DSP is the mathematics, the algorithms, and the techniques used to manipulate these signals after they have been converted into a digital form. This includes a wide variety of goals, such as: enhancement of visual images, recognition and generation of speech, compression of data for storage and transmission, etc. Therefore the importance of the DSP (digital signal processor) which can process fast and accurate digital signals for audio/image processing or data communication has been getting bigger. This paper includes information of the design and implementation of the DSP core on FPGA. The suggesting DSP has advanced Harvard architecture (SHARC) and instruction sets suitable for processing digital signals and MIPS instruction sets. Comparing with fixed point, floating point dsp provides high range, precision and accuracy. But it suffers from the problem of speed reduction for DSP computations. In order to compensate this speed reduction problem, MIPS has introduced in the DSP, in this paper.

2. DESIGN

The suggesting DSP has a couple of features of architecture because DSP has to carry out many digital signal algorithms at a real time. First, DSP has fast and optimized multiplier for operating the MAC instruction during one cycle. And it has the advanced Harvard architecture called super Harvard architecture to improve an operating speed. For a movement of bust bits used frequently in digital signal algorithms. The overall architecture of the suggesting DSP consists of data and address buses, a central processing unit, a control unit and memory interface unit[3], shown in Figure 1.

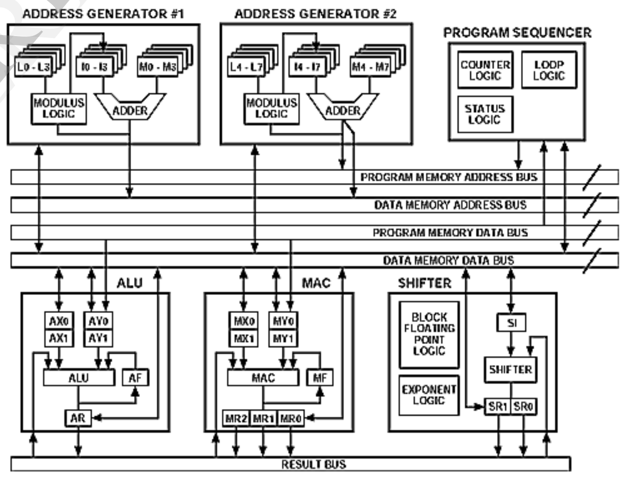


Figure 1. Typical DSP architecture

Figure 1, presents a more detailed view of the DSP architecture, showing the I/O controller connected to data memory. This is how the signals enter and exit the system. For instance, the DSP's provides both serial and parallel communications ports. These are extremely high speed connections. For example, at a 40 MHz clock speed, there are two serial ports that operate at 40 Mbits/second each, while six parallel ports each provide a 40 Mbytes/second data transfer. When all six parallel ports are used together, the data transfer rate is an incredible 240 Mbytes/second.

2.1 Super Harvard architecture

Figure 2 illustrates the sophisticated, the Super Harvard Architecture. DSPs, a contraction of the longer term, Super Harvard ARChitecture (SHARC). The idea is to build upon the Harvard architecture by adding features to improve the throughput. While the SHARC DSPs are optimized in dozens of ways, two areas are important enough to be included in Fig. 2, an instruction cache, and an I/O controller.

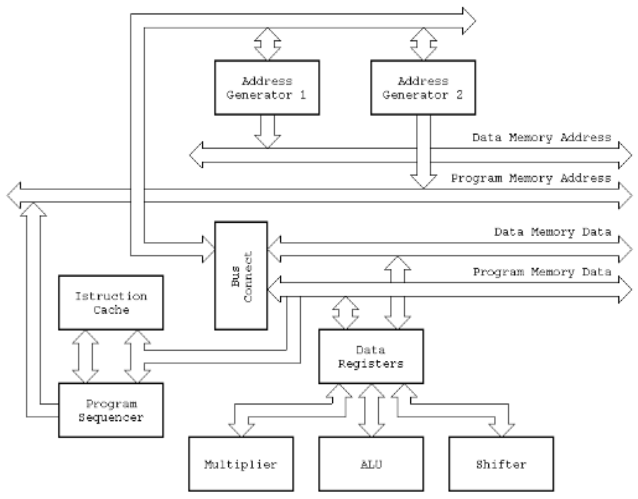


Figure 2. Super Harvard architecture

A handicap of the basic Harvard design is that the data memory bus is busier than the program memory bus. When two numbers are multiplied, two binary values (the numbers) must be passed over the data memory bus, while only one binary value (the program instruction) is passed over the program memory bus. To improve upon this situation, we start by relocating part of the "data" to program memory. For instance, we might place the filter coefficients in program memory, while keeping the input signal in data memory. (This relocated data is called "secondary data" in the illustration). At first glance, this doesn't seem to help the situation; now we must transfer one value over the data memory bus (the input signal sample), but two values over the program memory bus (the program instruction and the coefficient). In fact, if we were executing random instructions, this situation would be no better at all.

However, DSP algorithms generally spend most of their execution time in loops. This means that the same set of program instructions will continually pass from program memory to the CPU. The Super Harvard architecture takes advantage of this situation by including an instruction cache in the CPU. This is a small memory that contains about 32 of the most recent program instructions. The first time through a loop, the program instructions must be passed over the program memory bus. This results in slower operation because of the conflict with the coefficients that must also be fetched along this path. However, on additional executions of the loop, the program instructions can be pulled from the instruction cache. This means that all of the memory to CPU information transfers can be accomplished in a single cycle: the sample from the input signal comes over the data memory bus, the coefficient comes over the program memory bus, and the program instruction comes from the instruction cache. In the jargon of the field, this efficient transfer of data is called a high memory access band width.

2.2 Architecture of DSP

The suggested 32-bit floating point DSP has a bus architecture which is divided into two types, a program and data. The PAGEN (Program Address Generator) and DAGEN (Data Address Generator) generates each address, data and program. Through a memory interface unit such as MMU (Memory Management Unit) and EXTMMU, the DSP controls an internal or an external memory unit. The CPU such as ALU or MAU (Memory Address Unit) carries out the data operation.

2.2.1 CPU

The CPU (Central Processing Unit) carries out data operations received from the buses. It consists of specific operating blocks, such as the ALU (Arithmetic Logic Unit), MAU (Multiply and Adder Unit), CSSU (Compare Select and Store Unit), Barrel shifter and Exponent encoder[2], shown in Fig.3. Most of arithmetic and logical instructions are processed at this block during one or two cycle.

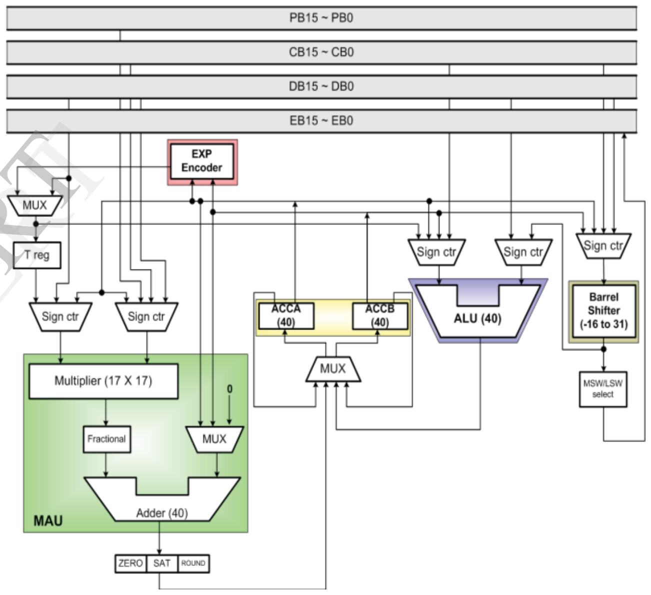


Figure 3. CPU architecture

2.2.1.1 ALU

The ALU consists of the 40-bit adder and logical operation units. As shown figure 3, operated output data of ALU are inputted to the barrel shifter, then after being shifted they are stored to a memory or an accumulator. When operated data are stored after being shifted, they have to be carried out within one cycle. And using an EXP encoder block CPU can express and operate a floating-point data.

2.2.1.2 Accumulator

The proposed DSP has two 40-bit accumulators (ACCA and ACCB). It stores data which are executed at ALU and MAU and can sends operated data to ALU again. Output data of ACCA can be an input data of the MAU. Also it is used to operate comparing-instructions MIN and MAX or parallel-instructions like LD||MAC which load data to the accumulator and operate them. 40-bit ACC data consist of three parts: 8-bit guard data, 16-bit high data and 16-bit low data.

2.2.1.3 MAU

The digital signal processing algorithms, such as FFT, FIR and Huffman decoding, have repetitive and complicated operations, so a performance of the DSP is up to the execution of these algorithms [2]. The proposed DSP has an optimized multiplier based on the modified radix-4 booth algorithm shown figure 4. And it carries out an operation like equation (1) at one cycle. The MAU operates MPY, MAC, and SQURA and so on

$$Y = \pm(ai \times bi(frct)) + ci \quad .(1)$$

2.2.1.4 Barrel Shifter

Under signal processing algorithm, when move or operate data, DSP has to shift one more bits of data. For reducing an operating time DSP has a barrel shifter in CPU. The range of data shifting is -16 to 31 bit. And it can carry out an arithmetic shift and a logical shift by a control signal.

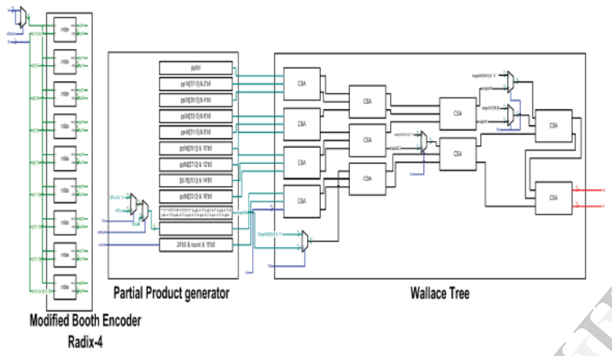
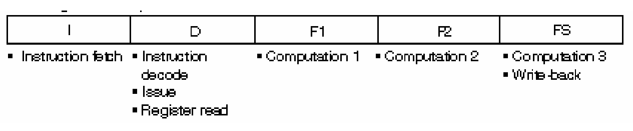


Figure 4. MAU architecture

2.2.2 Pipeline and Bus

In computing, a pipeline is a set of data processing elements connected in series, so that the output of one element is the input of the next one. The elements of a pipeline are often executed in parallel or in time-sliced fashion; in that case, some amount of buffer storage is often inserted between elements. As the assembly line example shows, pipelining doesn't decrease the time for a single datum to be processed; it only increases the throughput of the system when processing a stream of data.For data processing, the DSP has the Advanced Harvard architecture which reduces the waiting time of the buses. So it can access memory units through one program bus (PB), three data buses (CB, DB, EB) and each address bus (PAB, CAB, DAB, EAB). The program bus reads an opcode and an operand from program memory. And three data buses are divided into 2 type. Data reading bus (CB, DB) reads an operand from data memory, data writing bus (EB) writes data in data memory. Also it has a five-level pipeline: I (Instruction Fetch), D(Instruction decode, Issue, Register read), F1 (Computation 1), F2 (Computation 2), FS (Computation 3, Write back). When the instructions are operated, the DSP permits an instruction overlap and each pipeline level performs an independent function. The function of each cycle is explained on Figure 5.



I	D	F1	F2	FS
▪ Instruction fetch	▪ Instruction decode ▪ Issue ▪ Register read	▪ Computation 1	▪ Computation 2	▪ Computation 3 ▪ Write-back

Figure 5. Floating point Pipeline

2.2.3 Memory

The suggested DSP has 2k words x 32-bit internal ROM and 10k words x 32-bit internal RAM for accessing two data within one cycle simultaneously. And it has each 64k words x 32-bit memory space of the Data, Program and I/O.

2.2.4 Addressing mode

Two ARAU (Auxiliary Register Arithmetic Unit) generate two t at one cycle. ARAU works parallel with ALU and has 8 addressing mode : Short immediate addressing mode, Long immediate addressing mode, Absolute addressing mode, Direct addressing mode, Indirect addressing mode, Bit-reversed index addressing mode, Memory-mapped register addressing mode and Stack addressing mode.

2.2.5 MIPS (pipelined) Architecture

Figure 6 shows the Pipelined MIPS Architecture with 5 level pipeline stages. They are Instruction fetch (IF), Instruction decode register fetch (ID), Execute address calculation (EX), Memory access (MEM), Write back (WB).

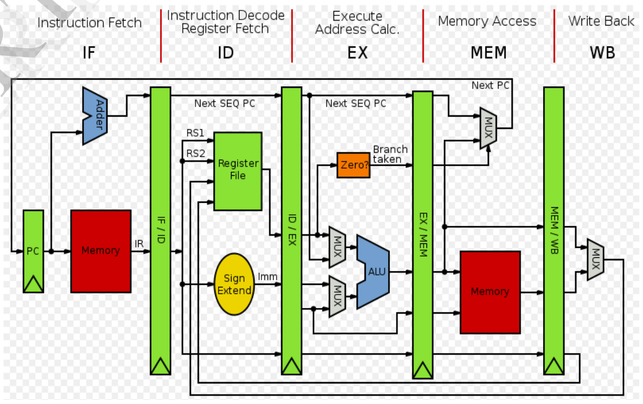


Figure 6. Pipelined MIPS Architecture

2.2.5.1 MIPS Instruction set

Full name of MIPS is microcomputer without interlocked pipeline stages. Another informal full name is Millions of instructions per second. MIPS have already been pronoun of MIPS instruction set and MIPS instruction set architecture. ISA (Instruction Set Architecture) of processor is composed of instruction set and corresponding registers. Program based on same ISA can run on the same instruction set. MIPS instruction has been developed from 32-bit MIPSI to 64-bit MIPSIII and MIPSIV since it was created. To assure downward compatibility, every generation production of MIPS instruction directly extends new instruction based on old instruction but not abnegates any old instruction, so MIPS processor of 64-bit instruction set can execute 32-bit instruction. The DSP has 32 MIPS floating point instructions, instruction sets suitable for processing digital signals and size of instruction is 32 bit. They are classified into five types shown in table 1 : Arithmetic operations, Logical operations and data transfer operations, conditional and unconditional operations.

Table 1. MIPS floating point instructions

Category	Name	Instruction syntax
Arithmetic	FP add single	add.s $x,$y,$z
	FP subtract single	sub.s $x,$y,$z
	FP multiply single	mul.s $x,$y,$z
	FP divide single	div.s $x,$y,$z
	FP add double	add.d $x,$y,$z
	FP subtract double	sub.d $x,$y,$z
	FP multiply double	mul.d $x,$y,$z
	FP divide double	div.d $x,$y,$z
Data Transfer	Load word coprocessor	lwcZ $x,CONST ($y)
	Store word coprocessor	swcZ $x,CONST ($y)
Logical	FP compare single (eq,ne,lt,le,gt,ge)	c.lt.s $f2,$f4
	FP compare double (eq,ne,lt,le,gt,ge)	c.lt.d $f2,$f4
Branch	branch on FP true	bc1t 100
	branch on FP false	bc1f 100

2.2.5.2. MIPS Instruction formats

All MIPS instructions are all 32-bit specified instruction and instruction address is word justification. MIPS divides instructions into three formats: immediate format(IFormat) ,register format(R-Format) and jump format(JFormat)[2]. Three instruction format shows in table 2.

Table 2. MIPS instruction formats

Type	-31-	format (bits)				-0-
R	opcode (6)	rs (5)	rt (5)	rd (5)	shamt (5)	funct (6)
I	opcode (6)	rs (5)	rt (5)	immediate (16)		
J	opcode (6)	address (26)				

Meaning of every instruction field as following:

_ *OP*: 6-bit operation code;

_ *rs:* 5-bit source register;

_ *rt:* 5-bit temporary (source/destination) register number or

Branch condition;

_ *Immediate*: 16-bit immediate, branch instruction offset or

address offset;

_ *Destination*: 26-bit destination address of unconditional jump;

_ *rd*: 5-bit destination registers number;

_ *shamt*: 5-bit shift offset;

_ *funct*: 6-bit function field

3. SIMULATION

To verify the designed DSP core, we goes through three processes: HDL code simulation, FPGA implementation.

3.1 Functional Simulation

When the 211 instruction sets are verified using a logic simulation tool, verifications of a state of internal registers and buses, and status flags are companied. Figure 7. Shows the result of an functional simulation, ADD instruction. After tests of each single instruction are finished, many combinations of single instructions are verified and they work correctly.

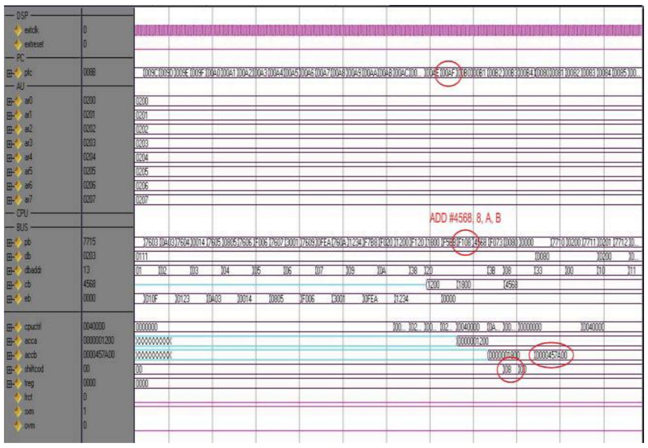


Figure 7. Functional simulation of ADD #4568, 8, $A, $ B

3.2 Application algorithm simulation

The G.726 ADPCM (Adaptive Difference Pulse Code Modulation) suitable for suggesting DSP core is coded and verified by comparing results of a C-code and results of a HDL. 174,950 for the input data with a sampling rate of 8 kHz encoded with ADPCM usage total of 60 instructions and 1,700 words program memory space.

3.3 MP3 decode

For more complicated verification, many cases of the instruction combinations in the MP3 (MPEG-I layer 3) decode algorithms are carried out. Like the ADPCM algorithm test, this algorithm is verified by comparing results of a C-code and results of a HDL shown in figure 8. Input data are from a sampling rate of 44.1 kHz stereo. These algorithms are executed at an average of 60 MIPS and use 12k words program memory space and 27k words data memory space. Under test of the MP3 decode algorithm, we confirm that a weight of instructions of addition and multiplication is large. So high performance DSP must have an effective and optimized an adder and a multiplier. Table 3. shows calculation weight of each routines of MP3 decode.

TABLE 3. Weight of the MP3 decoder routine

Name	MIPS	%
III_hufman_decode (Huffman decoder)	6.5	10.1
III_hybrid (IMDCT)	16.3	25.5
SubbandSynthesis	16.9	26.4
III_dequantize_sample (Dequantization)	9.4	14.7
File read/write	2.5	4
Total	64.0	100

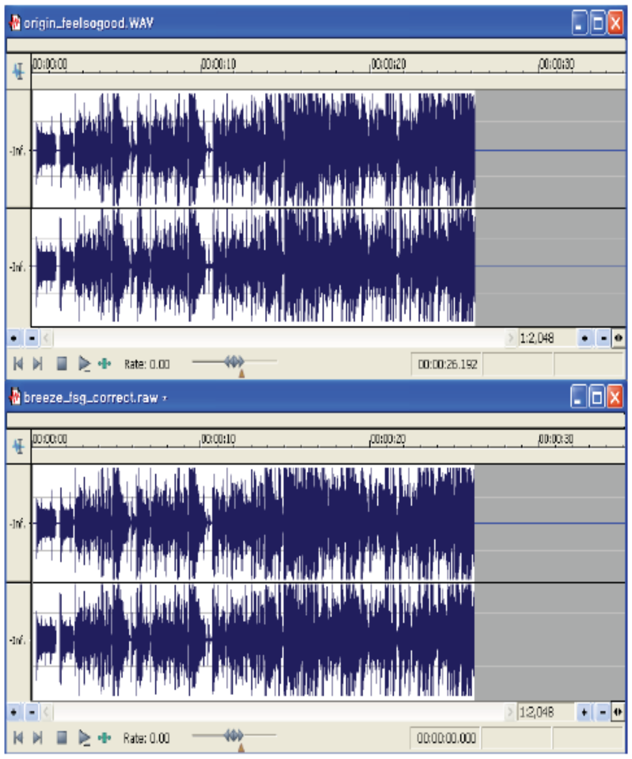


Figure 8. Results of MP3 decode algorithm of C-code HDL

3.4 Hardware Verification

HDL codes of DSP core are synthesised and downloaded to the Spartan 3E FPGA. Under the FPGA implementation, four types of data movement between an internal and an external, tests of single instructions and tests of combinations of single instructions are verified. DSP core on FPGA is carried out this test.

4. CONCLUSON

This paper describes the design and implementation of the 32-bit floating point digital signal processor with out interlocked pipeline stages. The processor core is designed using the VHDL and verified instruction sets and application algorithms, such as the ADPCM and MP3 decode, through a functional simulation. After the verification, FPGA of DSP core is implemented. The implemented DSP core on FPGA has 80,670 gates based on a two input NAND gate and can operate at the maximum 106MHz clock rates. And the designed DSP core will be applied to the SoC technology.

4. REFERENCES

[1] Donghoon Lee, Kyunsoo Kwon, Wontae Choi, IEEE 2008, a study on 16 bit DSP. Pusan national University.

[2] CS.Wallace, “A suggestion for fast multipliers”, IEEE Trans, On Electronic computers, Vol EC, no.3 pp .14-17,Feb

[3] Kui YI, Yue Hia DING, A study on MIPS, IEEE 2009, Wuhan Polytechnic University.

[4] Avatar Singh and S.Srinivasan, DSP Implementation : Using DSP Microprocessor with Examples from TMS320C54x, Thomson, Brooks, Cole, 2004

[5] http://en.wikipedia.org/wiki/MIPS_architecture