University LSI Design Contest

Title	:	**D-mark: A Tiny DSP Processor**
Authors	:	Satya Kiran.M.N.V, Jayram.M.N, Muralimohan.K Vinay Kumar.A, Sudhanshu Jayaswal, Jayadeva, B.Bhaumik
Speaker	:	Satya Kiran.M.N.V
Contact	:	Dr.Jayadeva, Dept. of Electrical Engineering, Indian Institute of Technology, Delhi, Haus Khas, New Delhi -110016, INDIA
Phone No	:	(011) 659 6103
Fax	:	(011) 658 1264
Email	:	jayadeva@ee.iitd.ernet.in
Area	:	2a (Full Custom / Cell based LSI) Digital Signal Processing

Brief Description:

D-mark is a RISC processor for simple DSP applications like digital filtering & FFT etc. The design fits in a 2.2mm x 2.2mm frame, using 1.5u CMOS technology. We followed a semi-custom approach with a view to minimize area and optimize instruction set. Various applications were profiled and analyzed to obtain the architecture and the instruction set. The processor was first modelled in VHDL, analyzed and simulated for various architectural changes before embarking on the physical design. All modules were custom designed except the control unit. Various modules were placed and routed manually with less then 5 % free area in the chip.

The main features of the design:

- Efficient RISC architecture with fixed execution cycles and fixed instruction length
- Compact bit reverse address generator for FFT
- Compact ALU with 12 different operations
- Signed hardware multiplier
- Move instruction with auto decrement/increment
- Register addressed JUMP instruction for reduced overhead

D-mark: A Tiny DSP Processor

A processor for efficient implementation of Digital filter and FFT

Satya Kiran.M.N.V, Jayram.M.N, K.MuraliMohan,
VinayKumar.A, Sudhanshu Jayaswal, Jayadeva, B.Bhaumik
Dept. of Electrical Engineering,
Indian Institute of Technology, Delhi
Haus Khas, New Delhi-110016, INDIA
Email: jayadeva@ee.iitd.ernet.in

Abstract - The design was a challenge to explore the feasibility of realizing a Digital Signal Processor in a 2mm X 2mm frame, while still having sufficient functionality to be able to run simple DSP applications such as FFT and filtering. The very small area available makes the design task extremely difficult. Several iterations of the instruction set, layout, placements and routing were needed before the goal could be achieved. The chip was designed for a 1.5u CMOS process. The entire processor has been simulated both at the behavioural level and at transistor level.

I. Introduction

It would be very difficult to implement digital processing application using conventional general-purpose processor. In order to gain insight into the design of DSP processor we initially implemented FFT, FIR and IIR algorithms on Intel 8085 processor. After analyzing the assembly code, we realized that by modifying the architecture by including special functional units and instructions, it is possible to improve the efficiency of the code for these applications. Additionally by cutting down unwanted hardware modules / instructions we will be able to reduce power dissipation significantly.

The important features needed for implementing the chosen applications include,

1) A hardware multiplier capable of implementing a signed multiplication
2) A large number of general purpose registers
3) Incorporating register indirect addressing with autoincrement and autodecrement facility.
4) Special address generators for FFT
5) Multiple accumulators or programmable destination register architecture rather than single accumulator architecture.

After profiling the applications, we came out with an optimum architecture and instruction set. We described the entire architecture initially using VHDL. After verifying the functionality, we synthesized the control unit using the Synopsys DC compiler. All the remaining modules were full custom designed for minimum area realization.

The basic system architecture is described in Section II, followed by the instruction set, features and simulation results in section III.

II. System Architecture

The processor organization can be broadly classified into datapath, control unit and register file. Fig. 1 shows the basic block diagram of the entire processor.

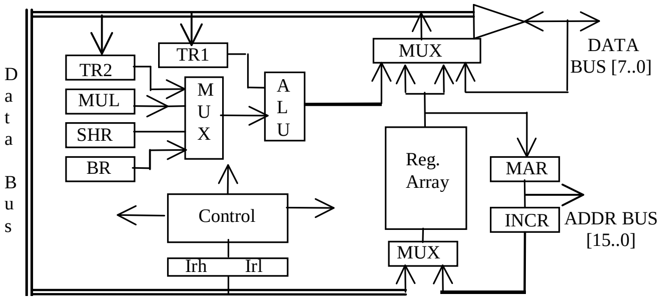

Fig.1 System Architecture

The control unit is a synthesized, positive edge sensitive FSM. The register file consists of eight rows of 16 bit registers. The Program Counter has also been included as a part of the register array and is denoted as R0. This register is not available for general-purpose programming. Among the remaining registers, four can be used for register indirect addressing. The register array loads the data during the high level of the clock. During read operation the entire row of 16 bits is read and passed to the MAR cum Incrementer (which forms the addressing unit) and MUX.

A. Datapath

The Datapath of D-mark has been efficiently designed with a bit sliced ALU architecture. Single bit ALU architecture is shown in Fig. 2. This unit is capable of implementing 12 different operation. Another unique aspect of the datapath is the ability to realize new types of instructions depending upon the application. For example, it is possible to multiply two numbers and simultaneously AND the result with another operand. The datapath also has a single bit combinational shifter capable of executing logical/arithmetic right/left shift. Also it has a signed Booth multiplier. The result from BR, SR, MUL (multiplier) or TR2 is selected by the MUX as one of the operands for the ALU.

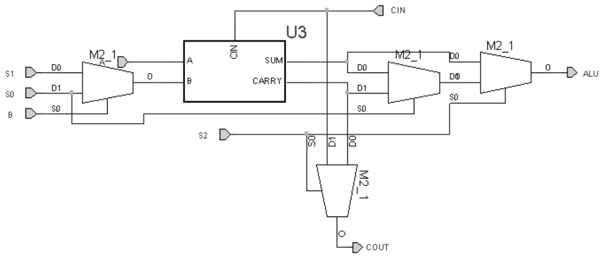

Fig.2 Bit Slice of ALU

B. *Bit Reversal Addressing Unit(BR)*

When using radix-2 computation for FFT implementation, we need to access samples from memory in a prescribed sequence at each stage. For example in an 8-point FFT one needs to access samples 0,1,2,3,4,5,6,7 in the first stage, 0,2,1,3,4,6,5,7 in the second stage and 0,4,2,6,1,5,3,7 in the third stage. We have identified that the following new method can generate these sequences. We add a control pattern to the base address were the samples are stored at each stage of FFT to get the new base address. For instance in the case of the third stage of FFT computation, one operand for BR unit is the base address, were the samples are stored and the second operand is the pattern 0000100. Fig. 3 shows the propagation of the carry in this scheme. Successively adding the control pattern to the new base address generates the required sequence viz. 0,4,2,6,1,5,3,7.

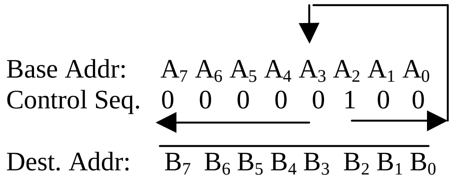
Base Addr: A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0
Control Seq. 0 0 0 0 0 1 0 0
Dest. Addr: B_7 B_6 B_5 B_4 B_3 B_2 B_1 B_0

Fig. 3 Propagation of the carry in BR unit

The hardware for this operation is referred as BIT-REVERSER. An instruction BR has been provided for invoking this addressing scheme for efficient FFT computation.

III. Implementation and Results

A. *Special Instructions*

All the instructions take 7 cycles for execution and use a 16-bit instruction code. The memory fetch is performed during the first 3 cycles. Execution and write back during the remaining cycles. The instructions which were added for efficient implementation of DSP algorithms are Bit Reversal (BR) addition, Move with auto increment/decrement, Jump instruction to the destination address provided by a specified register pair (useful in implementing Loops) and the multiplication instruction. Fig. 4 shows the layout of the processor.

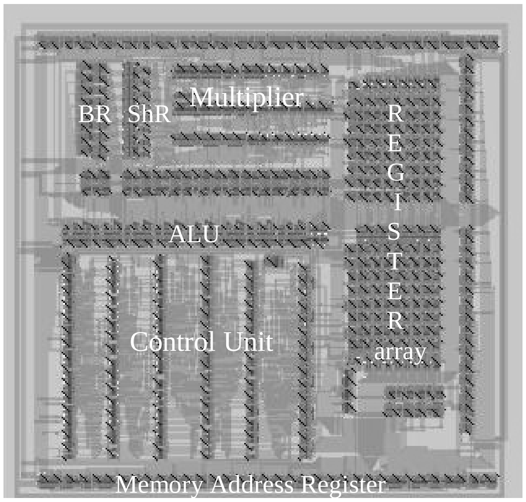

Fig. 4 The Complete Layout of D-mark without I/O Pad's

B. *Simulation Results*

The VHDL description of the entire processor with external memory model was simulated initially with benchmark programs like sorting array, IIR, FIR, and Convolution. The entire chip layout was drawn using Tanner Tools and finally individual modules were simulated using T-spice and IRSIM. Special pins were provided for observing the status of D-mark for testing purposes. The D-mark code was found to be compact for most DSP applications. The specifications are shown in Table I.

TABLE I: D-mark Chip Specification

Transistor count	5.6K
Die Size (mm^2)	2.2 x 2.2
Process Technology	1.5u CMOS
Supply Voltage	5 V
Design Frequency (MHz)	10
No. of Instruction	27
No. of I/O pins	36/40

IV. Conclusions

The entire processor was tested and verified and is expected to be fabricated shortly. The design has been optimized for minimum area and has a flexible instruction set. Overall we were able to implement a tiny DSP processor within the given area and capable of implementing FIR and FFT algorithms which were our benchmarks.

Acknowledgements

The authors acknowledge the active participation of VDTT and IEC students of EE Dept, IIT Delhi for their discussions during the course of this project.