Implementation of MAC Units on FPGAs for DSP Architecture

Shavan K. Asker

ECE Dept., College of Eng., University of Dohuk

Abstract

This paper is an attempt to design and implement MAC (multiply-accumulate) units for pipeline DSP architectures on FPGAs. An application has been chosen to evaluate the results of the architecture. Results show that these units are applicable and can be used by the developers especially by the lifting based discrete wavelet transform.

Keywords: MAC unit, FPGAs, Architecture.

تنفيذ وحدات الضرب التراكمية MAC على FPGA

لمعمارية معالجة الاشارة الرقمية

شبفان كمال عسكر

كلية الهندسة – جامعة دهوك

الخلاصة

في هذا البحث تم تصميم وتنفيذ وحدة الضرب والجمع لمعمارية معالج الاشارات الرقمية وبتطبيق مبدأ خط الانتاج (pipelining).

تم اختيار تطبيق لتقييم نتائج هذا التصميم. اظهرت النتائج بأن هذه الوحدة قابلة للتطبيق وممكن استخدامها من قبل الباحثين خاصة في مجال تطبيقات التحويلات المويجية المعتمد على اسلوب الرفع.

1. Introduction

Received 1 June 2006 Accepted 20 March 2007

The number and variety of products that include some form of digital signal processing has grown dramatically over the last years. DSP has become a key component in many consumers, communications, medical, and industrial products. These products use a variety of hardware approaches to implement DSP, ranging from the use of off-the-shelf microprocessors to field-programmable gate arrays (FPGAs) to custom integrated circuits (ICs). Programmable "DSP processors," a class of microprocessors optimized for DSP, are a popular solution for several reasons. In comparison to fixed-function solutions, they have the advantage of potentially being reprogrammed in the field, allowing product upgrades or fixes. They are often more cost-effective (and less risky) than custom hardware, particularly for low-volume applications, where the development cost of custom ICs may be prohibitive. And in comparison to other types of microprocessors, DSP processors often have an advantage in terms of speed, cost, and energy efficiency [1].

Digital signal processing applications require high computational performance due to their real-time characteristics. There are many implementation media available for signal processing. These implementations vary in terms of programmability from fixed-functionality hardware like ASICS to fully programmable like general-purpose processors. The emergence of the new architecture, which offers the same computational attributes as fixed-functionality architectures in a package that can be customized in the field, is driven by a need for real-

time performance within the given operational parameters of a target system and a need to adapt to changing data sets, computing conditions, and execution environments of DSP applications. In this paper we used three main ideas VHDL, architecture pipelining, and implementation of FPGAs. More details on FPGAs can be found in [2].

Very high speed integrated circuit Hardware Description Language (VHDL) can be used to model a digital system at many levels of abstraction, ranging from algorithmic level to the gate level. The complexity of the digital system being modeled could vary from that of a simple gate to a complete digital electronic system. The digital system can also be described hierarchically, timing can also be explicitly modeled in the same description. The VHDL language can be regarded as an integrated combination of the following languages; Sequential language, Concurrent language, Netlist language, Timing specifications, and Waveform generation language [3,4].

A field programmable gate array (FPGA) is a programmable logic device that supports implementation of relatively large logic circuits. FPGAs provide logic blocks for implementation of the required function, they can be used in all applications that use small-scale integration (SSI), medium scale integration (MSI) and PLDs. They also replace mask-programmable gate arrays in many applications that are limited to 10.000 gates and that do not require very high operational speed [5,6,7]. FPGA is a Very Large Scale Integration (VLSI) module and is considered as an extension of mask programmable gate array (MPGA). FPGA technology was derived from MPGA technology but MPGAs provide greater gate densities and clock speed than FPGAs and are non-volatile [8]. This design has been simulated using the modelsim software then implemented on FPGA kit of the model XC4003E.

2. A DSP-Type Architecture for Lifting

A filter independent DSP-type parallel architecture has been proposed by Martina, Masera, Piccini, and Zamboni [9]. The architecture consists of $N_t = \max_i \{k_{si}, k_{ti}\}$ number of MAC (multiply-accumulate) units, where k_{s_i} and k_{t_i} are length of the primal and dual lifting filters s_i and t_i respectively in step i of lifting factorization. The architecture is shown in Figure 1.

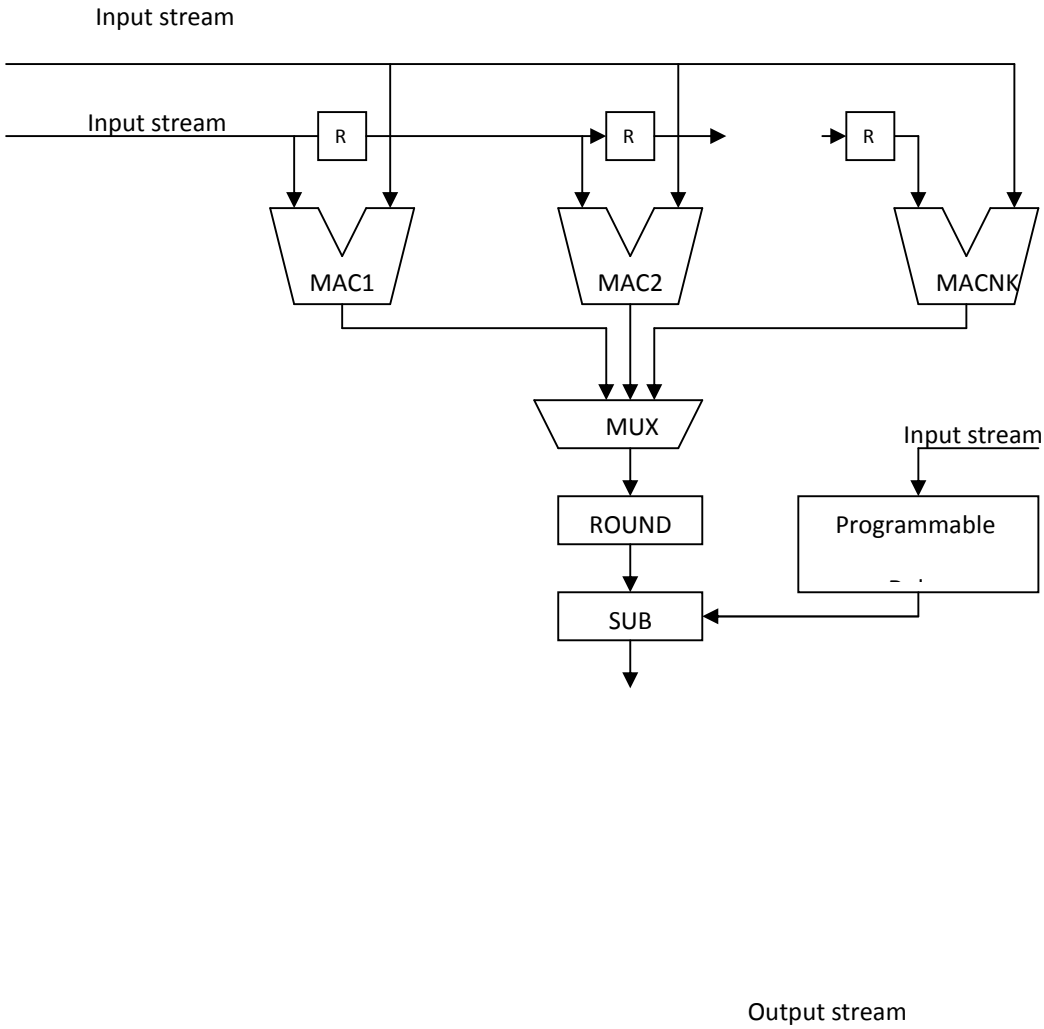


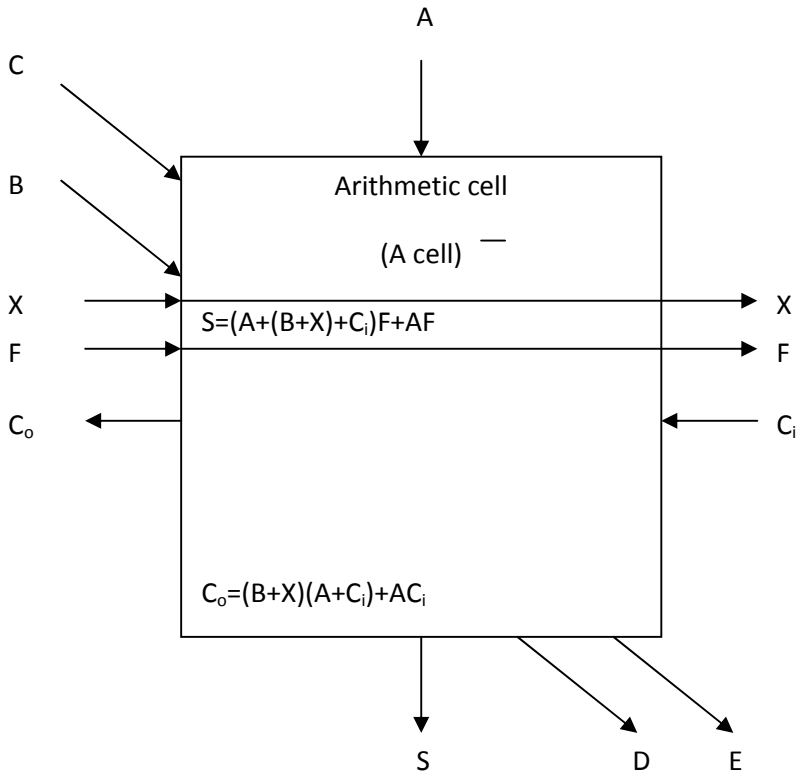
Figure 1: Parallel MAC architecture for lifting.

The above architecture is designed to compute n_t simultaneous partial convolution products selected by the multiplexer (MUX), where n_t is the length of filter tap for the lifting step being currently executed in the architecture. After n_t clock cycles, the first filtered sample is available for rounding operation at the output of the first $MAC1$ and subsequent

samples are obtained in consecutive clock cycles from the subsequent MAC units ($MAC2,....,MAC_{n_t}$). The "programmable delay" is a buffer that guarantees the subtraction consistency to execute corresponding $a_{out}[j]$ and $d_{out}[j]$ samples at the output. The Round unit in Figure (1) computes the floor function and the SUB unit processes the corresponding subtraction operations. The input sample stream (a two dimensional image) are stored into a RAM in four sub-sampled blocks in order to properly address the row-wise and column-wise processing of the image for 2-D lifting DWT implementation. A detailed memory addressing scheme and their access patterns have been discussed in great detail in [9,10].

3. MAC Unit Design and Implementation

The design is a multifunction pipeline that can perform different functions at different times upon program control or firmware control. We present a four-function pipeline proposed by Kamal, this pipeline can perform multiply, divide, squaring, and square root operations. Two types of cells building are used in this four-function pipeline construction. The two cell types are specifies in figures (2) and (3) by Boolean equations. The A cell is a controlled 1-bit adder subtractor with bypass signal lines. The K cells are for function selection and boundary carry control.



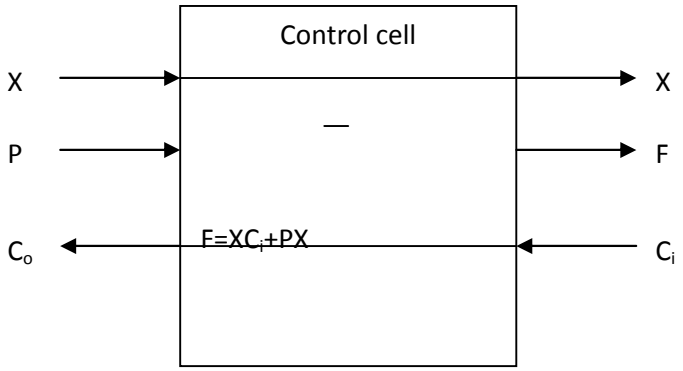
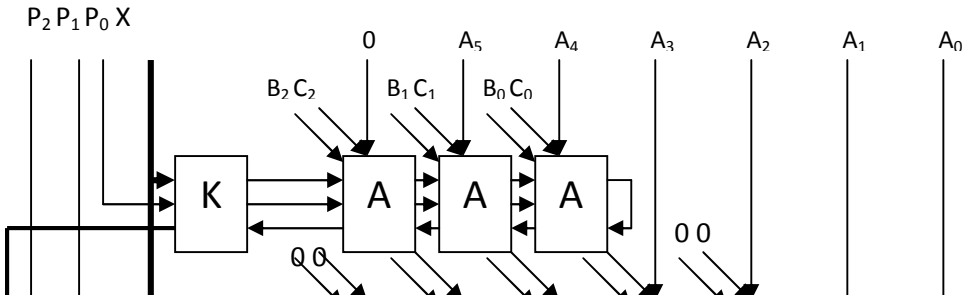


Figure 3. The K cell which is used in the construction of the four-function arithmetic pipeline.

The schematic design of four-function pipeline with A cells and K cells, which is implemented using VHDL is shown in figure (4). Arithmetic computations are specified as the following relationships:



Multiply operation

$(B_2B_1B_0)X(P_2P_1)=(S_6S_5S_4S_3S_2)$

With these conditions: X=0; $P_0=0$; $C_2C_1C_0= B_2B_1B_0$

Divide operation

$(A_5A_4A_3A_2)/(B_2B_1B_0)=(CK_2CK_1CK_0)$

With these conditions: X=1; $C_2C_1C_0= B_2B_1B_0$

Squaring operation

$(P_1P_2)^2=(S_6S_5S_4S_3S_2)$

With these conditions: X=0; $P_0=0$; $C_i=B_i=P_i$ (i=1,2)

$C_1=B_1=0$

Square rooting operation

$\sqrt{(A_5A_4A_3A_2A_1A_0)} = (CK_2CK_1CK_0)$

With these conditions: X=1; B_1C_1=00; B_2C_2=; B_iC_i=10

4. Results

We simulate the design using VHDL and to evaluate it and investigate that the simulated design is working properly we test the design and display the result as appeared in the VHDL program as shown in figures (4,5,6,7) and the table(1) summarize the results as a table of values where the first column represent the type of the operation performed and the other columns represent the values of (A,S,X,P,CK,B,C) and each of these values was mentioned in the past section.

Figure (5) shows the waveforms produced from the Multiply operation that implemented on the design and these values in the waveform are described in the first row of table(1) we can note that the value of X should be equal to 0 and the multiply occur between P_1P_2 and $B_2B_1B_0$ where P_1 and B_2 is the MSB and the result is stored in $S_6S_5S_4S_3S_2$, P_0 should be equal to the value of X also the C vector equal to the B vector.

Figure (6) and Figure (8) shows the waveforms produced from the Divide and square root operations respectively and these values in the waveform are described in the second and fourth rows of table(1). The value of X should be equal to 1 in the divide and square root operations and the result will be stored in $CK_2CK_1CK_0$ where CK_2 is the MSB. In the divide operation the result is produced from the division of ($A_5A_4A_3A_2$) by ($B_2B_1B_0$) and ($B_2B_1B_0$=$C_2C_1C_0$) but in the square root operation B_2C_2=00, B_1C_1=01, B_0C_0=10. Also in divide and square root operations the value of P_0 only entered and P_1P_2 don't matter on the result.

Table (1): summary of the results *(U mean Unspecified(don't care))*

OPERATIONS	X	$S_6S_5S_4S_3S_2$	$A_5A_4A_3A_2A_1A_0$	$B_2B_1B_0$	$C_2C_1C_0$	$P_2P_1P_0$	$CK_2CK_1CK_0$
MULTIPLY	0	00001	000000	001	001	100	000
	0	00010	000000	010	010	100	000
	0	00100	000000	010	010	010	000
	0	00110	000000	011	011	010	000
	0	01010	000000	100	100	110	000
	0	01111	000000	101	101	110	000
	0	01010	000000	101	101	010	000
DIVIDE	1	00000	1001UU	011	011	UU1	011
		00000	1111UU	011	011	UU1	101
		00000	0011UU	011	011	UU1	001
SQUARE	0	00001	000000	001	001	100	000
		00100	000000	010	010	010	000
		00000	000000	000	000	000	000
SQUARE ROOT	1	00000	100100	001	010	UU1	110

Finally the square operation shown in figure (8) and described in the third row in the table(1) in which B_2=X=0, and B_1B_0=P_1P_2 , the result will be stored in $S_6S_5S_4S_3S_2$.

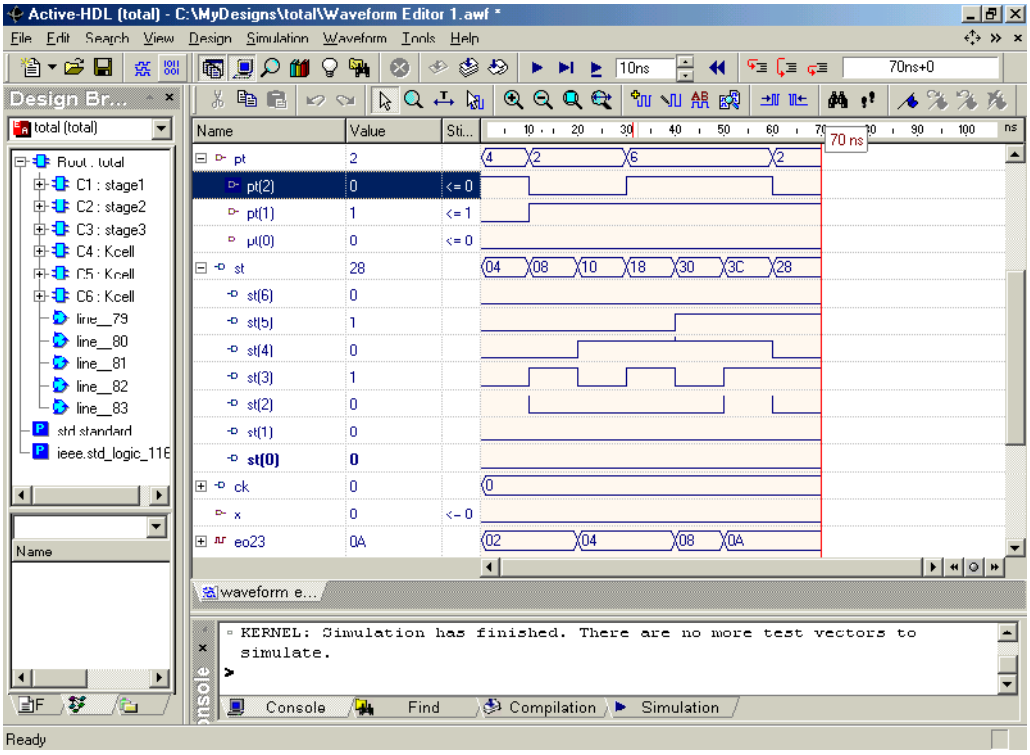
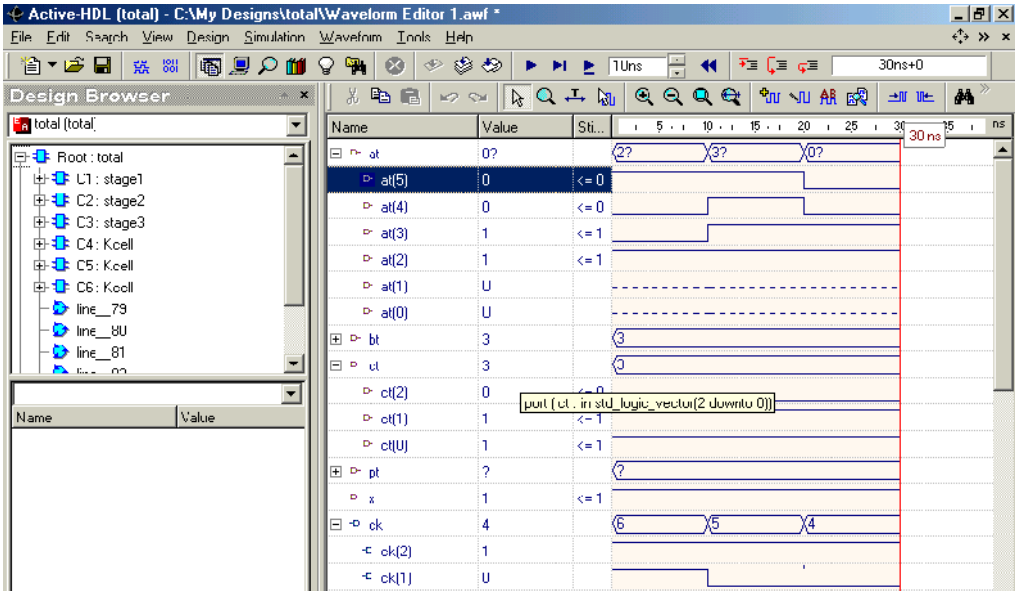


Figure 5. The Multiply operation



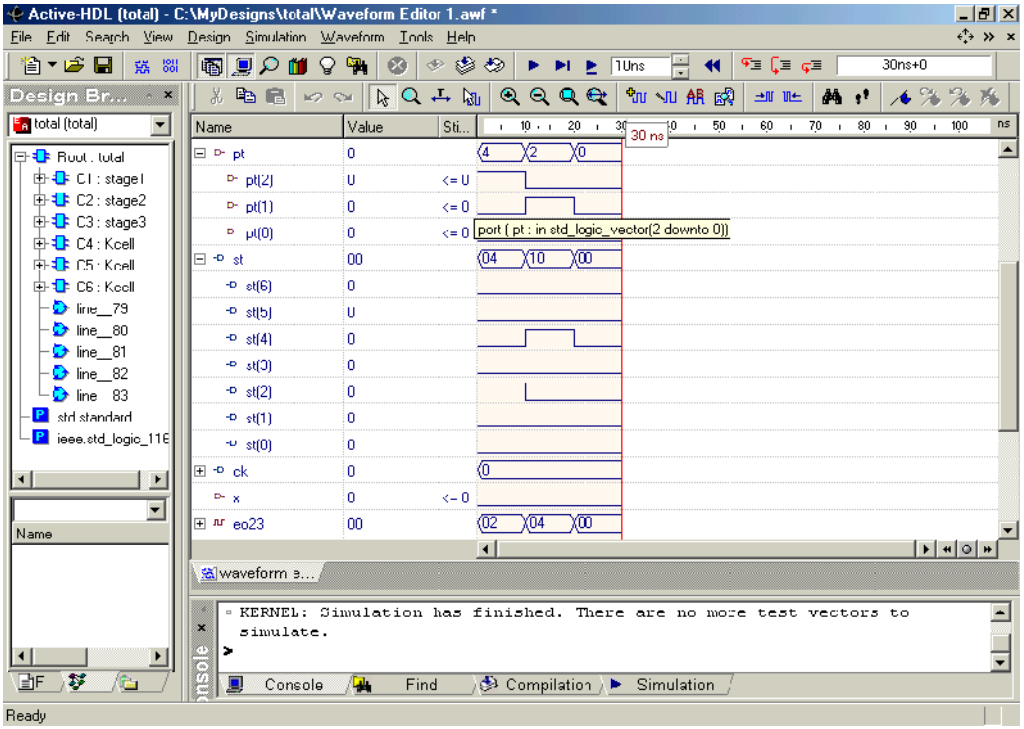


Figure 7. The Square operation

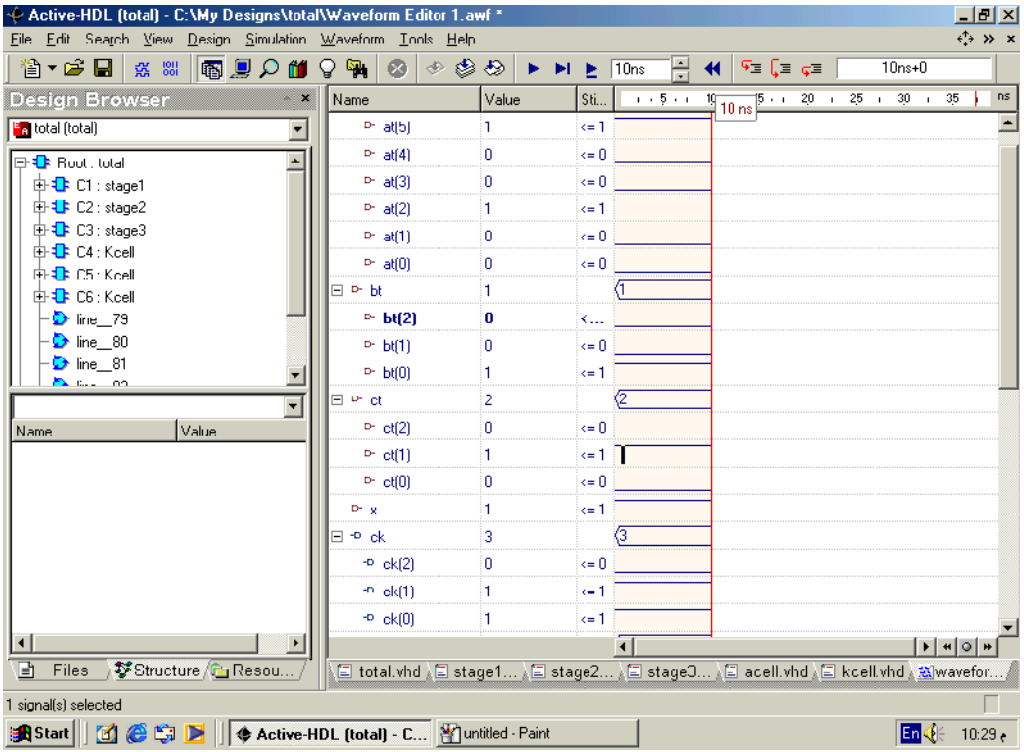


Figure 8. The Square root operation

The multifunction arithmetic pipeline was implemented on an FPGA of type XC4003E that its parameters are shown in table (2).

Table (2) XC4003E Parameters

No. of CLBs	100
Maximum logic gates	3000
IOBs	80
Flip-Flops	360
Bits per frame	126
Frames	428
Program data	53,936
PROM Size (bits)	53,984

Table (3) Consumed resources in the XC4003E FPGA to implement the multifunction arithmetic pipeline.

Resource type	No. of consumed resources	Consuming percentage
CLB	24 out of 100	24%
4 inputs LUTs	38 out of 200	19%
3 inputs LUTs	15 out of 100	15%
IOB driving Global buffers	0 out of 8	0%
Bonded IOB	16 out of 80	20%
IOB flops	0	-----
IOB latches	0	----
Clock IOB pads	0 out of 8	0%
Primary Clks	0 out of 4	0%

After implementing the arithmetic pipeline design on the FPGA, the ISE(Integrated software Environment) s/w generates information about the design in the form of report that shown in table (3). This table reveal the resources of FPGA that have been exhausted to implement the design.

Conclusions

In this paper we proposed architecture for the MAC units as processor core using reconfigurable FPGAs. Our design grew from an initial desire

to build a small, fast, and inexpensive sequence analysis machine into a general purpose parallel processor.

In addition it will enable many applications previously not possible due to the inflexible nature of ASIC implementations and the cost/performance/power inefficiency of previous programmable approaches. Simulation results show that our design is highly applicable and can be used by the developers.

References

1.Jennifer Eyre and Jeff Bier, "The Evolution of DSP Processors", Berkeley Design Technology,Inc.(BDTI white paper). http://citeseer.ist.psu.edu/eyre00evolution.html.

2. D.R.Martinez, T.J.Moeller, and K. Teitelbaum. Application of Reconfigurable Computing to a high performance Front-end Radar Signal Processor. Journal of VLSI Signal Processing, vol. 28, no. 1-2, pp. 63-83, May 2001.

3. S. Yalamanchili, "Introductory VHDL from Simulation to Synthesis", Prentice-Hall, Inc, 2001.

4. Xilinx Development System "VHDL Reference Guide", Xilinx, Inc, 1999.
http://toolbox.xilinx.com/docsan/data/alliance/dev/dev.htm

5. S. Brown and Z. Vranecis "Fundamentals of Digital Logic with VHDL Design" McGraw-Hill, Inc, 2000.

6. J. Bahsker "VHDL Primer", Prentice-Hall, Inc, 1999.

7. P. K. Chan "Digital Design Using FPGAs", Prentice-Hall, Inc, 1994.

8. J. R. Armstrong "VHDL Design Representation and Synthesis", Prentice-Hill, Inc, 2000.

9. C.T. huang, P.C. Tseng, and L.G. Chen, "Flipping structure: An Efficient VLSI Architecture for Lifting-based Discrete Wavelet Transform", IEEE Transactions on Signal Processing, Vol. 54. No. 4, pp. 1080-1089, April 2004.

10. Tinku Acharya and Ping-Sing Tsai, "JPEG2000 Standard for Image Compression; Concepts, Algorithms, and VLSI Architecture", John Wiley and Sons, Inc., 2005.

The work was carried out at the college of Engg. University of Mosul