LOW POWER 3-D DISCRETE WAVELET TRANSFORM ROCESSOR FOR MEDICAL APPLICATIONS

Wael Badawy, Guoqing Zhang, Mike Talley, Michael Weeks and Magdy Bayoumi
The Center for Advanced Computer Studies
University of Southwestern Louisiana
Lafayette, LA 70504
{wmb, gxz0233, mxt1206, mxw, mab}@cacs.usl.edu

Abstract - This paper presents a low power 3-D discrete wavelet transform processor for medical ›plications. The main target of this work is the compression of Magnetic Resonance Imaging ‹IRI) data using a wavelet based scheme. A prototype has been developed to realize a 3-D wavelet ›mpressor. The processor is based on an architecture that uses a centrally controlled unit to ›ordinate sub-systems which carry out the operations necessary for the data compression. The sub-‹stems include a small cache memory used for block data storage, parallel high and low pass filters ›ed for data calculation, and two coefficient units used to retrieve off-chip wavelet coefficients. The ›rocessor has been prototyped using 0.6 μm CMOS (three metal) technology, the prototype ›rocessor is modular. It has been simulated at the functional, circuit, and physical levels. The ‹erformance measures of the prototype, area, time delay, power and utilization has been evaluated. he prototype operates at an estimated frequency of 272Mhz and dissipating 0.5W of power.

NTRODUCTION

Wavelet analysis applies to many different fields, such as "digital communications, remote ›ensing, biomedical signal processing, medical imaging, astronomy and numerical analysis" [1], ›s well as digital signal compression, and noise reduction.

Discrete Wavelet Transform DWT can be used in several medical applications, such as ›nding cancer [2] in mammogram images, monitoring of fetal heartbeats, ultrasounds, and ›nalyzing electrocardiograms. One massive problem of medical images is the volume of data produced, so medical images must be compressed.

DWT considers correlation of images, which translates to better compression. According to Lee, et al., the 3-D DCT is more efficient than the 2-D DCT for x-ray CT [3]. Likewise, one would expect the 3-D DWT to outperform the 2-D DWT for MRI. Wang and Huang showed the 3-D DWT to outperform the 2-D DWT by 40-90% [4]. The DWT has advantages over the DCT. First, DCT difference coding is computationally expensive. Second, wavelets do not cause blocking artifacts, which are unacceptable in medical images. The DCT is not optimal for medical image compression, even if it performs well on video [5]. Currently, 3-D transforms have been done with other methods, like a network of computers, but a chip dedicated to this transform will give faster results. Due to better medical input devices (scanners) interslice distances get smaller, improving the correlation between images, resulting in even better results for the 3-D transform.

Medical applications deal with vast amounts of data as the following example illustrates. Lee, et al. reports that "the University of Washington Medical Center, a medium-sized hospital

0-7803-5650-0/99/$10.00 © 1999 IEEE

with about 400 beds, performs approximately 80,000 studies per year. At 30 Mbytes per stu the amount of digital images generated is 2.4 Tera (10^{12}) bytes of data per year approximately 10 Gbytes per day" [3]. The wavelet transform allows for a good deal compression without losing diagnostic accuracy. As Angelidis notes, MRIs "must be stored i long periods of time both for medical and legal reasons" [6].

Medical resonance images (MRI) are a series of 2-D slices used in medical diagnosti Compressing this volume of data is important since it needs a large amount of storage spa and/or bandwidth. A 3-D wavelet transform takes advantage of the fact that the third dimensi is related to the first two dimensions to provide a more efficient transform [7]. Each MRI pix in the test sequence is a 1 mm x 1 mm x 4 mm section of a human head, with 4 mm being t gap size between slices. Thus, the MRI completely maps out a 3-D body part as a 3-D block data. A typical test sequence has 53 slices of 256x256 pixels, and each pixel value has a wid of 8 bits.

To compress 3-D data, there are three steps. First, encode the data using a 3-D wavel transform. Second, an entropy constraint quantization eliminates transformed data below certain threshold. Third, code the quantized data using run-length coding followed by Huffm coding to reduce the amount of data. The decompression process simply reverses these step Alternatively, other quantization methods have been researched and shown to give even bett results, such as zero-tree encoding [7].

Many architectures for the Discrete Wavelet Transform (DWT) have been proposed sinc 1990. The DWT is not a straight-forward problem to implement on a chip, so there are sever types of designs. Different DWT architectures have been proposed in the last decade. All c them vary in terms of scalability (e.g. number of filter coefficients), area, control complexity latency, and memory. These architectures depend on the degree of the DWT i.e. 1-D, 2-D, or 3 D. Different design approaches have been reported such as systolic [8], semi-systolic [9, 10] folded [11], digit-serial [11], etc. Some architectures use centralized control structure [12, 13] while others use distributed control structure [14]. The centralized control signal is easier t implement, but it suffers from the limitation of the scalability [15]. Organization of the memor architecture varies from architecture to another on the following basis MUX-based, systolic RAM, reduced storage, and distributed [16].

This paper presents a 3D-processor prototype based on the 3D architecture in [15]. Severa design issues and parameters are addressed, analyzed and evaluated. The prototype processo consists of a single filter pair performing the calculation for one dimension. The output of each filter is stored in on-chip memory, it is considered as input for the next step. The filters are efficiently folded with an adequate amount of memory between the filters. The proposed architecture is area efficient, provides parallel execution for the filters and consumes less power

The organization of this paper is as follow: Section 2 introduces the proposed architecture. Section 3 desribes the prototype processor. The prototype evaluation is presented in Section 4. Section 5 decribes the implementation and simulation results and conclusions are summarized in Section 6.

HE PROPOSED ARCHITECTURE

The processor architecture is based on centralized filters and it does not require large on-ıip memory. The processor reads the data as blocks instead of the row-column fashion. Of ıurse, a large RAM could buffer the row-column data and send it out in block format. With is capability, it needs a data block of size L1*L2*L3 to compute the 8 output streams of the 3-DWT. Figure 1 shows what this looks like for the filter sizes 4, 4, and 2. If another wavelet ere used instead, the only changes besides the filter lengths would be the block size and on-ıip memory. The block size would have to be 9x9x2 for the 9-7 biorthogonal spline, since the and 7 sizes apply to the low and high pass filters respectively, and the data in each dimension ill be sent to both low and high pass filters. Figure 1 shows how the transformer would request ıe data blocks. The block selector moves from left to right and front to back. The blocks are ad, skipping every other block horizontally, vertically, and between images in order to take ɔwnsampling into account.

Figure 2 shows how one block of data becomes 8 outputs for the 3-D DWT for a 4x4x4 ansform. First, a 4x4x4 data block goes through the X filters. Each X filter takes 4 inputs and oduces 1 output. So the intermediate data is now 2 blocks of 1x4x4. Next the Y filters each ake 4 inputs and produce 1 output, forming 4 blocks of size 1x1x4. Finally, the Z filters take in inputs and produce 1 output each. Thus, the end result is 8 blocks of 1x1x1, or simply 8 utputs. This holds for any size filter - a $L_1xL_2xL_3$ data block will become 2 $1xL_2xL_3$ blocks, hen 4 $1x1xL_3$ blocks, then 8 1x1x1 blocks as it passes through the transformer.

The proposed architecture is shown in Figure 3. The centralized control unit will be esponsible for coordinating all activity of the processor. Initially, it will retrieve all off chip ata and load it into the system. The off-chip memory could be a hierarchy, such as a RAM and ache setup. The control unit generates the addresses needed to get the correct input block. hough the figure does not explicitly show it, the control unit will also select which coefficients o pass on to the low and high pass filters. The memory on the chip will be small compared to he input size.

Multiplexing the calculations in space instead of time is the best choice for this architecture. This design choice is made due to the fact that time multiplexing slows down the architecture. For the 3-D DWT, speed is very important. There is one low and one high pass filter calculation done in every cycle. Also, the filters should be parallel, since this produces the computation with the least latency. The processor calculations are done on one block as an atomic operation. No partial calculations are done, and a minimal latency is needed.

THE PROTOTYPE PROCESSOR

The prototype processor consists of one processing element, which contains both the high pass and low pass filters and 1 K on-chip cache. The wavelet coefficients are loaded into the 16-bit coefficient registers. Figure 3 shows a generic block diagram of the 3-DWT processor. The typical sequence of operations involves loading data blocks into the cache from an off-chip memory, while the transform coefficients are loaded into the coefficient register. Then the 3-D

wavelet transform is scheduled using the coefficients and data block as 3 consequent wavelet transform. Figure 3 shows the complete system including all modules and ti connectivity. In our proposed prototype, low power presents a major performance factor. L power has been considered at several levels. It is achieved by applying the following strate in the design flow:

1. Usage of specially designed low power building block cells such as adder, multiplier, etc
2. Minimizing the number of processing elements by using the Central Control design introducing less circuit complexity in the architecture.
3. Maximizing the usage of sub-chip components by eliminating any redundant modules.
4. Making compromises concerning the tradeoffs of power, speed, and circuit complexity.

The High Pass and the Low Pass Filter Both the High Pass and the Low Pass filters identical with respect to their internal organization, Figure 3, although their respect coefficients vary. The filters include 16-bit Carry Look Ahead (CLA) adders and 16 bit Bo multipliers, to perform the arithmetic computations. After considering the attribute of ma adders, the CLA was found to have the best possible trade-off in terms of power, speed, s and complexity. Figure 3 shows the filter computational tree where the * is the 16-bit Bo Multiplier and the + is the 16 bit-Carry Look Ahead adder, which uses a basic low power 1- adder cell [18]. The cell inherits a computational parallelism by using two independent d paths for calculating the Sum and the Carry. The output from the Booth Multiplier is 32 bits i the most significant 16 bits are discarded as described in [17]. After the layouts are generat we use compactor to compact layout to a more regular shape. This compaction uses *MOSI* based rules, less compact each dimension independently.

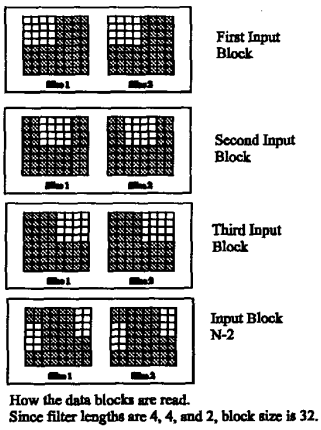


Figure 1 Block reads for the architecture

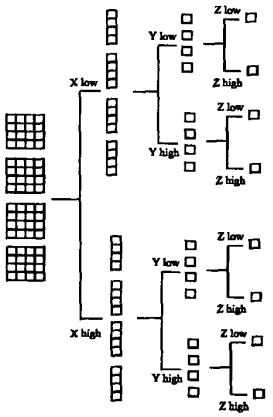


Figure 2 How a data block becomes 8 output

The Coefficient Module where DWTC is the discrete wavelet transform coefficient 64-b register. The R0, R1 and R3 are three 64 bit registers that contain the discrete wavelet transfor

oefficient. The MUX is a 3-1 multiplexor, it selects which register is passed on to the control nit. Each 64-bit register is simultaneously loaded from an off-chip memory with four different 6-bit wavelet coefficients.

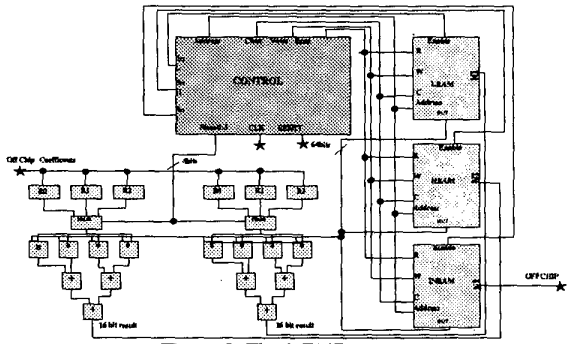


Figure 3: The 3-DWT processor

The Memory Module Unit the prototype processor uses 1K on-chip cache memory. The cache unit is required for temporary storage of intermediate wavelet values. The unit consists of three independent cache modules, the INRAM, LRAM and HRAM. The INRAM module stores a whole block from an off-chip memory while the LRAM and HRAM modules contain the computed results from the filter modules. The INRAM has a 64-bits input/output data width, while both the LRAM and the HRAM has a 16-bit input data width, and 64-bit output data width. This is to accomplish the filters input/output width.

The Control Unit coordinates the wavelet processor operations. It is basically a state machine, which consists of 25 internal cycles. Nine states are used for loading off-chip data, while the remaining states are used for the wavelet computation. Figure 6 illustrates the states of the controller. This sates show the flow of the required steps to compute the wavelet transform. The processor loads the eight data blocks from off-chip memory in states 1-8. The controller sends the necessary signals required for this operation. After the data is loaded from off-chip, intermediate results are calculated.

Both low pass and high pass filters generate the output using the off-chip data during states 9-16. The final values of the wavelet calculations are performed in states 17-25, using the intermediate data calculated in clocks 0-7 during clocks 9-16.

ARCHITECTURE EVALUATION

The architecture uses programmable 16 bits wavelet coefficients, which are loaded from an off chip memory. Giving to perform the DWT with different wavelet coefficients. It uses fixed-point numbers with an assumed radix. The architecture does all calculations for 8 outputs as an atomic calculation. Storing partial results for later use requires additional memory and an increase in control complexity. The RAM unit of the architecture needs to store the input block, and the X and Y filter outputs. The RAM must conceptually read L_1 16-bit values each cycle,

and write up to three values: L_1 16-bit input values (regarded as one large input), and two 16-b filter output values.

The architecture can easily work in parallel with others. Placing more than two processo next to each other does not even require revision of the architecture. All they would need i different (though slightly overlapping) data streams. The overlap comes at the boundaries. Fc example, if processor 1 does the 3D DWT on slices 1-16, and processor 2 handles slices 17-3 processor 1 will need to access slice 17 while processor 2 will need access to slice 1, assumin circular inputs.

Clock cycle analysis The number of clock cycles needed per block varies with the filter sizes It is assumed that the busses are L_1 values wide within the chip. The architecture processes th X dimension first, which will take L_2*L_3 clock cycles to read L_2*L_3 values and write L_2*L_3* values. The Y dimension will get L_2*L_3*2/L_2 values and store L_3*2*2 values. Finally, the Z dimension will obtain L_3*2*2/L_3 values and send 8 outputs off-chip. For every value read (o step), there must be 1 clock cycle, so that the total number of clock cycles per block = L_2*L_3 + $L_3*2 + 4$ clock cycles. Thus the total number of clock cycles for the architecture is:

Total clock cycles = $(N*M*P / 8) * (L_2*L_3 + L_3*2 + 4)$

Each clock cycle of the architecture involves $\max(L_1, L_2, L_3)*2$ multiplications, since there are two filters of size $\max(L_1, L_2, L_3)$. When the number of coefficients does not match the filter size, the coefficients are padded with zeroes. This contributes to the number of multiplications, though the multiplier's results are not used. The number of additions per clock cycle also depends on largest filter size, since it performs $\max(L_1, L_2, L_3)-1$ additions. The multiplications are done in parallel, while a tree of adders calculate additions, taking $\log_2(\max(L_1, L_2, L_3))$ amount of time.

Table 1 lists the lowest operating frequency for the two architectures to achieve 30 fps and 60 fps. Remember that these results are based on one of each chip. Two processors running in parallel only need to operate at half the clock rates listed below. These tables come from the frames per second equations:

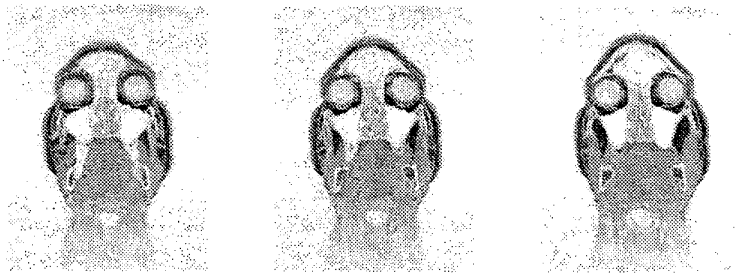
fps = $8*Speed / (NM*(L_2L_3 + 2L_3 + 4))$

Image size:	256x256		512x512		720x1280		1080x1920	
FPS:	30	60	30	60	30	60	30	60
4x4x2	4	8	16	31	55	111	124	249
12x12x4	15	29	59	118	207	415	467	933

Table 1 - Minimum Clock Speed (in MHz)

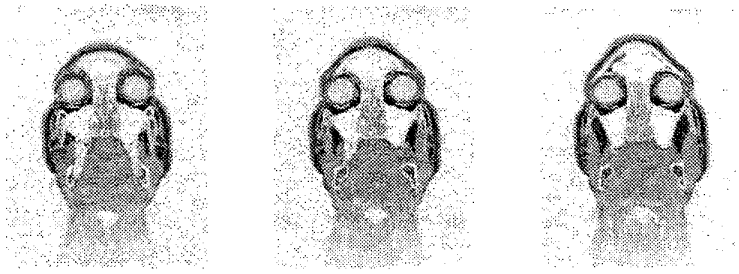
Compression results - The following experiments analyzed 3-D MRI data with different wavelets and compressed the results. Design decisions about the architecture, such as precision, affect the results. Therefore, the simulation takes architecture characteristics into account. The low-low-low pass results were stored in 16 bit integer format, while the detail signals were quantized and stored in 8 bit integer format. Low-low-low pass results have a range of -32768

32767, while the detail signals have a maximum range of -128 to 127. The following figure ve examples of the original and the wavelet the compressed ratios.



1 2 3

Original



1 2 3

Wavelet: 4x4x2 1 Octave Compression ratio: 1.51

Figure 4 Decompression Results

IMPLEMENTATION AND SIMULATION RESULTS

The chip has been implemented using 0.6 μm CMOS technology with three layers of metal. Synopsys™ tools are used for the "front-end" steps, which capture and simulate the desired behavior of the architecture, while Cadence™ tools are used for the "back-end "steps.

The behavior of the architecture is modeled using Verilog™, and it is synthesized, simulated and analyzed using Synopsys™. Then, Cadence ™ is used to do schematic capture and modification, using gate-level Verilog code to generate final layout.

The technology libraries and symbol libraries that were used in both Synopsys™ and Cadence™ for synthesis and simulation were generated based on the standard-cell library. This guarantees that the synthesized design has the nearest simulation results as the final layout.

Transferring the design from Synopsys™ to Cadence™ is accomplished by generating Verilog™ netlist, which creates symbol and schematic views for all levels of the desi hierarchy. Power and ground pads are then added to the top-level schematic.

Two groups of simulation are used, the first one, proving the correctness of the design different design steps, while the second one measuring different performance parameters for i proposed architecture.

Many verification simulation steps are involved in the design flow. The first one, us Verilog simulation tools to verify the correctness of the architecture's behavior. The second applied to confirm the functionality of the extracted gate level circuit. The third ensures that ti gate-level schematic, which has been imported and edited in Cadence™, is equivalent to ti proposed architecture. These are followed by the DRC (Design Rule Checking) and the LV (Layout Versus Schematic), which are done with Cadence Diva. Once LVS is done, it assumed that the layout is functionally correct. Figure 5 show the processor layout, which has total area of 9935.50 x 6236.40 μm^2. The evaluation of the implemented architecture conside different parameters as follows:

Area: The prototype of the architecture uses 168k transistors, the areas of the mai building blocks are as follows: (Table 2)

Power Consumption: According to the Synopsys™ simulation, the architecture expected to consumes power 0.5 W. Table 2 shows the power evaluation of the main buildin blocks.

Latency: The latency measures the speed of the operation. In this evaluation, we expres the latency as a function of the input rate. Table 3 shows the time delay for the building block The latency for the processor is to perform the 25 states, described in Section 3.

CONCLUSIONS

This paper presents a low power 3-D DWT processor based on centralized control uni architecture. The simulation results show the efficiency of the wavelet processor. The prototyp processor consumes 0.5 watt with total delay of 91.65 ns. The processor operates at ma frequency of 272 Mhz. The prototype processor uses 16-bit adder, 16-bit Booth Multiplier an 1 KB cache with a maximum of 64-bit data bandwidth. Lower power has been achieved b using low power building blocks and the minimal number of computational units with hig throughput.

The architecture has a single low/high filter pair to compute all the outputs. It requires small amount of storage, based on the filter size, $O(L_1 x L_2 x L_3)$, and not the input size. The filte sizes are much smaller than the input size ($L_i << N$). The latency is small, it depends on th filter sizes and not the input size. For example, using a 4x4x2 wavelet, the first output comes a the 12th clock cycle. The architecture can be used in parallel to transform the data in half th time (or less, if more than 2 are used). Complex control and the large number of clock cycle are the major drawbacks. Also, a large off-chip buffer is needed to allow block inputs.

:FERENCES

Andrew Bruce, David Donoho, and Hong-Ye Gao, "Wavelet Analysis," *IEEE Spectrum*, October)6, pages 26-35.

W. Wayt Gibbs, "Making Wavelets," *Scientific American*, June 1996, pages 137-138.

Heesub Lee, Yongmin Kim, Alan H. Rowberg, and Eve A. Riskin, "Statistical Distributions of DCT :fficients and Their Application to an Interframe Compression Algorithm for 3-D Medical Images," *EE Transactions of Medical Imaging*, Volume 12, Number 3, 1993, pages 478-485.

Jun Wang and H. K. Huang, "Three-dimensional Medical Image Compression using a Wavelet ansform with Parallel Computing," *SPIE Imaging Physics* [Society of Photo-Optical Instrumentation ·gineers], Yongmin Kim (editor), San Diego, Volume 2431, March 26-April 2, 1995, pages 16-26.

] John D. Villasenor, "Alternatives to the Discrete Cosine Transform for Irreversible Tomographic ıage Compression," *IEEE Transactions of Medical Imaging*, Volume 12, Number 4, 1993, pages 803-1.

] P. A. Angelidis, "MR Image Compression using a Wavelet Transform Coding Algorithm," *Magnetic :sonance Imaging*, Volume 12, Number 7, 1994, pages 1111-1120.

] Michael A. Pratt, C. Henry Chu, and Stephen Wong, "Volume Compression of MRI Data using ·otrees of Wavelet Coefficients," *Proceedings Wavelet Applications in Signal and Image Processing '*, Michael A. Unser, Akram Aldroubi, Andrew Laine (editors), Denver, Colorado, August 16-19, 1996, ·ges 752-763.

] M. Vishwanath, R. M. Owens, and M. J. Irwin, "Discrete Wavelet Transforms in VLSI," *Proc. of the :t. Conf. on Application Specific Array Processors*, Berkeley, August 1-2, 1992, pp 218-229.

] Jimmy Limqueco and Magdy Bayoumi, "A Scalable Architecture for 2-D Discrete Wavelet ransform," *VLSI Signal Processing IX*, San Francisco, Oct. 30-Nov. 1, 1996, pp 369-377.

0] Jimmy Limqueco and Magdy Bayoumi, "A VLSI Architecture for Separable 2-D Discrete Wavelet ransform," *Journal of VLSI Signal Processing*, Vol 18, 1998, pp 125-140

1] Keshab K. Parhi and Takao Nishitani, "VLSI Architectures for Discrete Wavelet Transforms," *IEEE ransactions on Very Large Scale Integration (VLSI) Systems*, Vol. 1, No. 2, 1993, pp 191-202.

2] Michael Weeks, Magdy Bayoumi, "3-D Discrete Wavelet Transform Architectures," *IEEE Int. ymposium on Circuits and Systems (ISCAS '98)*, Monterey, California, May 31 - June 3, 1998.

3] G. Knowles, "VLSI Architecture for the Discrete Wavelet Transform," *Electronics Letters*, Vol 26, Jo. 15, 90, pp. 1184-1185.

4] Jose Fridman and Elias S. Manolakos, "Distributed Memory and Control VLSI Architectures for the -D Discrete Wavelet Transform," *IEEE Proc. VLSI Signal Processing VII*, La Jolla, California, October 6-28, 1994, pp. 388-397.

5] Jose Fridman and Elias S. Manolakos, "Discrete Wavelet Transform: Data Dependence Analysis and ynthesis of Distributed Memory and Control Array Architectures," *IEEE Transactions on Signal rocessing*, Vol. 45, No. 5, May 1997, pp 1291-1308.

6] Chaitali Chakrabarti, Mohan Vishwanath, and Robert Owens, "Architectures for Wavelet ransforms: A Survey," *Journal of VLSI Signal Processing*, Vol. 14, No. 1, 1996, pages 171-192

[17] Michael Weeks, Magdy Bayoumi, "3-D Discrete Wavelet Transform Architectures," *IEEE Int. Symposium on Circuits and Systems (ISCAS '98)*, Monterey, California, May 31 - June 3, 1998.

[18] W. M. Badawy, A. M. Shams and M. A. Bayoumi "An Enhanced Low-Power Pipelined Multiplie Accumulator for DSP Applications", *The 7Th NASA Symp. on VLSI*, Albuquerque, NM Oct.1-2,98.

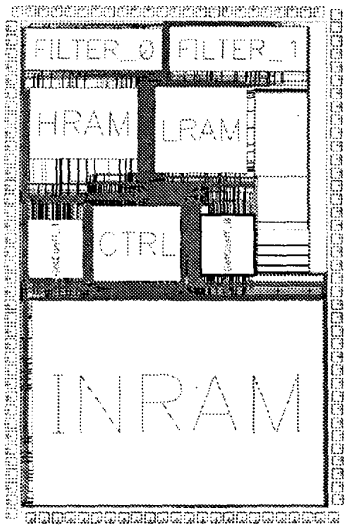


Figure 5: The Processor Layout

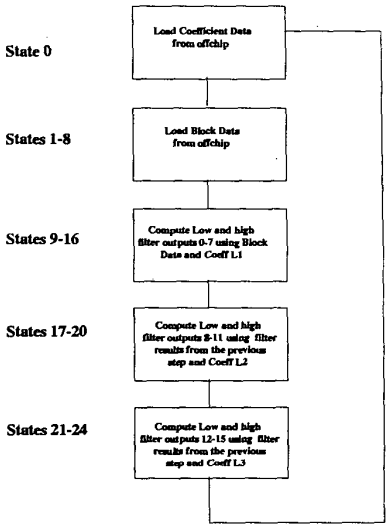


Figure 6: States of Controller

The block	The Area in (μm^2)
16-bit adder	368.80 X 99.70
16-bit booth multiplier	954.50 X 476.60
RAM	1883.50 X 1753.60

Table 2: Area of different building block

The block	The Power (mW)
16-bit adder	5.5828
16-bit booth multiplier	13.0729
The processor	500

Table 3: The Power of different building bloc

The block	The delay (ns)
16-bit adder	4.7
16-bit booth multiplier	8.93
The processor	91.65

Table 4: The delay of different building block