Tina Memo No. 1995-001
Presented at: ICPR,3, pp266-273, Jerusalem Oct. 1994.

# Specification and Design of a General Purpose Image Processing Chip.

N.A.Thacker, P.Courtney, S.N.Walker, S.J.Evans and R.B.Yates.

Last updated
1 / 12 / 1998

This document forms part of the **Recognition and Intelligence** Series available from www.tina-vision.net.

| | |
|---|---|
| 2007-001 | Retinal Sampling, Feature Detection and Saccades: A Statistical Perspective. |
| 2006-008 | Statistical Principles for Selection of Computer Vision Algorithms as Modules for Visual Perception - Show Me the Errors. |
| 1991-001 | Designing a Layered Network for Context Sensitive Pattern Classification. |
| 1997-002 | Supervised Learning Extensions to the CLAM Network. |
| 1996-003 | Tutorial: Algorithms For 2-Dimensional Object Recognition. |
| 1997-005 | Speechreading Using Probabilistic Models. |
| 2000-002 | Solving Shape Based Object Recognition from a Computational Standpoint - Practical and Physiological Constraints. |
| 1995-004 | Assessing the Completeness Properties of Pairwise Geometric Histograms. |
| 1996-004 | Robust Recognition of Scaled Shapes Using Pairwise Geometric Histograms. |
| 1996-005 | Multiple Shape Recognition Using Pairwise Geometric Histogram Based Algorithms. |
| 2007-007 | Automatic Identification of Morphometric Landmarks in Digital Images. |
| 1999-002 | A Feature Representation for Map Building and Path Planning. |
| 2001-015 | Colour Image Segmentation by Non-Parametric Density Estimation in Colour Space. |
| 2001-006 | What is Intelligence?: Generalised Serial Problem Solving. |
| 1994-002 | A Correlation Chip for Stereo Vision. |
| 1995-001 | Specification and Design of a General Purpose Image Processing Chip. |


Imaging Science and Biomedical Engineering Division,
Medical School, University of Manchester,
Stopford Building, Oxford Road,
Manchester, M13 9PT.

# Abstract

The computational and communication requirements of a wide range of image processing algorithms were analysed and used to specify the design of a general purpose image processing device. The algorithms considered are restricted to image-in and image or feature-out type algorithms. These algorithms are catagorised by the deamands that they would place on a hardware architecture. We conclude that the major consideration is the off-chip input/output limitation which needs to be able to exploit redundancy in both the data and processor instructions. This paper summarises: the main conclusions of this work, an architecture which we have designed in an attempt to meet the identified constraints and how it differs from previous attempts.


# 1 Introduction

Our interest is to develop robust and high performance computer vision systems for real applications. The aim of the work reported here has been to identify a basic core requirement of vision algorithms and to see to what extent it is possible to implement a general purpose computational architecture using VLSI technology. This paper summarises the main conclusions and describes an architecture which we have designed in an attempt to meet the identified constraints.

# 2 Analysis of algorithms

A general purpose processor should (by definition) be capable of perfoming a range of processing tasks and these need to be identified before a general purpose architecture can be designed. If we consider the broad range of data manipulations normally required in the areas of image processing, image compression and machine vision it is possible to conclude quite quickly that there is a great degree of overlap between the computational requirements of each of these fields. However, it is clear that there are many such algorithms in the literature and a complete review would be impossible. We therefore restricted our selection and analysis in three ways. Firstly we tried to identify those algorithms which would be considered as fundamentally useful. Secondly we tried to concentrate on implementations of algorithms which could be considered fundamentally parallel in nature and which preserved the organisational structure of the processed data (ie image based operations). Thirdly we attempted to select algorithms which conform to the basic configuration of input/output requirements defined below. These constraints were imposed in order to focus on those algorithms for which a hardware architecture may benefit from regular data structures and data reuse. The resulting set of algorithms can be considered as those which could be implemented on an image processing pipeline. We believe that the resulting selection is representative of the kind of computationally intensive component of algorithms that are common in the areas of image communication [6], image processing and machine vision [9] [7] [1].

In order to get a feel for the required data processing scheme we have attempted to broadly categorise the various algorithms in terms of input/output (I/O) and computation. Making no assumptions about the computation architecture at this point, we look only at the minimum amount of new data required before each step in the computation. Most of the algorithms can be characterised by three data input and output categories:

- *A*: Loading a row of image data into one register of the processor.
- *B*: Loading an image block into one register of the processor.
- *C*: Loading blocks of pre-defined coefficients into one register of the processor.

- *a*: Those which generate a small quantity of pixels or other values.
- *b*: Those which generate a row of output image data at a time.
- *c*: Those which generate a block of output image data at a time.

The algorithms can then be further subdivided in terms of processing requirement:

- 1: A pixel Operation.
- 2: A 2D Image Operation per block.
- 3: Multiple 2D Image Operations per block.

The algorithms considered in this analysis are summarised in table 1.

Those algorithms which are likely to present the greatest difficulty to a general purpose processor are those which contain categories $C$ and $a$ (I/O bound) and those from catagory 3 (computation bound). Algorithms with catagory $b$ output are not regarded as I/O bound since they can generate relatively larger portions of the output image in a given time than those in catagory $a$.

A similar analysis was carried out by Cipher and Sanz [2] to evaluate the suitability of a range image processing algorithms to a number of SIMD communications architectures. Our analysis differs in that we have specifcally excluded pointer manipulation common in higher level symbol processing tasks but have included compression-related and neural networks algorithms which have grown in importance in recent years.

| **Algorithm** | **In** | **Out** | **C** |
|---|---|---|---|
| IMAGE PROC. | | | |
| Thresholding | $B$ | $c$ | 1 |
| Histogram | $B$ | $b$ | 1 |
| Maxima detection | $B$ | a | 1 |
| Filtering | $B$ | $c$ | 1 |
| Median filtering | $B$ | $c$ | 2 |
| COMPRESSION | | | |
| Block Match | $A$ | a | 3 |
| Vector quantisation | $B+C$ | a | 2 |
| Differential coding | $B$ | $c$ | 1 |
| 2D transforms | $B+C$ | $c$ | 2 |
| Fractal coding | $B$ | a | 3 |
| MACHINE VISION | | | |
| Image algebra | $B+B$ | $c$ | 1 |
| Morphology | $B$ | $c$ | 1 |
| Convolution | $A$ | $b$ | 3 |
| Correlation | $B+C$ | a | 2 |
| Image interpolation | $A+C$ | $b$ | 3 |
| Neural networks | $B+C$ | a | 2 |

Table 1: Algorithms catagorised by computational and communications requirements

Most of the important constraints appeared to be generated in areas of compression and machine vision. A number of algorithms operate on blocks of data (image data and coefficents) and their I/O requirements could be somewhat alleviated by reusing coefficents stored locally. With the notable exception of block matching for image compression, the computational bottleneck of the selected algorithms appears to be caused by the need to calculate a 2D sum over a large number of products. This process is characteristic of both correlation and convolution processing which are also fundamental to the vast number of low level machine vision algorithms. Although some algorithms operate on bit planes of image data a great many (and often the most computationally demanding) operate on multi-bit grey level data. Even when each computational unit for such operations are restricted to image in image out processing, the data inevitably undergoes bit growth as the result of each process (for instance multiplication doubles the number of bits). In general both the size of the image region and the number of bits required for input, computation and output vary according to the algorithm.

This simple algorithm analysis is sufficient to demonstrate that the I/O requirements varies widely and a general purpose processor will need to be able to deal with all of these. Most image processing algorithms are fundamentally parallel, a good processing architecture will be one that can exploit this parallelism and the data redundancy inherrant in region based processing.

We can summarise the results of the algorithm analysis by saying that it indicates that a general purpose image processor should have the following features:

- High input/output bandwidth - to deal with I/O intensive operation
- An image caching strategy to allow some form of data reuse.
- A large onchip cache for coefficient data.
- Variable bit accuracy - to deal with data width growth
- Variable kernel size - for different applications
- The ability to perform a wide variety of mathematical functions - it should be especially efficient at multiplication and 2D accumulation.

A VLSI chip, the DIP (Digital Image Processor) chip, has been designed with a specification based on these requirements. From a practical standpoint the requirements of I/O intensive algorithms such as $C$ (large on-chip coefficient caches) and computation intensive processing category 3 (large numbers of processors) seem to place conflicting requirements on the use of silicon resources for a practical device. An attempt to resolve both of these difficulties in a single architecture would result in a device unsuited to both applications. For this reason the decision was taken to concentrate on the convolution and correlation functions which form the larger proportion of the selected algorithms and not to include a large coefficient store on the chip.

# 3 The DIP chip

## 3.1 Overview

This device (Figure 1) consists of a 16 bit RISC processor and a programmable 2-D vector accelerator based around the idea of SIMD (Single Instruction Multiple Data) array processing. The RISC processor and array processor are connected to allow data to be passed freely between the two sub-systems and so enable data dependent control. The RISC processor adheres very closely to the original RISC philosophy and has separate instruction, and data memories. The array, which is the main computation engine, acts as a slave to the RISC and is designed to accelerate algorithms of the correlation/convolution type. These consist of performing a local computation and then summing the results across a 2-D window as follows:

1. The array sums downward to perform a 1-D accumulation pass.
2. These partial results are fed to the RISC which carries out the final summation.

Thus the RISC has the task of performing non-parallelisable calculations and replaces the tree adder in conventional SIMD approaches. It also controls all the functions of the chip including the control of all external input/output (I/O) operations.

The array has its own instruction memory and is connected to three 16 bit wide image transfer buses for on-chip data communication.

## 3.2 The array processor

To meet the specific requirements of image processing the SIMD concept has been extended. The design employs a 16x16 array of processing elements which can be partitioned into concurrently operating subarrays and pipelines.

## 3.3 The processing elements

Each processing contains a 1-bit ALU capable of performing addition or subtraction and they are interconnected in a four way nearest neighbour scheme. This allows rapid imaGe and kernel movement across the array and allows the implementation of local region operations used in morphological operations and cellular automata. Each processing element controls a total of 160 bits of RAM, organised as two banks of three registers denoted A (16 bits), B (32 bits) and C (32 bits) (Figure 2). The two banks permits simultaneous read and write by reading from one while writing to the other. In normal use the A register will hold the input image, the C register the output image and the B register intermediate calculations and any spatially distributed control bits. These memories are used to access the three global image I/O buses via three corner turn line buffers.

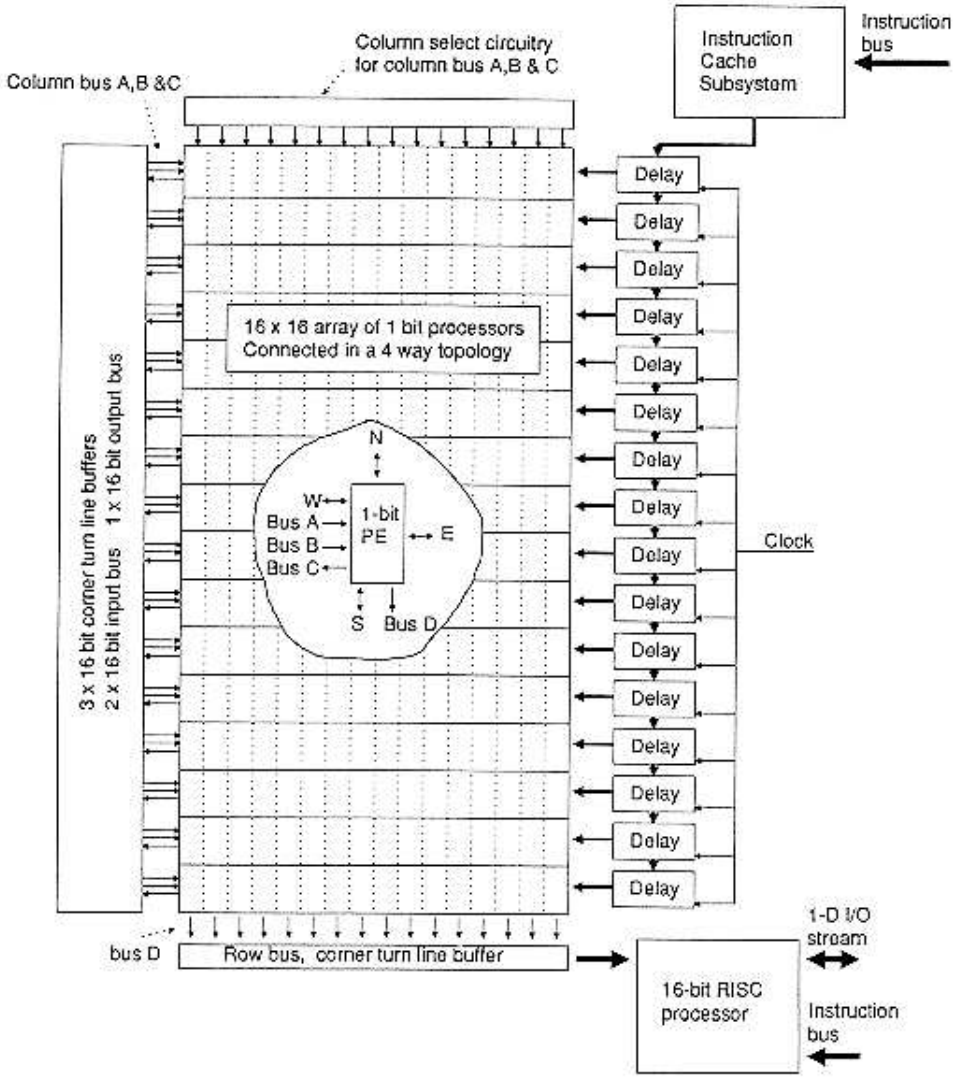


Figure 1: DIP chip

### 3.4 Array programming modes

The use of single bit processing elements has well known advantages in terms of efficient use of silicon resources and the ability to tune algorithms to the bit precision required. Many algorithms can be implemented very efficiently on traditional SIMD processors where every row receives the same instruction. This applies whenever 2-D streams are used to calculate outputs which are themselves 2-D streams, for example interframe differencing. However, a problem arises when an accumulation of a row of processors is attempted.

On traditional SIMD arrays accumulation has to be carried out a bit at a time, and row at a time, due to the single instruction nature of the array. Thus for M bit numbers being accumulated across N rows this takes over M*N clock cycles, but during accumulation across the whole array, most of the processing elements are idle. The architecture described can avoid these problems by allowing different rows to use different RAM addresses. The instruction register forms a pipelined instruction sequencer such that an instruction fed into the pipeline at the top is shifted down through each row in turn, under program control. Thus the first row can begin computation of the first bit of the result, and pass it to the second row. On the next cycle the first row can begin computation of the second bit of the result, while the second row begins the computation of the first bit of the result, and the results passed to the rows below. After N clock cycles the first bit of the result is contained in the bottom row of processors from where it can then be fed to the RISC for final summation. This multi-pipeline mode is more useful when 2-D streams are to be converted into single results or 1-D streams, and these can be executed very quickly and efficiently. In addition, communication can be inhibited on any processing element 2D processing array by use of a flag register. This allows computation pipelines to be broken during the process of sum accumulation so that the full array can be treated as multiple local processing arrays for algorithms requiring small convolution kernels,

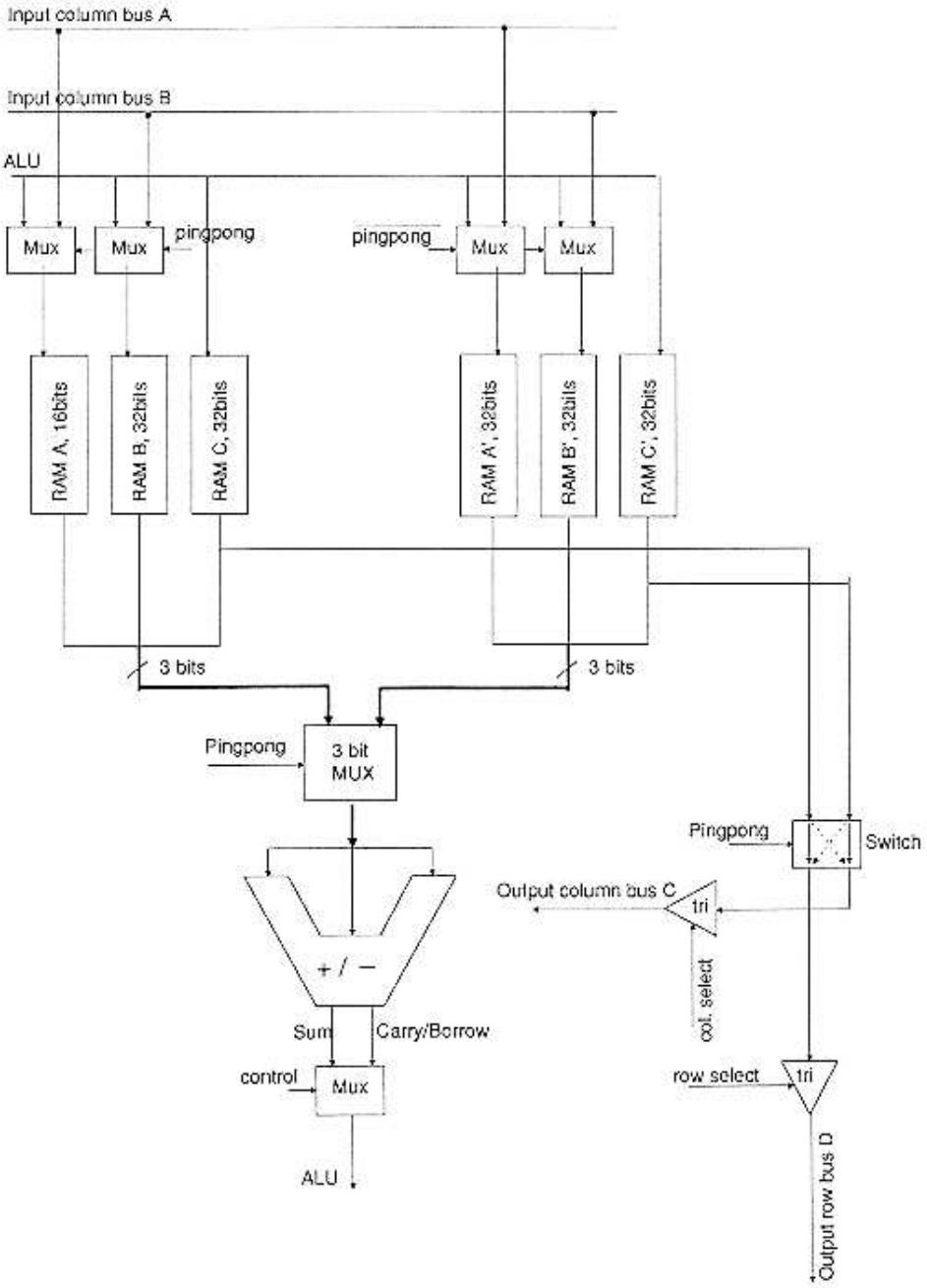


Figure 2: DIP chip processor

for example: four 8x8 discrete cosine transforms can be computed simultaneously.

In Cipher's terminology [2] the array is not a strict SIMD but rather has independent addresssing and communications and therefore differs substantially from the other SIMD devices reviewed there such as the CLIP4 and GAPP. This design therefore makes use of many of the enhancements proposed in [4] and is similar in some ways to the VIP from Sweden [3].

### 3.5 Input/Output

One of the greatest problems associated with conventional SIMD architectures is that of getting image and coefficient data onto and off the chip. The combination of DMA, on-chip state machine and double buffered registers addresses this problem directly. Three global buses are connected to the array providing two input and one output channel which can operate simultaneously, overlapping I/O with computation. A further D bus connects the array to the RISC via a separate corner turn buffer. The use of corner turn buffers also permits the operating speed

of the external circuitry to be independent of the array clock. The ability to activate individual columns under programme control also reduces the I/O overhead.

Array programmes mostly operate on M-bit arithmetic and therefore contain very simple loops of length M. Encoding these as single instruction again reduces the required I/O bandwidth and allows slow external memory to be used.

### 3.6 Physical

The device was constructed on 1 micron CMOS and contains a total of 850,000 transistors. The RISC was designed using standard cell (250,000 transistors) with full custom for the array (600,000 transitors). The array is clocked at 40MHz while the RISC and external interfaces run at 20MHz. The die shown in figure 1 is 14mm by 12mm. Further details are available in [5].

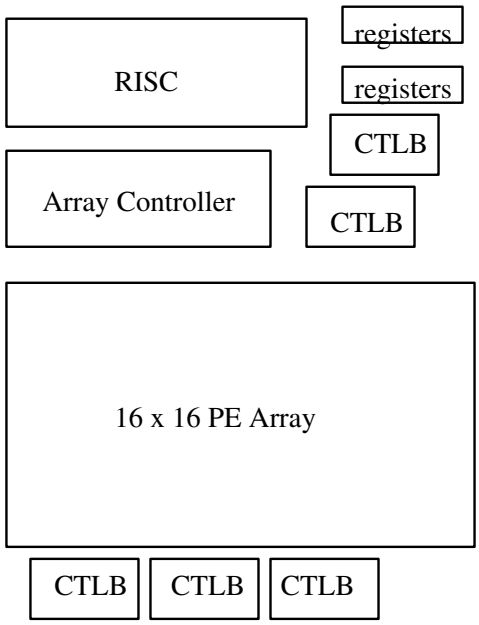


Figure 3: DIP chip floor plan

## 4 System level operation

The device was developed as part of a project on multimedia communications (AMICS) where it is intended to provide the basis of an engine capable of rapid image coding, decoding and transcoding at rates far beyond that possible on a conventional workstations.

The data processing requirements for compression schemes such as H.261 are very high, typically 1.2Gbits/sec, with much higher internal bandwidths. Conventional communications systems such as buses are not able to sustain this kind of continuous and uninterrupted data throughput. On the DIP chip, this is maintained by virtue of the 3 DMA channels which control three small (4K) cache memories holding image and data blocks arranged as two banks.

To initialise a transfer, these DMA channels send out 4 user-defined words to the off-chip DMA controllers which specify transfer source and destination. A separate IO processor coordinates the transfer to/from the system framestore for one cache bank whilst the DIP chip array accesses data from the other bank. See figure 3. The data transfer rate becomes progressively faster the further away from the frame store and closer to the DIP chip. Control along this route is kept to a minimum and sent directly to the IO processor. The DMA channels are controlled by the RISC which must perform all cache management functions. The IO processor in its simplest form is a Xilinx FPGA state machine but depending on the application may be more complex.

The frequency with which the caches must be refilled depends upon the algorithms being used but greatly reduces the bandwidth requirements. Thus communications with the framestrore can be across a system bus which can be of moderate performance (eg Sbus, VME, PCbus). The bus can be shared between multiple units and also transfer

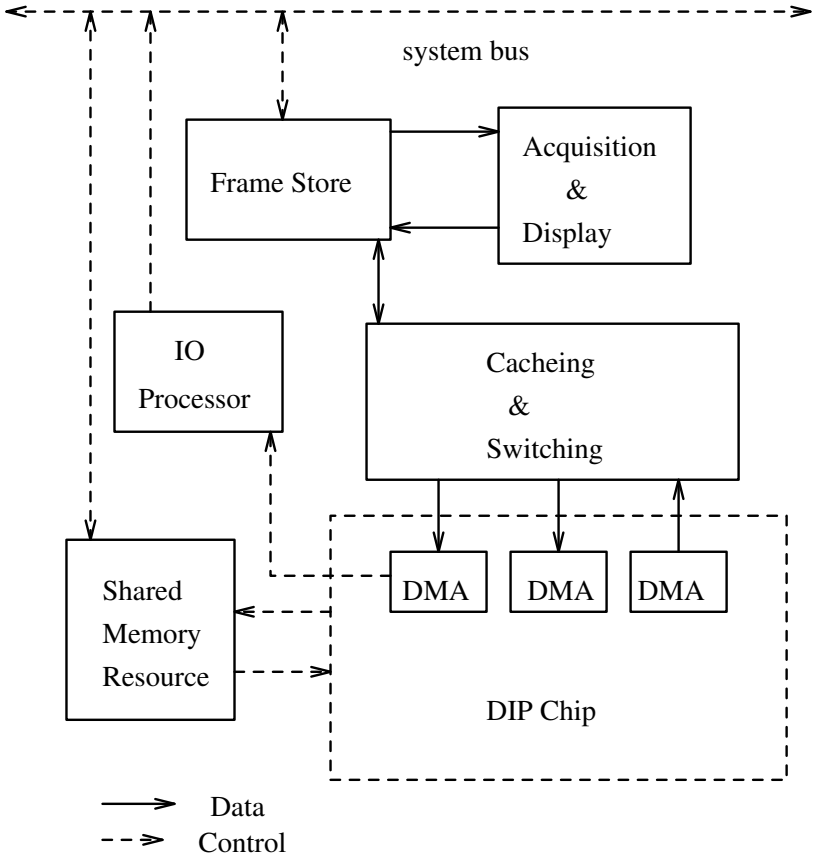


Figure 4: DIP chip system configuration

array code as well as RISC code and data. This approach decouples the the DIP chip from the rest of the system allowing low cost acquisition and display hardware to be used.

The AMICS engine is presently being implemented as a double Eurocard board within an Imaging Technology IT150/40 rack. This provides framestores with Area-of-Interest control, a fast video bus and interfaces to a networked SUN workstation. We have recently developed a compiler for the array called DISPL and are currently using it in our evaluations.

# 5 Conclusions

As the use of multi-media applications and machine vision algorithms for robot control become commonplace, a large demand for a flexible general purpose image processors can be expected to develop [3]. We have shown how a large number of these algorithms can be characterised in terms a small number of IO and data processing categories which raise specific problems for a computation engine.

The DIP chip design will be able to efficiently execute all algorithms which are directly parallelisable and which do not have excessive IO requirements. We estimate that the device will be able to efficiently carry out approximately eighty percent of the algorithms we have reviewed. A second device designed to perform the excluded tasks is currently under development and will be described elsewhere [8].

In addition, the DIP chip architecture is scalable to larger arrays. This will allow easy upgrade of the design to make use of the improved density in future chip manufacturing processes. We conclude that a modified SIMD is well suited to the requirements of 2D image based processing tasks and is a scalable approach to vector acceleration of image processing algorithms.

# Acknowledgments

This work was mostly funded by the CEC under a RACE project AMICS 2056 (Advanced Multimedia Image Communications Services). The silicon was fabricated by European Silicon Structures Ltd.

# References


[1] J.F. Canny, (1986), A Computational Approach to Edge detection, IEEE Trans. PAMI-8(6), 679-698.

[2] R. Cypher and J.L.C. Sanz, (1989), SIMD Architecture and Algorithms for Image Processing and Computer Vision, IEEE Trans. Acoustics, Speech and Sig. Proc., 37(12), 2158-2173.

[3] P-E. Danielsson, (1991), Smart Algorithms bit-serial Arrays, Progress in Image Analysis and Processing II, 473-493, Proc. 6th Intl. Conf. Image Analysis and Processing, Como, Italy, 4-6 Sept.

[4] M.J.B. Duff and T.J. Fountain, (1990), Enhancing the Two-Dimensional Mesh, Proc. 10th ICPR, Atlantic City, NJ, USA, 16-21 June.

[5] S. Evans, N.A. Thacker, R.B. Yates and P.A. Ivey, (1993), A Massively Parallel Vector Processor for Image Communications, Proc. IEEE Image Com. 93, Bordeaux, 303-308.

[6] S.J. Evans, N.A. Thacker, R.B. Yates and P.A. Ivey, (1993), A Survey of Image Coding Devices, Dept E&EE-Electronic Systems Group Memo 93/2.

[7] R.A. Lane, N.A. Thacker and N.L. Seed, (1993), Stretch-correlation as a realtime alternative to feature-based stereo matching algorithms, Image and Vision Computing, 12,4, pp 203-212, 1994.

[8] R.A.Lane, N.A.Thacker and P.A.Ivey, 'A Correlation Chip for Stereo Vision.' to be presented at BMVC 94, York, September, 1994.

[9] P.A.Riocreux, N.A.Thacker and R.B.Yates, 'An Analysis of Pairwise Geometric Histograms for View-Based Object Recognition.' to be presented at BMVC 94, York, September, 1994.