Proceedings of The Institute of Acoustics

A 24 BIT 10 MIPS DIGITAL SIGNAL PROCESSOR

David G. Malham.

Electronic Music Studio, University of York.

INTRODUCTION

Work over the past three years at York University's Electronic Music Studio has culminated in the developement of a 24 bit digital signal processor capable of executing 10 million instructions per second. This module will form the basis of our intended conversion over the next few years, from a very 'classical' analog/tape based studio to a largely digital environment for the realisation of electro-acoustic music.

PROJECT HISTORY

At the beginning of the 1980's we were considering two possible ways in which to accomplish the change-over to digital technology that we saw as essential if we are to maintain York's position as one of Britain's foremost university based electronic music studios.

Firstly we could obtain a powerful minicomputer and use it as a non-real time compositional engine, the system under consideration being a PDP11 running Music 11. The other and preferred option was some form of real time digital synthesizer with, if possible, the option of inputting signals from external sources. Whilst the first course had much to recommend it in terms of power and facilities its' inability to produce sounds immediately on receipt of a stimulus weighed heavily against it. The presence in the market place of the American made DMX1000 digital signal processor therefore persuaded us to put in a budget request for one of these systems, since this would provide us with reasonably powerful real time analysis/synthesis capabilities.

Somewhat to our surprise, we got most of the money we asked for but unfortunately in the meantime the pound-dollar exchange rate had deteriorated to such an extent that we decided to have a re-think. This resulted in a decision to design and build a digital signal processor ourselves which, given the advances in technology since the DMX1000 was designed, would have several times the power but would cost considerably less. From that point it took me around two and a half years working on and off to get to the point where, early in 1985, we got the first sounds out of the device. I propose to spend most of the rest of the paper looking at the reasons for the various design choices that were made during that period and how some of them were implemented.

A 24 BIT 10 MIPS DIGITAL SIGNAL PROCESSOR.

DESIGN CHOICES

General Requirements.
The brief from the musicians involved in defining project aims was for me to provide a flexible, real time system which was capable of synthesising sounds of sufficient complexity and quality as to be indistinguishable from 'naturally' produced sound. It should provide all the normal filtering and equalisation functions available in analog form, together with as many others as possible. Naturally (they said) it should also provide for the possibility of digitally recording and editing sounds as well as being capable of analysing externally generated sounds. From an engineering point of view, I wanted to end up with some kind of module which would form a basic building block for the all-digital studio which we were proposing to have up and running by the end of the 1980's. This lead me to reject one early idea which was to design a whole series of cheap, relatively simple cards, each devoted to one particular function. As I indicated earlier, the eventual project specification was for a general purpose digital signal processing which was philisophically similar to the DMX1000 but much enhanced in line with the advances in integrated circuit technology and with a parts cost of under £2000.

DMX1000
This is a high speed 16 bit computer based on 2901 4-bit slice computing elements. Instructions are coded into a 32 bit microcode word by an external host computer. A fair number of operations may take place in parallel and all main instructions execute in 200 nS. Somewhat unusually, there are no branch instructions whatsoever, only a form of conditional 'no-op'. This is because the system sampling rate is determined by the number of instructions in a program loop, N, and the instruction rate, 5 MHz [1]

$$\text{SAMPLING RATE} = \frac{5000}{N} \text{ KHz} \qquad (1)$$

which means that any change in loop length, such as could occur as a result of a conditional branch, would result in sampling rate jitter. It has two blocks of memory, a 'data' memory and a slower 'delay' memory both limited to a theoretical 64 kilowords of capacity. There is a direct access channel to the data memory fromthe external host computer, up to four each DAC or output and ADC or input channels. The rather small microprogram memory which controls the machine can have at most two locations updated in each sample loop.

The York DSP.
The most serious limitation of the DMX1000 was felt to be its' word length. As this was only 16 bits, there were problems when programming table look-up oscillators [2] . We intend to work mainly at a sampling rate of 44.1KHz, as this will make interfacing with cheap digital recorders such as the Sony PCMF1 relatively easy. Since the formula for the frequency resolution of a table look-up oscillator is

A 24 BIT 10 MIPS DIGITAL SIGNAL PROCESSOR.

$$Fres = \frac{Fs}{2^{n-2}} \quad (2)$$

Where F_S is the Sampling rate and n is the number of bits in the phase angle register then a 16 bit machine has a frequency resolution of 0·673 Hz at this sampling rate. Some experiments I did in the early 1970's at University College, Cardiff indicated that even though it was impossible to discriminate between isolated sine tones 0·5 Hz apart at 1KHz, a change of half a Hertz in a continuous tone around this frequency was easily audible. Work by Rakouski [3] indicated a capacity in some listeners for discerning a change of as little as 0·03 Hz at 160Hz, so we can see that 16 bits of phase angle resolution is totally inadequate if we hope to get smooth glissandi at a good sampling rate. The only way round this with 16 bit arcitectures is to use double precision arithmetic which immediately at least halves the effective instruction rate - in the case of the DMX 1000 from 5 MIPS (Million Instructions Per Second) down to 2.5 MIPS. This was unacceptable and indicated the need for a greater word length.

Moreover, I had to consider the effects of mixing or otherwise modifying signals. It can be seen from the formula for the dynamic range of a digitally encoded signal [4] .

$$SNR = 6{\cdot}02n + 1{\cdot}76 \text{ dB} \quad (3)$$

that a 16 bit signal has a dynamic range of around 98 dB at best. This is just adequate for most audio work unless one has to mix in another signal. In order to avoid the possibility of overflow when mixing two signals then both signals must be reduced in range by 1 bit, from 16 to 15 bits in the case of a 16 bit system. This can be generalised to

$$R_S = n - B \quad (4)$$

if $S = 2^B$, R_S is the allowable resolution of each signal, in bits, n is the word size of the system and s is the number of signals to be mixed which simplifies to

$$R_S = n - \frac{\log(S)}{\cdot 301} \quad (5)$$

In a multi channel digital mixing configuration the maximum effective dynamic range (SNR_E) of any input signal is, from [3] and [5]

$$SNR_E = 6{\cdot}02\ R_S + 1{\cdot}76 \text{ dB} \quad (6)$$

if system overflow is to be avoided. Similarly, when passing signals through high Q filters the presence of gain in the filter must be compensated for by a lowering of signal level if clipping is not to occur. This again dictates the need for a greater word length to avoid a reduction in dynamic range caused by bits 'falling off' the end of the signal word, even though this can be ameliorated by the judicious addition of small amounts of white noise or dither [5] .

Weighing up all these factors, and the economics of the various options, a desicion was made to go for a 24 bit word length, giving a frequency resolution of 0.00263 Hz at 44.1 KHz sampling rate and a dynamic range of 146 dB for a single signal or 96 dB for each signal in a 256 component mix.

A 24 BIT 10 MIPS DIGITAL SIGNAL PROCESSOR.

I then had to consider the size of word to be used in the micro-coded program memory. The number of bits in this is directly influenced by how many different operations you wish to be able to code to happen in parallel. From the base set of 32 in the DMX1000 design, the number of bits rapidly increased as a result of 'creeping featuritism' first to 40 bits then on upwards. I eventually called a halt at 64 bit since anymore would have needed another 96 way connector on each board, to add to the two already used to carry the main bus signals.

In the micro code there is direct access to four DAC and four ADC channels. This number reflects my bias towards anything Ambisonic but any number of extra channels can be added by using memory mapping within main memory addressing space. This totals some 2^{24} or nearly 17 million 24 bit word locations, which can also be used to add RAM's for delays, reverberation etc. or ROM's for permanent waveshape tables and other functions. There is also access to a PSRB white noise generator, two different multiplier locations and a small but fast cache memory on the CPU board. For most purposes this acts almost like an extension of the 2901 bit slice's register set.

A decision was made early on not to go for any of the more esoteric technologies such as ECL or semi-custom gate arrays since this was undesirable for economic reasons and because they would make the design less reproducible. However, by using largely the FAST (74F) Advanced Low-power Schottky (74ALS) and Advanced Schottky (74AS) families of TTL logic as well as the latest version of the 2901 bit slice component - the 2901C I was able to achieve a processing rate of 10 MIPS without too much trouble.

THE DESIGN PROCESS

Once the basic design choices had been made the main thrust of the design process was aimed towards getting the maximum processing speed out of the technology. The main weapon in the design armoury was, as is usual for most high speed computers, the use of pipe-lining allied with the extensive use of parallelism.

Pipelining.

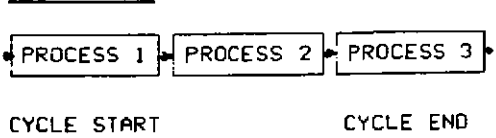


Fig (1)a

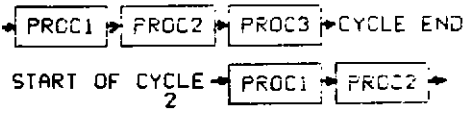


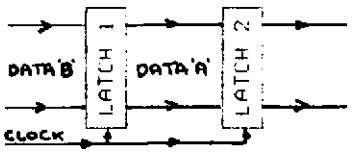
Fig (1)b

Figure 1(a) is a simplified representation of what happens to data during a single machine cycle. At the start of the cycle, process 1 would typically be the latching of data and instructions for that cycle, process 2 would be some arithmetically or logical operation on that data and process 3 the latching of the results of process 2.

A 24 BIT 10 MIPS DIGITAL SIGNAL PROCESSOR.

If each process takes 50 nano seconds (nS) then total machine cycle time is 150 nS and the instruction rate is 6·666 MIPS. If, however, we overlap the process 1 of one machine cycle with the process 3 of the previous machine cycle, as in fig 1(b) we can claw back 50 nS of the 150 and increase the instruction rate to 10 MIPS. This order of improvement is not achievedwithout some difficulty. Even in a relatively simple machine such as this one the number of different combinations of source and destination involved in any one instruction runs into the thousands. Naturally many of these combinations are degenerate in that they are essentially duplicates of others but they still have to be checked for possible conflicts caused by the use of pipelining. For instance, it might be the case that for some particular instruction, the process 3 of the previous cycle must be completed before it can start its' process 1. Overcoming this sort of conflict is probably the most tedious and time consuming part of the design process for this kind of device. Fortunately, the use of the newer generations of TTL logic mentioned above has eased this sort of problem somewhat since their manufacturers now specify not only the minimum speed of devices but also their maximum speed. If it may seem strange that one should worry about devices being too fast in a design where special speed is of the essence, consider Fig (2)

Fig.(2)



Here we see a pair of edge triggered latches in a pipeline configuration, What is supposed to happen is that data 'A' out of latch (1) is supposed to be clocked into latch (2) while simultaneously data 'B' is being clocked into latch (1). If, however, latch (1) is so fast and latch (2) is so slow that data 'B' propagates through to the outputs of (1) before (2) finishes the process of latching data 'A' considerable confusion could result. It is essential to achieving a stable design to know both the maximum and minimum propagation times of all critical devices. It proved possible in the present design to achieve 10 MIPS reliably and with a resonable safety margin without using any esoteric devices.

Parallelism.

Apart from the raw machine speed, system throughput is increased by having as many operations occuring simultaneously as possible. If we consider the case of a microprocessor based system where we want to send the results of an operation to one or more DAC's and to a delay memory for later use and have it level adjusted (multiplied) and used in another operation. Each of these would need at least one instruction to accomplish in a typical microprocessor system but by extensive use of parallelism they can all be done

A 24 BIT 10 MIPS DIGITAL SIGNAL PROCESSOR.

simultaneously resulting in an enormous improvement in system throughput. In the present design, data from an operation can be sent to up to a dozen different destinations simultaneously and at the same time the next operation can get under way.

Sampling rates, Branching and the Microcode Memory.

Rather than use a program loop length method of determining sampling rate, I have implemented a form of interupt driven system. This means that each DSP can be locked to an overall sampling rate clock which would govern the whole studio. As a bonus this meant that I could also implement a rudimentary branching capability which, used with care, should not interfere with system operation. This, of course, considerably eases the problem of programming compressors, expanders, noise gates and other signal level dependent devices. The microcode program memory is rather large, at 8192 words of 64 bits each and is of a fairly sophisticated interleaved design which allows the external host or control computer to update one program whilst another is simultaneously being executed by the DSP. Indeed the size is such that at least 37 separate programs can be held in memory for a 44·1 KHz sampling rate.

Testing.

The problems involved in testing such devices are, of course, well known. My initial approach to debugging the logic was to build a test rig where the action of the microcode memory was simulated in a static or semistatic mode by a set of switches. This was eventually replaced by a computer program written by a third year Music student, Rob Wills, which allowed toggling of indivual bits within microcode memory to be done under keyboard control. It also provided a way of inputting and running simple programs to do full speed dynamic tests. Together with a 48 channel, 50 MHz logic analyser and a variable rate machine cycle clock this enabled me to solve the remaining problems and to run maximum instruction rate tests under severe conditions such as high and varying temperatures.

THE FUTURE

We now have an operational digital signal processing component on which to base the development of the York studio. Although we do have an assembler (also written by Rob Wills) running on our DEC 10 mainframe computer we have yet to port it to the Atari 520 ST personal computers which we intend to use as host computers for the DSP modules. Indeed the main work now seems to be in software development with an enhanced version of the assembler planned and higher level languages under consideration. Hardware development will still continue, particularly in respect to intelligent peripherals and bulk storage systems. The possiblity of implementing sections of the system in silicon using gate arrays or semi-custom logic is also under consideration. All in all, it is a very exciting development and I would like to thank Richard Orton, the head of the electronic music studio at York University for his support and encouragment and also Rob Wills for the assembler, the microcode memory program and a very useful wiring

A 24 BIT 10 MIPS DIGITAL SIGNAL PROCESSOR.

list processing program all of which he wrote in has spare time while taking his music degree.

REFERENCES

[1] Digital Music Systems Inc., DMX1000 Programming Manual (1979)
[2] John Snell; 'Design of a Digital Oscillator which will Generate up to 256 Low Distortion Sine Waves in Real Time' Computer Music Journal, Vol.1. No. 2 pp4-25 (1977)
[3] A. Rakowski; 'Pitch Discrimination at the Threshold of Hearing' Proceedings of the Seventh International Congress on Acoustics. Vol. 3. pp 373-376 (1971)
[4] B. Blesser: 'Digitization of Audio, A Comprehensive Examination of Theory, Implementation and Current Pratice' JAES Vol. 26. No. 10 page 743 (1978)
[5] J. Vanderkooy and S.P. Lipshitz 'Resolution Below the the Least Significant Bit in Digital Systems with Dither' JAES Vol. 32. No. 3 pp 106-114 (1984)