Nuclear Instruments and Methods in Physics Research A307 (1991) 110–115
North-Holland

Readout control for high luminosity accelerators

R. Belusevic and G. Nixon

Department of Physics and Astronomy, University College London, London WC1E 6BT, UK

Received 15 April 1991

In this article we discuss some aspects of data acquisition at high luminosities and offer a set of design principles concerning readout control electronics and related software. As an example we include a brief description of a data transfer and processing system for future hadron colliders, featuring a transputer-based crate controller and a set of readout cards. This is a simplified and more efficient version of our design recently published in Nuclear Instruments and Methods. [A295 (1991) 391].

1. Introduction and general discussion

By the turn of this century large hadron colliders are expected to become the principal tools for high energy particle physics. These machines will provide a new energy regime and a significant increase in luminosity compared to present-day accelerators. The collision energy will be boosted by an order of magnitude while the detectors will have to handle up to 100 million collisions every second (three orders of magnitude greater than the highest event rates at present) in order to study processes with very small cross sections. High collision and low physics rates will require extraordinarily selective trigger procedures. Multiple collisions occurring in every bunch crossing and the predicted physics signatures will require detectors capable of extracting high-level information in the shortest possible time. Since the number of readout channels is also expected to increase by an order of magnitude (leading to an overall increase in data volume by a factor of ten thousand), the sheer size of the detectors will make signal synchronization exceedingly difficult.

It is clear that the existing bus-based systems are not adequate for the tasks facing data acquisition electronics in such an environment. Although the new concepts now under development (e.g. Futurebus) should achieve comparatively larger data transfer rates, the inherent complexity of these designs (in both hardware and operating software) is bound to reduce their flexibility and increase real-time overheads. In this context it is worth pointing out that since the usefulness of future detectors will depend on their ability to extract high-level information in real time, a large fraction (up to 60%) of detector budgets will have to be spent on data acquisition electronics. With this in mind we have designed two versatile readout controllers capable of the highest data transfer rates and processing power currently possible, while keeping the complexity of hardware and operating software to a minimum. We have also demonstrated the feasibility of data compaction and trigger processing within these systems by developing appropriate software (see ref. [1] for a detailed description of the earlier design).

In our approach we were guided by what we believe to be the most important requirements a future readout system must satisfy:

- data transfer speed of at least 100 Mbytes/s, including real-time overheads;
- processing power to match the above;
- concurrency with respect to data transfer and processing to increase the overall system throughput;
- flexibility (independence of detector and crate type);
- simplicity of hardware and operating software, which leads to increased reliability and reduces heat dissipation and cost;
- data compaction and trigger processing before readout by the crate controller.

To achieve the above goals we suggest the following hardware and software solutions, which amount to cutting the Gordian knot of self-perpetuating complexity of bus-based systems:

- reduce interfacing logic and eliminate bus arbitration by using:
  a) single controlling microprocessor with an advanced instruction set and simple operating software which supports multitasking,
  b) dual-ported memories throughout the system,
  c) LSI circuits (e.g. versatile microsequencers and PALs);
- reduce real-time overheads by means of specially designed hardware;
- provide easy access to data for the most powerful (vector) microprocessors;
- form a pipeline of reduced data (using digital signal

processors or "superscalar" microcomputers for data parameterisation and compaction) on the readout cards, which results in a better utilisation of board space and avoids the transfer of unwanted data;
- use fast (1 Gbit/s) intercrate serial links.

It is important to note that most of what was said above is also valid for future B-factories (the one now planned at KEK in Japan, for example, is expected to be completed a couple of years before LHC). These machines will operate at very high luminosities (up to 10^{34} cm^{-2} s^{-1}), and although the electromagnetic cross sections are orders of magnitude smaller than the hadronic, the short beam intersection times (a few ns) and sophisticated pattern recognition algorithms will require fast and efficient data transfer systems and a considerable amount of on-line data processing. We believe that the new generation of "superscalar" microprocessors (the Intel i870, for example) will be needed in the future to replace the presently available chips (the i860, SPARC, MIPS, Motorola 88000, etc.).

2. A fast transputer-based data transfer system

The system described in this article is a result of our attempt to pursue the above strategy as far as possible. We have implemented three modifications with respect to our original design. The crate controller (CC) now contains two address generators: one for the local dual port memory (DPM) and one for the DPMs on the readout cards, thereby reducing the amount of hardware per crate. As before, the address generators enable direct memory access (DMA) operations, running at the maximium data transfer speed of 80 Mbytes/s. Furthermore, we have replaced binary counters, which originally served as address generators, by versatile single-chip sequencers, in order the keep the interfacing logic on the crate controller as simple as possible and to increase flexibility of the data transfer hardware. Most importantly, fast intercrate data transfers can now be accomplished by connecting the DPMs *directly* to a 1 Gbit/s serial link (or transceivers, whose outputs may drive a set of twisted pairs), as shown in fig. 1. Each readout card (RC) contains a digital signal processor (DSP) for data compaction and trigger preprocessing. The reduced data, pipelined inside dual-ported memories, can be accessed by the transputer and transferred to a common DPM on the CC card for further processing by a powerful i860 microcomputer. The use of a transputer as the sole controlling processor, together with DPMs, renders bus arbitration unnecessary and leads to very simple interfacing logic and operating software. The latter is written in the Occam language, which was specially developed for programming concurrent systems based on transputers. The four high speed serial links of the transputer greatly facilitate downloading of programs and intercrate communications. This scheme also enables easy bootstrapping of the embedded microprocessors, without the need for EPROMs.

In what follows we shall describe only the data transfer hardware and its operation. The controlling and data processing software (also the interfacing of the embedded microprocessors) is the same as that for the system described in ref. [1], and therefore will not be discussed here.

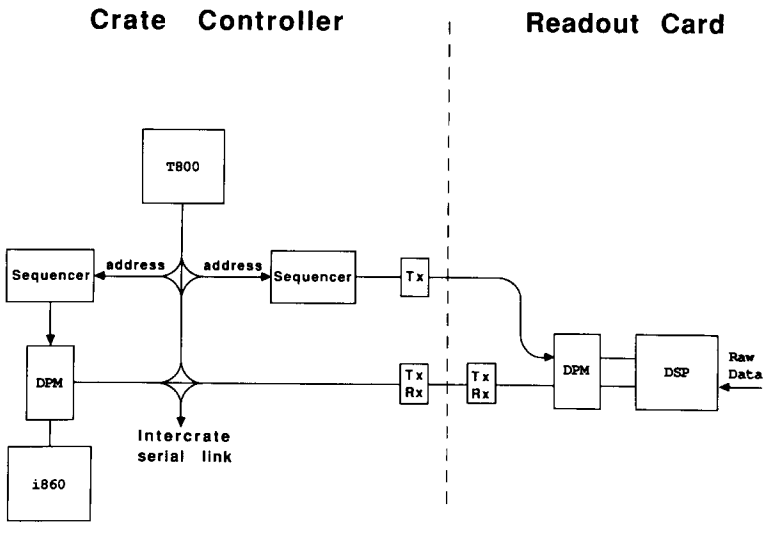

Fig. 1. Block diagram showing system's main components and data flow within a crate.

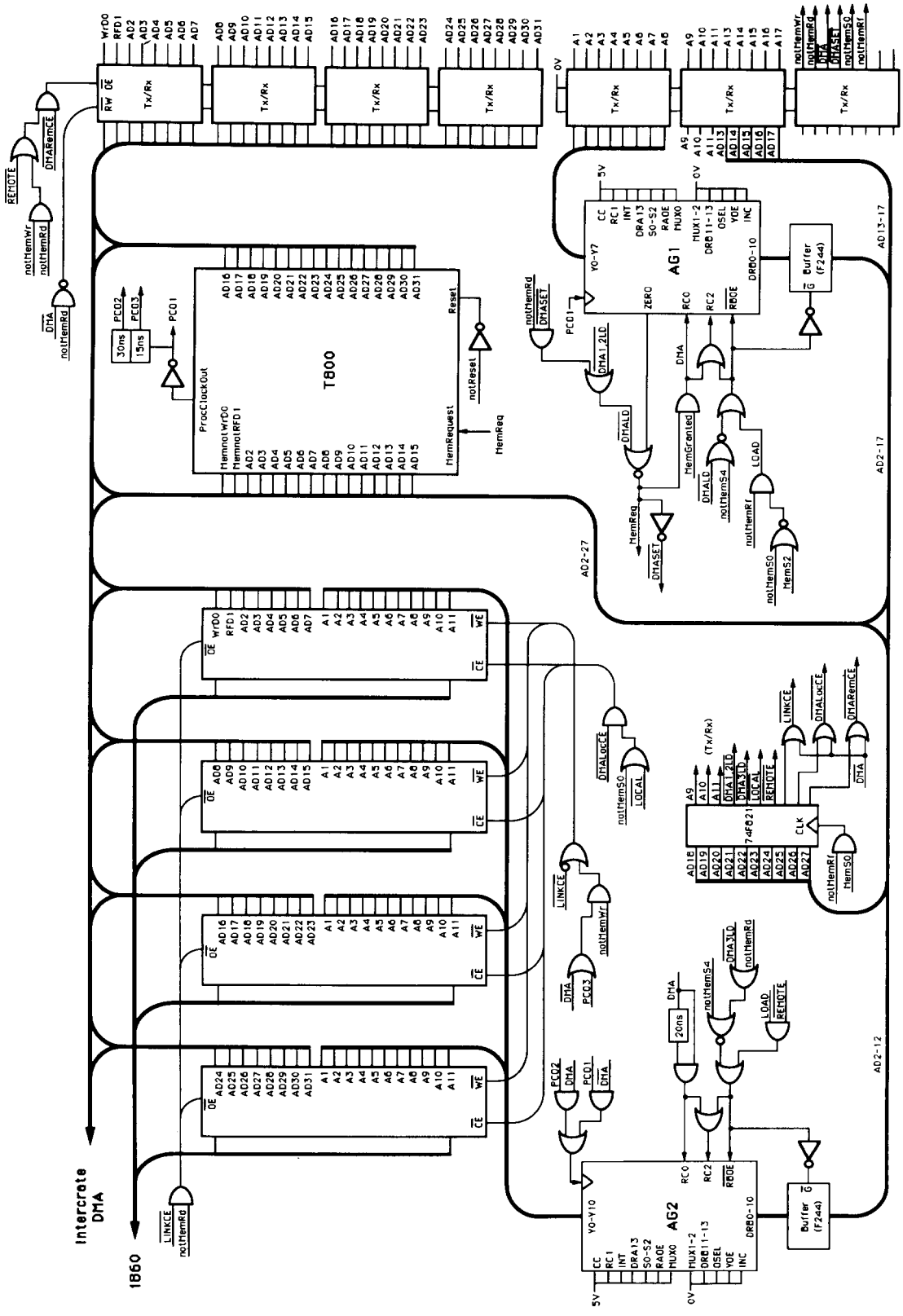

Fig. 2. Crate controller: circuit diagram of the data transfer hardware.

2.1. Crate controller

The circuit diagram of the readout controller is shown in fig. 2. The controlling processor is an Inmos T800 transputer, a 20 MHz, 32-bit microcomputer with 4 Kbytes of on-chip RAM, a multiplexed data and address bus and four serial links (maximum link speed 20 Mbits/s) for interprocessor communications. The 32-bit wide memory interface of the transputer, which can access an address space of 4 Gbytes, can be configured both internally and externally. For this design we have chosen the internal configuration achieved by connecting the MemConfig pin of the transputer to the address/data pin AD10. Since this configuration generates dynamic RAM refresh cycles, the ordinary external accesses are qualified by the notMemRf signal (see fig. 2).

The Inmos transputer is an advanced RISC microcomputer which supports multitasking and parallel processing in the instruction set, eliminating the need for an operating system (for our type of application). The controlling software has been written in the Occam language, which was specially developed for programming concurrent systems based on transputers.

Two microsequencers, Texas Instruments type 74AS-890, on the crate controller are used to latch transputer-generated addresses, and also as address generators for high speed data transfers. One microsequencer serves the memories on the readout cards, and the other the local memory on the crate controller. The memory used throughout the system is a 32-bit, 8 Kbyte static DPM (4 times VT7132A; access time 30 ns).

The microsequencer features two configurable registers, of which we use only one. A three-bit input (RC0–2) is decoded internally to provide load, hold, and decrement functions for the register. The register of one microsequencer (AG1) is connected, via the backplane, to the address lines of the readout cards, and that of the other (AG2) to the local address lines. Since the register input and output share the same internal bus, a unidirectional buffer serves to disconnect the latter from the external data bus when appropriate. Any type of TTL-driven backplane (VME-J1 in this case), properly terminated, can be used. As in our previous design, the line drivers are fast, high current (160 mA) Signetics 74F30245 transceivers.

The transputer-generated addresses consist of three parts. The least significant 11 bits AD2–AD12 (8 bits in the case of the readout memories) form the internal DPM address. The second part is the board address (bits AD13–AD17), including, when appropriate, the readout memory sector address (bits AD18–AD20). The third part determines whether the current access will be a random access cycle or a DMA setup cycle (bits AD21–AD27; see fig. 2).

To achieve fast transfers of a crateful of data throughout the system, we have developed hardware for three types of DMA operations:

- DMA1: from readout to controller memory;
- DMA2: from readout memory to serial link;
- DMA3: from controller memory to serial link.

In DMA1, the register in the microsequencer AG2 is first loaded with the start address of the block of data which will accumulate in the local memory. (The latching of a transputer-generated address is achieved, for this particular transputer memory interface configuration, by generating a load pulse formed from the strobes notMemS0 and MemS2.) This need only be done once at the start of a series of data transfers. It is assumed that data corresponding to a single event will be transferred from a set of blocks in the readout memories to form a single block in the controller memory. It is further assumed that data blocks in the readout memories lie on sector boundaries.

The next step is to load the microsequencer AG1 with the start address of the first block to be transferred. This is accomplished in a single processor cycle, as described below. The transputer generates a predetermined address in the readout memory, which serves as a pointer to the actual start address (SA) of the data block (the DSP on the readout card is responsible for loading the start address into the pointer). On latching the pointer address, the start address appears as data and is loaded into the sequencer register during the data period of the cycle (the transputer also acquires the start address in its internal register for use if needed). This loading of data is possible only for the DMA setup cycles, which are qualified by the latched address bits AD21 ($\overline{\text{DM A1, 2LD}}$) or AD22 ($\overline{\text{DM A3LD}}$), and takes place during a period defined by the strobe notMemS4.

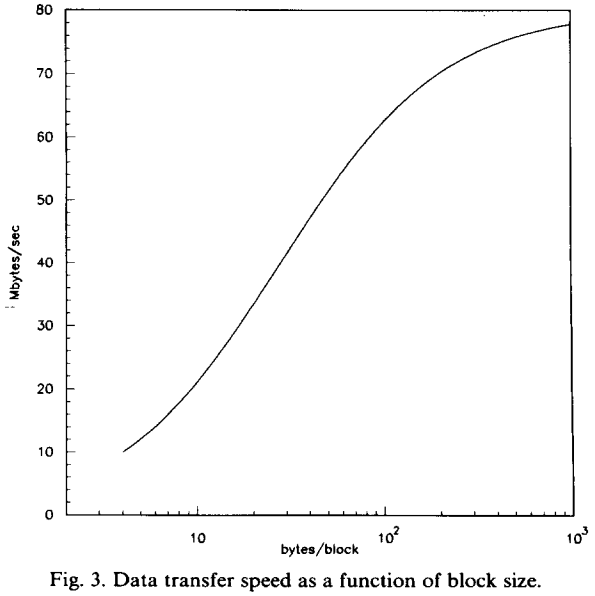

Fig. 3. Data transfer speed as a function of block size.

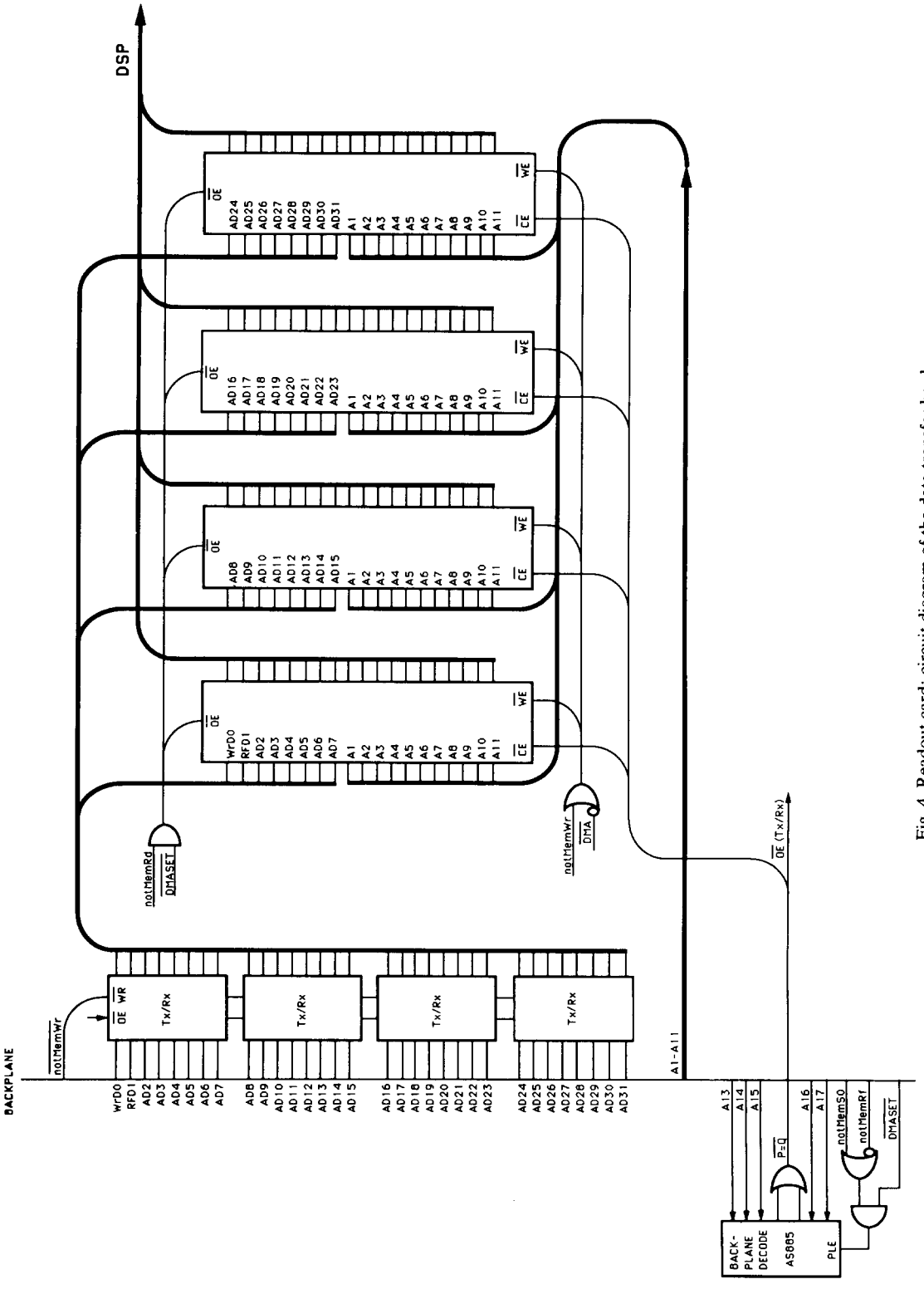

Fig. 4. Readout card: circuit diagram of the data transfer hardware.

At the end of the cycle the transputer, which has received a MemRequest signal from the hardware, relinquishes the bus and issues MemGranted, setting up a hardware DMA status and causing both AG1 and AG2 registers to decrement on each clock rising edge (20 MHz). The clock used is an inverted ProcClockOut signal (PCO), suitably delayed in the case of AG2. An output ZERO signal generated by AG1 stops the DMA process when the sector boundary is reached. (Since ZERO is set as soon as the register contains a digital one, i.e. effectively two decrements short of zero, the start (topmost) address should correspond to the value $N + 2$, instead of N, where N is the block length in a particular memory sector.) Once DMA has finished, the transputer regains the control of the bus and can set up the second block transfer. In this way very efficient data transfers, with the real-time overhead consisting of a single processor cycle, can be achieved (see fig. 3).

The second type of DMA transfer can be carried out in a similar way, except that the data is not sent to the local memory but directly to an intercrate link (enabled by $\overline{\text{LINKCE}}$ and clocked using a suitably delayed PCO signal). We intend that this should be an FDDI standard serial link, such as the single-chip transmitter/receiver "HotRod" by Gazelle, when it becomes commercially available. This chip could accept 32-bit data at the transputer clock speed of 20 MHz, which, with added protocols, would be compatible with a serial transmission rate of 800 Mbits/s. A second chip situated on another crate would receive the data and, with the aid of similar circuitry to the above, deliver it to a dual port memory. The system can use either a coaxial or an optical link. This would be preferable to a parallel link on account of the much reduced volume of intercrate wiring, always a bulky and vulnerable part of any installation. Although a few high speed serial link chipsets are already commercially available (e.g. those by Vitesse or GigaBit Logic), their cost and the number of components involved makes the "HotRod" by far the most favourable solution.

The third type of data transfer is from the readout controller to another crate via the serial link. In this case the start address, supplied by the i860 microprocessor on the crate controller, is obtained from the local memory and latched into the AG1 register in the manner described above. Note that in each of the three cases AG1 is the master sequencer, which receives the start address of the data block to be transferred.

We envisage that data buffering be provided on the readout cards, resulting in an efficient utilisation of board space. We also propose that data compaction and trigger preprocessing be performed on the readout cards using DSPs, as described in ref. [1]. Special trigger information can then be transferred to the crate controller for further processing, while the bulk of the data awaits the trigger decision. This scheme minimises the amount of time wasted in transferring unwanted data.

2.2. Readout cards

The circuit diagram of the relevant part of the readout card is in fig. 4. We have shown the dual port memory and the means of access to it from the backplane. A set of four transceivers drives the data lines from the readout card to the crate controller. The board address is decoded on the backplane from the address bits A13–A17 and latched to form a board select signal using a single chip (74AS885). This signal is used to enable both the readout memory and the transceivers. The memory address lines A1–A11 are connected directly to the backplane. The other side of the dual port memory is connected to a DSP (see fig. 5 of ref. [1]) and to the outputs of readout pipelines. As can be seen by comparison with the ref. [1] design, the data transfer hardware on the readout cards (of which there can be up to 20 per crate) has been significantly reduced by placing the RC address generator on the crate controller.

Acknowledgement

We are grateful to Dr. J. Lane for his invaluable help with software development throughout the project and his interest in our work.

Reference

[1] R. Belusevic, G. Nixon and D. Shaw, Nucl. Instr. and Meth. A295 (1990) 391.