The Cambridge Backbone Network
An Overview and Preliminary Performance

David J. Greaves
Olivetti Research Ltd.
University of Cambridge, Computer Laboratory
Krzysztof Zieliński
Institute of Computer Science
University of Mining & Metallurgy

January 1, 1991

1 Introduction

The CBN (Cambridge Backbone Network) is a collaborative project between Olivetti Research Limited and the University of Cambridge Computer Laboratory. The project has designed and build an experimental communication network for integrated computer data, voice and real-time video applications.

The CBN architecture can be characterized in general terms as follows:

- Backbone network 500 to 1500 Mbit/sec slotted ring topology,
- ATM and multicast modes of operation,
- 200 km of fibre – metropolitan dimensions,
- 30 km link length easily supported,
- Expected configuration typically 30 stations offering 30 Mbit/sec each,
- Direct compatibility with Cambridge Fast Ring [4].

The CBN offers an ATM LAN/Man architecture based around a source release slotted ring [1, 4]. The network operates on monomode optical fibre and is currently designed for an eventual line rate of 1000 Mbit/sec. The bandwidth of the fibre optic channel can be partitioned into a number (currently four) TDM channels. This enables stations of varying cost and bandwidth to be attached to one network, parameterised by the number of channels a station can use concurrently.

This paper gives an overview of the CBN station hardware and software design and reports preliminary performance measurements of the first implementation in terms of throughput and response time. The main purpose of these studies was to identify the potential bottlenecks in the current CBN hardware and software architecture to give some ideas for future development.

2 Backbone Network Station Hardware Design

The hardware configuration of a simple half-duplex station [1] is shown in Figure 1. The Backbone Network uses 4B5B/NRZI modulation. Each station has a local, crystal-locked transmit clock and retimes the received data using an elastic buffer. The received data from the optical fibre is passed to an ECL access chip which performs two types of function. The first concerns the transformation of the serial data to eight bits wide and provision of clock and byte synchronisation. This requires the following functions:

- Decide whether he current input voltage is a one or zero using D-type flip-flop,
- Convert from NRZI to NRZ,
- Decode 5 bit blocks into 4 bit data word or special code 'syn' used for synchronization,
- Gain bit synchronization using a slip method,
- Signal a code error on codebook violation,
- Gain frame synchronization on frame header, and
- Encode output stream into 4B5B and thence NRZI.

The second type of function provided by the access chip is at the frame level. The format of a Backbone Network frame is shown in Figure 2 .

Each frame starts with a header block. The frame then has the full/empty (F/E), monitor passed (M) and type (T) flag bits of each slot. The frame trailer contains response (R) and qualifier (Q) for each slot and a 12 bit CRC that covers all fields from F/E to Q inclusive.

The investigated function includes:

- Check CRC of received frame,
- Examine and update F/E bits to decide which slot is to be read or written,
- Generate data strobes for reading or writing the desired channel,
- Check and update response and qualifier bits of outgoing frame, and
- Write new valid CRC to outgoing frame.

The 8 bit data streams from the ECL, access chip are connected to the RAM packet buffer through the semi-custom demultiplexer devices. Their purpose is to widen the data from 8 to 32 bits.

Figure 1: Block diagram of a simple CBN station.

Header	Full	Data area for four mini-packets.	Response	CRC
	Monitor	Each contains nine 32 bit words and		
	Type	including the routing tages.	Qualifier	
(4)	$(4+4+4)$	$(4 \times 9 \times 32 = 1152\text{bits})$	$(4+4)$	(12)

Figure 2: CBN frame. It contains four CFR size slots.

3 V1S Interface

The first implementation of the CBN station uses the V1S CBN station interface. This interface provides access to the network over the VME bus and is builded around the Motorola MVME147 68030 (20 MHz) card. This interface was designed for minimum complexity, and their raw performance is simply determined by the bandwidth of their buffer RAMs.

All cells waiting to go onto the network or which have just been received share the same RAM array, which consists of four single-ported bytewide CMOS static RAMs, operating at a fixed rate 3.2 MHz. Therefore their terminal bandwidth is 100 Mb/sec, which is also the peak rate that data will be received off the CBN ring, owing to its multi-channel architecture. Each V1S stations receives from only one channel, allocated on a per station basis, and always transmits a cell on the appropriate channel for the destination station.

From the functional point of view, the RAM array is organized in one FIFO and four transmit FIFOs one for the every channel. Each FIFO is implemented as a circular buffer 256 cells long.

The RAMs are shared by both the receive and transmit sides of the station, and also no word of data can be sent or received without first passing, in FIFO fashion, through them. The essential point is that there is contention for the RAM, and the maximum data rate through the stations, when simultaneously transmitting and receiving, is 25 Mb/sec.

The V1S interface perform no protocol processing; cell headers are copied over the VME interface, when generated by or for checking by the host processor board. With V1S there is no DMA, each 32 bit word must be copied by the processor.

The V1S interfaces help the CPU in one way: they allow received cells to accumulate in the interface, in the receive FIFO, until a cell is encountered which generates an interrupt condition. The interface looks in the cell data fields which are specific to the MSDL segmentation and reassembly protocol (details in the next section). The interrupt condition is programmable on a per VCI (Virtual Circuit Identifier) basis, and can set to interrupt on:

1. end fragment cell from an MSDL PDU only,
2. beginning MSDL fragment cell only,
3. the both mentioned above,
4. a per cell basis (every cell on that VCI).

No interrupts are required for transmitting; the transmit side is assumed to be always ready, which it is, owing the light loads on our current network.

4 Protocol Architecture

The CBN is currently running under the MSN (Multi-Service Network) protocol stack supported by the Wanda micro-kernel. This was originally developed in the Cambridge

Figure 3: Relationship of MSN elements to OSI.

Type	VCI	Data Field	Response
1	16	256 + 16 = 272	1

Figure 4: Backbone Ring Cell Format. Field sizes are in bits. The response and type bits are supported in the hardware, but not used by MSDL.

Computer Laboratory and is currently ported onto many popular workstations and VME boards.

The layered reference model [3] for the MSN architecture is presented in Figure 3 Certain aspects of this model are similar to those of the ISO OSI reference model. The use of different layers to provide services is still present.

The network layer protocol MSDL (Multi-Service Data Link) performs fragmentation and re-assembly on a per-VCI basis. MSNL (Multi-Service Network Level) has no impact on the performance results reported here, since in the MSN architecture, network level interconnections are not multiplexed over MSDL virtual circuits. MSNL simply performs out-of-band connection set up. The MSDL layer is based on lightweight virtual circuits, referred to as associations, where there is no hop-by-hop error recovery, and any node involved in the circuit is free to unilaterally terminate it at any time.

The MSDL protocols can accommodate any cell size, but when used over fixed size cell network such as the CBN, MSDL must use the physical network size. The CBN was designed with the cell size the same as was used on the CFR [4], since both preceded the CCITT adoption of 5 + 48 bytes [5]. The CBN (and CFR) use a 4 + 32 format, with the header including 16 bit VCI and 16 bits of SAR information in the header (Figure 4 and 5).

This paper refers to the CBN V1S interface evaluation, so the main issue is of the efficiency of the MSDL implementation on the current hardware. So that the MSN software suite could easily be ported onto multiple different network types and configurations, the subset of functions which have to be implemented in the network driver was limited only

VCI	Sequence	RID	Part	Start
16	8	6	1	1

Figure 5: The format of the first 32 bits of a CBN cell when MSDL is being used for block fragmentation. Field sizes are in bits.

to the fragmentation operation and the hardware depended send and receive operations. The more sophisticated re-assembly operation, which is not as hardware dependent , was performed in the separately compiled MSDL layer. Hence on the transmit side, the driver software has direct access to the association's I/O buffers, but the same is not true for the receive side. This approach is fully justified by the demand of easy portability, but in the context of very high speed networking, needs more careful consideration.

The implementation of MSDL for the Backbone Ring uses the first two data bytes of a cell for fragmentation and re-assembly information. As shown in Figure 4, only the first 32 bit word of the cell need be manipulated by MSDL protocol. The RID is a re-assembly indentifier which is common for all cells from one block. The Start flag is set for the first cell from a block and for this first cell the Sequence number contains the number of cells that compose the block. The Sequence number counts down for each cell so that a value of one indicates the last cell of a fragment. If the Part full flag is set, then the cell body does not all contain valid bytes is stored in the last byte of the cell.

5 Preliminary Performance of V1S CBN Station

The performance results presented in this section were measured over a CBN of approximately 10 km long working at a frequency 512 Mhz. The rotational latency was 22.7 microsecond the physical ring contained 7 CBN frames, giving a total of 28 slots. The ring had no other traffic during the experiments.The experiments concerned:

1. Measurement of the maximum transmit operation speed,
2. Analysis of the data transfer speed achieved between two CBN stations,
3. Measurement of the two-way delay transfer time (ping time).

According to the former description, the MSDL, block of data is directly copied by the driver software from process I/O buffer space, hence the final transmit operation is determined by: buffer handling overhead, fragmentation operation overhead, the speed of copying the data to the FIFO in the interface and the underlying performance of the hardware communication subsystem itself. The obtained results have been shown in Table 1.

he increase of transmit speed speed with data block length is explained by the amortization of buffer handling cost and the reduction in frequency of interaction with the driver. For the long data blocks the obtained in-block transmit data rate was 16 Mbps. This corresponds to over 18 Mbps actual transmit rate (headers + data).

The first measurements of the data transfer speed between two CBN stations were rather disappointing. A careful review of the receive side software made clear that a few improvements were possible:

no.cell	Block length [Byte]	Rate [Mbps]
1	32	1.2
2	64	2.28
5	160	4.76
10	320	7.31
20	640	9.88
30	960	11.27
40	1280	13.33
44	1408	14.54
63	2016	13.33
125	3200	14.54
250	6400	15.23

Table 1: Transmit data rate versus number of cell per block.

No cell	Block length [Byte]	stage 0 rate [Mbps]	stage 1 rate [Mbps]	stage 2 rate [Mbps]	stage 3 rate [Mbps]
1	32	0.87	0.90	0.91	0.91
2	64	1.51	1.62	1.68	1.74
3	160	2.78	3.04	3.44	3.6
10	320	3.8	4.3	5.15	5.8
20	640	4.65	5.4	6.88	8.15
30	960	5.03	5.99	7.77	9.48
40	1280	5.25	6.28	8.23	8.88
44	1408	5.31	6.36	8.40	10.4
63	2016	5.38	6.74	8.75	11.4
125	3200	5.8	7.1	9.48	12.5
250	6400	6.0	7.34	9.89	13.33

Table 2: Transfer data rate versus number of cells per block for different stages of the receive side modification.

No. cell	Block length [Byte]	First delay [μs]	Last delay [μs]	Each delay [μs]
1	32	562	562	562
2	64	587	625	587
5	160	694	818	694
10	320	875	1150	881
20	640	1225	1813	1256
30	960	1581	2469	1619
40	1280	1937	3125	2019
44	1408	2075	3394	2150

Table 3: Two-way time delay versus length of data block for the three different interrupt condition settings.

1. Removing the additional copy operation from the CBN receive FIFO to the buffer inside the CBN driver by direct copying to the association buffer.
2. Merging of the association handling and re-assembly operation within driver receive interrupt handler, that eliminates an additional up-call.
3. More efficient coding of the combined association handling, re-assembly, and copy algorithms.

The obtained transfer rate after each step of improvement is given in Table 2. Stage 0 corresponds to the original version of software. For the longest data blocks, the obtained transfer data rate was over 14.2 Mbps, which corresponds to over 15.6 Mbps transfer rate and was limited by the speed of receive process.

The two-way delay transfer time (ping time) is the time for a block data to be sent to a process in the remote station and back to the sending process. This parameter characterises the latency of the communication subsystem and speed of the Wanda kernel. The obtained results for the improved version of software and the different setting of the receive VCI condition are shown in Table 3. The table shows that the response time is almost independent of whether the interrupt condition is set for start of block or on every cell. In contrast, setting the interrupt condition for the end of data block breaks the receive-transmit process pipelining and leads to a substantial increase of the latency. Evidently there is a trade off between context switching overhead and increase of latency caused by the breaking of the transmit/receive pipeline and the consequent loss of cut-through.

6 Future Work and Conclusion

On the measurement side, we need to perform similar experiments on a more heavily loaded network. The V1S interfaces include false traffic generators, so that the the load can be artificially increased. Another interesting case is the duplex communication situation, where the interference between simultaneous receive and transmit processing needs examining. Finally, the effect of the ATM substrata on higher level protocol performance needs measuring.

As far as hardware development goes, we must bear in mind that V1S interfaces were designed as a cheap interface with about 30 Mbps simultaneous receive and transmit throughput. There has been an impact on host throughput as a result of the current CBNs being operated at only about half their design speed, that is 512 MHz instead 1000 MHz.This has increased the contention resolution time for FIFO buffer access and the currently obtained 12 μs copy time might have been expected to be about 8 μs in a full speed system.

A simplistic, single cell DMA engine is being designed. This should be able to copy a cell in about 4 μs, and if this copying time becomes parallel with the per-cell protocol processing activity on the microprocessor, an overall speed up of about 3 times may be envisaged.

In general, it seems that the only satisfactionary solution needs some form of dual-processor architecture, where the 'host' processor is relieved of as much context switching overhead as possible. The results in this paper enable us to predict that a current technology, general purpose processor might be able to keep up with per-cell processing for rates up to 50 Mbps (75 Mbps with 48 byte cells).

Acknowledgements

Many thanks for invaluable help are due to Joe Dixon, Mark Hayter and the rest of the Wanda group at the Computer Laboratory. Thanks are due to the rest of the systems group, and especially to D (Mac) McAuley for the MSN picture. Dimitris Lioupis, David Milway and Andy Hopper greatly contributed to the Backbone Network project. The work in this paper was funded by Olivetti Research Limited.

References

[1] 'The Cambridge Backbone Network.' DJ Greaves D Lioupis and A Hopper. IEEE Infocom 90, San Francisco June 1990. Also in Proceedings European Fibre Optic Conference (EFOC/LAN 88) Amsterdam, June 1988.

[2] 'Backbone Ring V1s Station Interface Specification.' DJ Greaves. Olivetti Research Limited 1990.

[3] 'Protocol Design for High Speed Networks.' University of Cambridge TR 186. DR McAuley, December 1989.

[4] 'The Cambridge Fast Ring Networking System.' A Hopper and RM Needham. IEEE Transaction on Computers, Vol.37 no 10, October 1988.

[5] CCITT I series recommendations, especially I.321, Geneva May 1990.

[6] 'An analysis of TCP processing overhead.' DD Clark, V Jacobson, J Romkey and Salwen, IEEE Communication Magazine, June 1989.