Communicating Process Architectures 2006
Peter Welch, Jon Kerridge, and Fred Barnes (Eds.)
IOS Press, 2006

SpaceWire – DS-Links Reborn

Barry COOK and Paul WALKER
4Links Limited, The Mansion, Bletchley Park, MK3 6ZP, UK
{Barry, Paul}@4links.co.uk

Abstract. DS-links were created to provide a low-latency, high performance data link between parallel processors. When the primary processor using them was withdrawn these links largely disappeared from view but were, in fact, still being used (albeit not for parallel computing) in the Space industry. The potential for these links, with their simple implementation, led to their adoption, in modified form, for a growing range of data communication applications. In 2003, the European Space Agency published a definition of DS-links known as SpaceWire. We briefly describe the original DS-links and detail how SpaceWire has kept or modified them to produce a now popular technology with a rapidly increasing number of implementations and wide take-up.

Introduction

Concurrent processing systems using a number of physically separate processors must exchange data in a timely fashion. Performance is dependant on data throughput and, more crucially, communication latency. The first system-on-a-chip processor designed for multi-processor computing, the Transputer, contained four communication links running at 10 or 20Mb/s which, at the time, was considerably faster than existing networks, such as Ethernet, could supply. That this speed, matched to the processing capability available, could be used effectively was demonstrated by a number of impressive applications such as real-time ray-tracing graphics programs.

Development of a faster processor, the T9000, required faster communication links. These were upgraded to 100Mb/s and facilities provided to multiplex many logical links (virtual channels) over a single physical link. Complex networks could be built using a routing switch, the C104. These communication links were standardised as IEEE-1355 [1]. These links and networks were described in [2].

Alas, the Transputer family was not developed further and the link technology was only used in a limited number of areas for which it was particularly well suited. One notable instance being a 1000-node system at CERN, used for data capture in high-energy physics experiments – this system proved the long-term reliability of such links.

Spacecraft applications, due to the inherent difficulty of making changes after launch, require components and systems that are predictable and reliable. They also require ever more capability as missions become more adventurous. Increased data transmission throughput requirements led to consideration of new network implementations. The link technology of the T9000 was considered a good place start and, with relatively minor changes, it has become a new standard for space applications [3].

IEEE-1355 was made available in a Radiation-hardened version and used for point-to-point links in the successful missions Rosetta [4], Mars Express [5] and Venus express [6]. Early versions of SpaceWire are flying on SWIFT [7] and several others (classified for

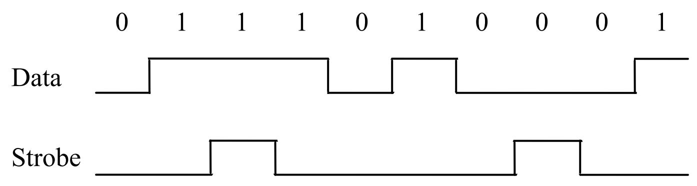

Figure 1. Data-Strobe encoding

commercial or other reasons). SpaceWire is planned for use on a wide variety of different missions throughout the world. The European Space Agency plans to use SpaceWire for most, if not all, of its future missions. A number of national missions, such as Taiwan's Argos Satellite, are using SpaceWire. Key US missions are the James Webb Space Telescope (formerly known as "Hubble 2") [8], the Lunar Reconnaissance Orbiter [9] and GOES-R [10].

One contrast with competing technology such as Ethernet is in the difficulty of implementation. Fast Ethernet, and Gigabit Ethernet even more so, requires analogue processing of the received signals to extract data – a silicon-hungry process. SpaceWire achieves similar data rates with a minimal silicon requirement. At least one team has abandoned attempts to implement Ethernet in a form suitable for Space applications.

1. DS-links – as Originally Defined

DS (Data-Strobe) links provide a high performance, low latency, point-to-point communication mechanism. Data is sent as self-contained packets and a higher level protocol provides message transfers.

1.1 The Physical Layer

Data encoding by the use of a pair of lines provides an important facility to transmit high-speed data whilst tolerating relatively broad skew margins. Data is sent, unaltered, on a signal line known, unsurprisingly, as "Data", and the second line, known as "Strobe", carries a signal that can be used to re-create a clock at the receiver. The strobe signal is generated very simply in that if, at any bit period boundary, the data signal does not change, then the strobe signal does, see figure 1.

The data clock signal is re-created simply by an exclusive-or of the data and strobe signals. This clock can be used to latch the data, noting that both the rising and falling edges must be used, and to drive the receiver circuits. This extremely simple transfer of the clock signal, without the need for phase-locked loops gives an "auto-baud" facility. Speed variation can be used to conserve power by running the link slowly when high speed is not required, as when idling and sending only null tokens, with an instant return to full-speed operation. T9000 links run at 100Mb/s and a differential signal is used (IEEE-1355: DS-DE) to give transmission over a few metres of twisted-pair cable.

New connectors were designed for DS links in a format giving a high density and a large number of connectors can be placed on standard circuit boards.

The IEEE-1355 encoding scheme has been adopted for the IEEE-1394 standard [11] and the Apple Computer version of IEEE-1394 known as FireWire.

1.2 Character Level

The sequence of bits transmitted from device to device is logically broken into a series of tokens. Each token contains a parity bit and a bit indicating that this is either a control token, with a total length of 4-bits, or a data token, with a total length of 10-bits, see Figure 2. The parity bit covers the "data" bits preceding it, and the control bit following, to give security against errors in transmission. Figure 3 illustrates a data stream, showing the data-line, and the bits covered by each parity bit.

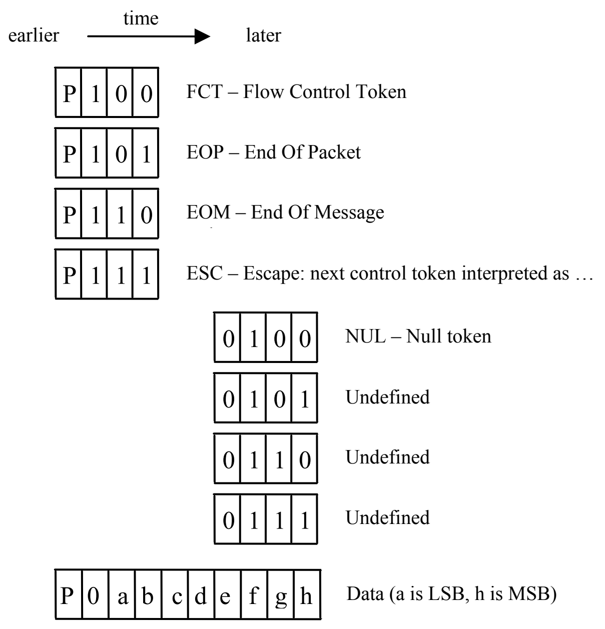

Figure 2. Tokens

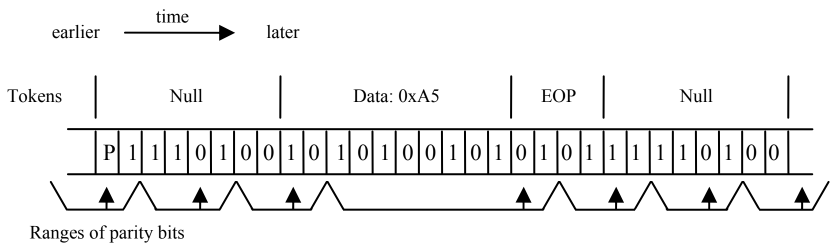

Figure 3. Example stream of bits

There is no provision for re-synchronisation to the continuing data stream and the above interpretation relies on the link starting up from idle and staying in step thereafter. The inserted parity bits allow detection of errors that may cause a loss of synchronisation, and thus result in the link stopping and re-starting. Bits are continuously sent over the link – when there is no useful data to transfer null tokens are sent.

In order to ensure that no data is lost, no tokens that need to be stored at the receiver (data, EOM or EOP tokens) are sent until the receiver indicates it has space for them. The receiver sends a flow control token (FCT) for each 8-bytes it can accept, several may be sent to indicate a large buffer, and they may overlap data transfers to allow uninterrupted data transmission.

This low-level flow control happens frequently and is best hidden, by suitable hardware link controllers, from message handling software or hardware.

1.3 High Level Protocol

Efficient transfer of data requires low overheads and sending each message as a single, possibly large, transaction appears to be best. There are two reasons why this may not be the best strategy for a system:

- A message that becomes blocked part-way through transfer, due to full buffers, may prevent the transfer of other, unrelated messages. This situation is likely to be a problem in networks with wormhole routers, such as T9000/C104 systems.
- The receiver may need to store, temporarily, incoming messages if they arrive before the receiving process has allocated space for them.

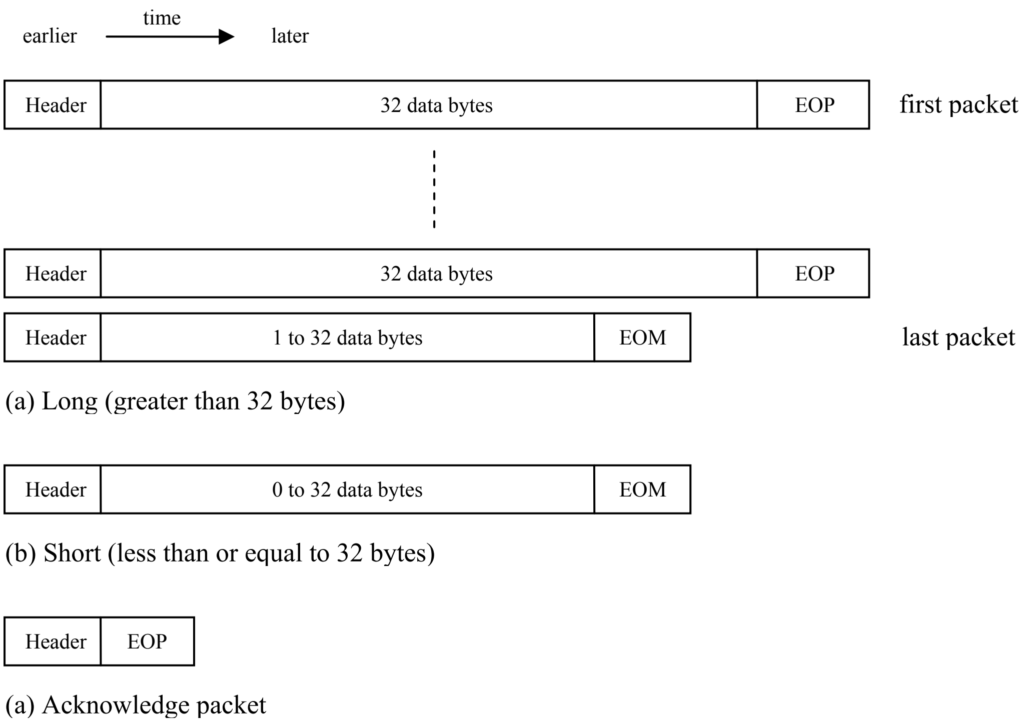

Figure 4. Packet formats

Instead, the T9000 breaks messages into packets with a known maximum size (32-bytes) so that receiver buffers can be allocated and network blocking is limited – see Figure 4. A short (0 to 32-byte) message is sent with an end-of-message (EOM) token. Longer messages (more than 32-bytes) are broken down and one or more 32-byte packets terminated with an end-of-packet (EOP) token are sent before a final 32-byte, or less, packet with EOM. Each packet has to be acknowledged before the next is sent so that the receiver has to buffer, at most, one packet of 32-bytes.

Information to guide the routing of packets around the network is added in the form of a header at the front of the packet.

Since packets terminated with EOP are as large as possible, it is possible to use a shorter than expected packet with EOP as an acknowledge. Usually this is a zero-length packet, as shown in figure 4.

The flow of data and acknowledge packets is illustrated in figure 5 for a message that has been split into more than three packets.

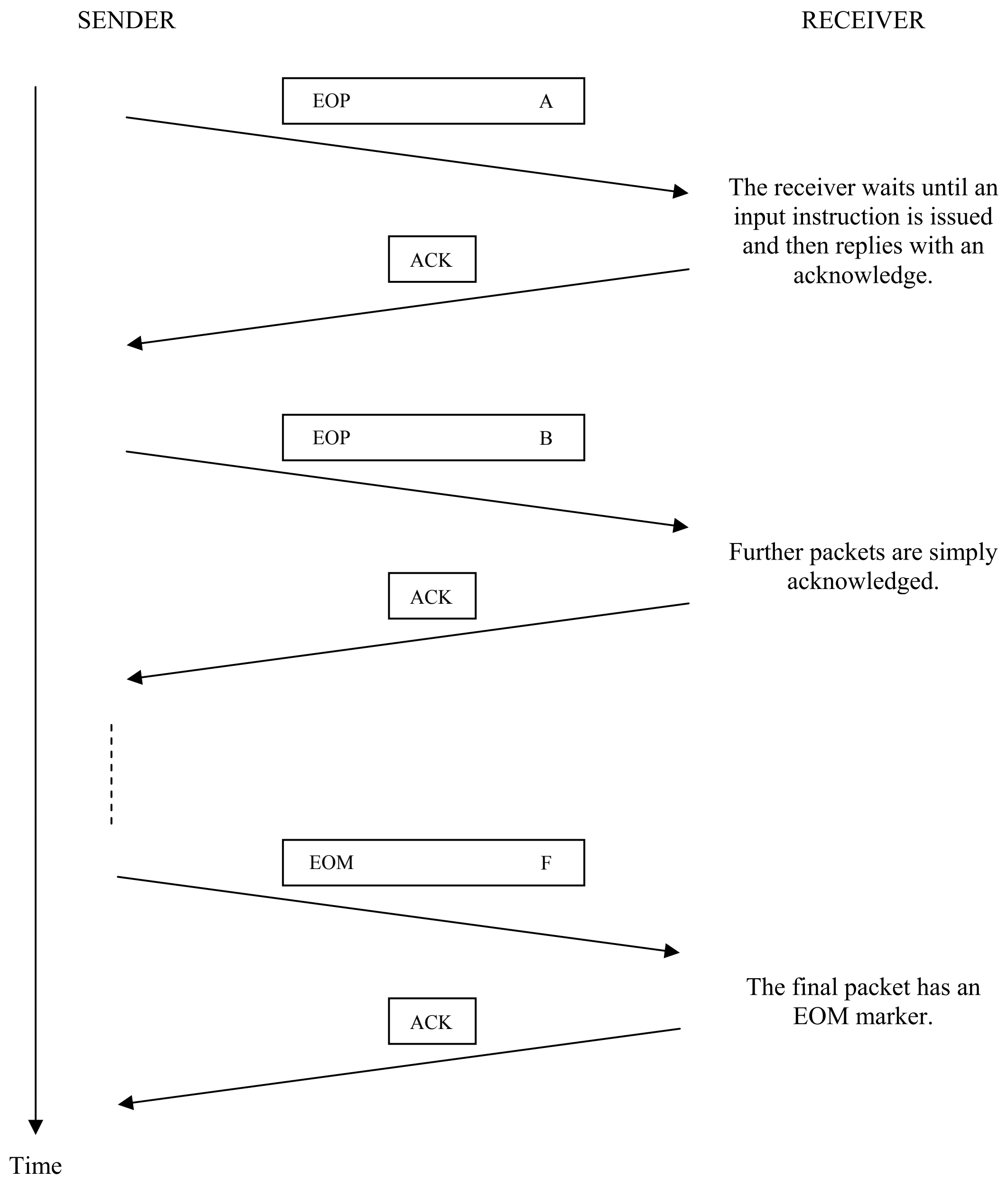
SENDER
RECEIVER
EOP A
The receiver waits until an input instruction is issued and then replies with an acknowledge.
ACK
EOP B
Further packets are simply acknowledged.
ACK
EOM F
The final packet has an EOM marker.
ACK
Time

Figure 5. Flow of packets and acknowledges

1.4 Scheduler Interaction

Apart from automatically breaking messages into packets, and re-assembling them, the most important feature of the link engine is its close integration with the process scheduler. Processes waiting on the transfer of a message are automatically re-inserted into the process run queue as soon as the massage has been transferred. This, hardware, scheduling is very much faster than systems using interrupts and software to re-schedule processes. The improved (reduced) latency this imposes on message transfer is directly reflected in improved (greater) performance.

1.5 Routing Switches

Networks of processors are connected via one or more routing switches – the C104 [12] [13]. This switch has 32 ports and a very high performance implementation. Wormhole routing is used to minimise latency; packets were forwarded toward their destinations as soon as the route was clear.

Routing is based on a packet header, the first byte(s) of a packet being used to access a table in each switch and values contained in the tables used to select the appropriate output port. Output port grouping increases throughput and/or provides redundancy [13].

Configuration of routing switches is through two dedicated links which allow a set of routers to be daisy-chained together.

2. SpaceWire

The relatively small silicon area needed to implement a DS-link is particularly attractive to users concerned to minimise volume, and weight – such as the Space industry.

Some aspects of IEEE-1355 are not suited to the Space environment and others are not familiar enough to be accepted as-is. As a result, a variant link has been specified that is very similar to DS-links but contains some significant differences. The result is known as SpaceWire, or SpW and is specified in a European space industry document [3]. This section, using section heading corresponding to the last section, describes the differences between SpaceWire and IEEE-1355.

2.1 The Physical Layer

Data-strobe encoding has been retained exactly as described above. Where IEEE-1355 used PECL logic for the serial interface, SpaceWire used LVDS – at much reduced power consumption. Advances in technology allow speeds of 200Mb/s to be easily obtained with test equipment such as that from 4Links [14] is specified to 400Mb/s (and actually operates to 500Mb/s) and at least one design is running at 625Mb/s. The practical limit to speed turns out to be significantly affected by the connectors chosen for SpaceWire.

The connectors designed for IEEE-1355 are not considered suitable for Space applications where a more robust construction is required. Cable suitable for the wide temperature range in space, and the vacuum environment, are much thicker than needed for terrestrial applications. SpaceWire chose to use micro-miniature D type connectors with 9 pins – similar in shape and style to the serial interface connectors found on a PC, but smaller (and much more expensive!). These connectors do not attempt to provide a controlled electrical environment for differential signalling at high speed. As a result, higher speed links increasingly are limited by the performance of the connectors and this is a significant issue as speeds rise above 500Mb/s. A better connector has been designed but has not, yet, been accepted as an official alternative.

2.2 Character Level

The same coding is retained as for IEEE-1355 (figure 3) but modified in the case of end-of-packet, and extended to provide an additional feature – time codes.

2.2.1 End-of-Packet

IEEE-1355 defined the two end-of-packet markers as EOP-1 and EOP-2, noting that EOP-2 may be used as an alternative end-of-packet marker, or as an indication of error. The T9000 uses EOP-2 as an alternative end marker to provide end-of-packet and end-of-packet tokens – although it was also useful for error conditions in a network [2]. SpaceWire restricts EOP-2 to an indication of an error condition and renames it EEP – error-end-of-packet. Virtual channels, if required, cannot easily be implemented as they were in the T9000.

2.2.2 Time Codes

Several code combinations, those starting with an ESC token, were undefined in IEEE-1355 (although one or two were tentatively reserved for improving link performance but never, so far as we know, appeared in a commercial chip).

P	1	1	1	1	0	t_7	t_6	t_5	t_4	t_3	t_2	t_1	t_0

Figure 6. Time-code is ESC-Data

SpaceWire defines an ESC token followed by a data token as a "time code" – Figure 6. This is intended to provide a broadcast mechanism to distribute time ticks throughout a system.

The data byte contains a value that is incremented each time a time-code is sent from the (single) time-code master. If the received value corresponds to the last received value plus one (modulo the size of the counter – 6-bits of the 8 available) then a correct code has been received and action can be taken. If it is not as expected then no action is taken. In all cases the local value is updated with the received value ready for the next time-code to arrive.

Each routing switch in a network receives time codes and, if they contain the expected value, broadcasts them to neighbouring switches and end-nodes. Time-codes are thus spread from switch to switch/end node throughout the whole system.

Redundant networks are not trees but graphs – they contain alternate routes and bidirectional links on routes cause loops. Loops are normally disastrous for broadcast traffic and thus strictly forbidden in Ethernet networks (and disabled in IP networks by reducing the network to a tree with the Spanning Tree algorithm [15]). Redundancy is a fundamental requirement in a reliable network and hence broadcast is not supported by SpaceWire – except for time codes. A time code that is received by a node that has previously seen the code will identify it, because it contains the last-seen value, as not-for-action and discard it rather than cause an infinite sequence of transmissions.

The received time codes suffer delays which depend on their route through a network and jitter that depends on the link implementation – times of the order of tens of microseconds are typical in a small network. It has been shown that delay can be compensated and jitter can be reduced to the order of nano-seconds by careful design of link components [16].

End-node silicon produces time ticks and, optionally, time values when a time code is received. It is envisaged that for more stable timing a local oscillator would be phase locked to the to the incoming tick The time value has only 6-bits giving only limited information. A separate message giving coarse time information, with loose delivery requirements, is used in association with time-codes to give a complete time. For example, messages giving date and time of the next minute are sent normally and time ticks once per second indicate precisely when the next minute starts and hence the combination gives a complete, precise, date and time.

2.3 High Level Protocols

Space applications are varied and although there is strong pressure to use a minimal set of protocols (in the hope of increasing inter-operability and reducing costs) it is acknowledged that more than one protocol is required. A SpaceWire network is able to carry more than one protocol at the same time and a protocol for protocols has been specified – each packet must identify the protocol it uses. Each packet, as it arrives at its destination, starts with a protocol identifier (PID), one or two bytes, which indicate how that packet should be processed [16]. One incidental benefit is that a mis-routed packet sent to a node that does not know how it should be handled can immediately reject it (mis-routing is just one of the many possible abnormal behaviours possible in the hostile Space environment).

In contrast to Transputer thinking, in which the packet receiver has complete control, the first new protocol defined – and expected to be widely used – is the Remote Memory Access Protocol (RMAP).

Some ingenuity in the use of SpaceWire protocols, however, would allow close integration with a scheduler and even virtual channels. For example, RMAP, is defined as a memory transfer mechanism … but its parameters could be re-interpreted. There is no reason why an RMAP "memory address" could not be treated as a virtual channel number!

2.3.1 Remote Memory Access Protocol (RMAP)

Permitting data transfers to or from memory on a remote node, RMAP [17] is seen as underpinning many higher level protocols.

A packet contains the memory address at which the data is to be placed or retrieved, the number of bytes to transfer, how the transfer is to take place, what response, if any, to give, a transaction identifier, data (if a write) and limited error detection. Transfers can take place as soon as data arrives or can be delayed, buffering the data, until it has been checked for errors. Acknowledgements can be sent indicating the status of the transfer, or the transaction can be silent.

There is not space here to describe this protocol in detail and it is not, in any case, completely final nor published at the time of writing. It is, however, worth mentioning two aspects of the protocol – overhead and performance. Although compact for its function, memory access necessarily requires several bytes of overhead for memory address, data length, function etc. and this seems massive compared with the T9000 virtual channel format. Although not specified as a requirement, the protocol is simple enough to allow its implementation in hardware with DMA transfers. This allows the data transfer to be completely offloaded from the CPU until an interrupt signals the end of the transfer.

Several transfers can be in progress at the same time and can provide, in effect, capabilities similar to the T9000 Virtual Channel Processor (VCP), although there is a fundamental difference in security – the VCP closely controlled access at the receiving end whilst RMAP is given relatively free-reign to read/write any data is wishes. As mentioned above, we can see ways to reconcile the differences and use RMAP in a more secure manner but these have not been discussed within the SpaceWire community.

2.4 Scheduler Interaction

Nothing in the SpaceWire standard is aimed at supporting close integration with a scheduler. Traditional interrupt mechanisms are assumed. Although this can supply multi-processor performance in the way that Ethernet does, this approach fails to achieve the superb performance demonstrated by the Transputer with its integrated communications and scheduler.

2.5 Routing Switches

SpaceWire routing switches are designed to be connected as networks, typically with more than one physical path between nodes to provide redundancy and enable fault tolerance. The method of directing a packet through the network is the same in principle but differs in detail from that used with C104 routing switches.

Each packet contains a sequence of bytes which, although indistinguishable in form are interpreted according to location. Each routing switch interprets the first data byte as an indicator of how to route the packet – and is often called a “header” byte. The header byte may be interpreted as a physical or logical address.

Physical addresses directly indicate which port of the switch is used to output the packet and that byte is deleted from the packet (“header deletion”). Valid physical addresses are 0 to 31 where ports are numbered from 1 to 31 – port 0 is used to direct the packet to the control port of the switch itself. Router configuration uses the data network – the mass (as in ‘weight’) of a separate configuration network being unacceptable.

Logical addresses are used to address an internal routing table to determine which port(s) should be used to output the packet. It is possible to specify a group of ports as equivalent (“grouped adaptive routing”) to provide redundancy and/or increased bandwidth. Broadcast is NOT supported – indeed, it is forbidden; a network containing multiple paths contains loops and broadcast would induce deadlock. In this mode the header byte may be deleted or left in place – a single logical address can be used for all hops in the network from source to destination, provided all routing tables are set appropriately.

3. Components

Although rather hidden from wide public attention by reason of being within a specialised application area, the Space industry, there are now many devices and designs for the SpaceWire variant of DS-links. There are chips implementing SpaceWire links with simple streamed data interfaces, routing switches and integrated CPU with SpaceWire devices – although the CPU’s are not high performance by today’s standards, they are optimised for radiation tolerance.

Link, router and CPU designs are also available as Intellectual Property (IP) for a wide, and widening, range of implementations from Field Programmable Gate Arrays (FPGA) to full-custom Application Specific Integrated Circuits (ASIC). Several suppliers are active and generating new designs with a variety of interfaces.

3.1 Link Chips, IP and Interfaces

Atmel have produced a radiation hardened IEEE-1355 chips, one having a single link, the SMCS116 (also known as T7906E) and one having three links, the SMCS332 (also known as TDSS901E). A modified version of the latter will shortly become available in a version that (nearly) implements SpaceWire – the SMCS332SpW.

On the surface, a SpaceWire link design appears simple and many groups have started designing their own. There are, however, some subtle aspects that can catch the unwary – and fool test-by-simulation approaches to validation. The highest performance design we are aware of reaches 625Mb/s.

4Links has produced six generations of SpaceWire design for various devices and in various styles – it is surprising how many different ways there are to implement the specification. We have found some areas of the design to be more prone to error than others and the design tools available fail to give the necessary support. Some design style / tool combinations lead to less reliable results than others. Our focus is on test equipment and the designs we now use have proved very reliable – using a design style that is well supported by design tools – although not the smallest possible implementation. We are considering releasing some of the lower-performing versions of link and support functions in netlist form.

Due to the variety of possible designs it is difficult to give a definitive size estimate – further complicated by the ability to choose buffer sizes and optional features such as time-codes. Very roughly, the smallest designs are in the region of a one to five thousand FPGA gates (use your favourite conversion factor to get ASIC gates). One design from 4Links, including time codes, a 16-byte receive buffer (SpaceWire allows 8 to 56 bytes), capable of link speeds exceeding 50Mb/s and implemented on a Xilinx™ Virtex 2 Pro™ FPGA requires 173 flip-flops and 371 (4-input) LUT's, "6000 gates" – just 2.5% of a V2P20.

Power consumption is equally difficult to state, in the general case, as it varies with features, clock rates / link speeds and process technology.

3.2 Router Chips, IP and Boxes

By C104 standards, the routing switches available and forthcoming are modest – 8-ports compared with C104's 32-ports. ESA is funding a routing switch design and first silicon is due by mid-2007. This will have a local port as well as 8 SpaceWire ports.

4Links offers IP for a simple routing switch in Xilinx™ FPGA. They also offer a flexible routing switch in their standard equipment range with considerably enhanced capabilities. Each unit can be configured as one or more routers, and/or a router may span more than one unit. Large routing switches can thus be built to match, or even exceed, the C104 capability.

3.3 Access to SpaceWire

4Links commercial SpaceWire interfaces use TCP/IP data streams which are supported by virtually all available operating systems – drivers being supplied as part of the operating system. Experience has shown that this approach guarantees both portability and performance – it is easily possible to achieve throughputs above 95% of that theoretically available on 100Mb/s Ethernet. Gigabit Ethernet is used to support the faster SpaceWire interfaces and performance figures here are equally encouraging.

Users are provided with access to the low-level TCP data stream and simple APIs, for which the full source code is supplied. APIs are currently available in C and Java. A simple example of the C interface is shown in Listing 1.

It has been demonstrated that it is possible to integrate network-based communication as channels in languages such as occam [SPoC / KRoC].

4. Conclusion

Far from being a technology of the past, DS-links – in the form of SpaceWire – are actively being developed. As with Transputer links, take up followed standardisation. There is an understandable reluctance to adopt an unknown technology but the decision is easier when there is an approved standard. The process is slow and relies on long-term commitment from advocates who not only tell the world there are benefits but demonstrate them. 4Links have a demonstration fault-tolerant network that has been a significant factor in convincing engineers and, perhaps more importantly, managers that they would benefit from adopting SpaceWire.

```
#include "EtherSpaceLink.h"
char buffer[1024];
int n, EOP;
...
// Open a connection to the interface
EtherSpaceLink link = EtherSpaceLink_open( “192.168.0.24” );
// Set the SpaceWire link speed
EtherSpaceLink_set_speed ( link, 200 );
// Send a 40-byte packet
EtherSpaceLink_write_packet( link, buffer, 40, EtherSpaceLink_EOP );
// Receive a packet
n = EtherSpaceLink_read_packet( link, buffer, 1024, &EOP );
// Close connection
EtherSpaceLink_close( link );
```

Listing 1. Example use of the C user interface

Point-to-point data communication using IEEE-1355 enabled users in the Space industry to gain experience with the technology and see its advantages. Initial applications for SpaceWire are also simple point-to-point connections. The use of networks to provide redundancy and fault tolerance is still developing.

To date, communication has been between a controlling processor and remote data sources and sinks. Future plans for the remote data controller – known as a Remote Terminal Controller (RTC) – include a processor. Thus we see a trend toward multi-processor systems and issues of reliability will generate interest in CSP [19] and related developments. There is active development of integrated processors and links but no indication that a Transputer-like design is planned. Parallel processing is an emerging interest – a single chip four processor design [20] is well advanced – primarily to supply future processing demands.

Many of the research results previously developed will have a new application in the Space industry and safety-critical systems in general. Many ideas concerning reliable computing that were not taken seriously are important to the Space industry and, in turn, to other industries due to their Space pedigree.

Industrial DS-links have spun-in to Space, been reworked as SpaceWire, and are set to Spin-out again to industries that weren’t ready for them when originally offered. Associated theories of guaranteed behaviour by networks and software may very well follow.

References

[1] IEEE 1355-1995: Standard for Heterogeneous InterConnect (HIC) (Low Cost Low Latency Scalable Serial Interconnect for Parallel System Construction), IEEE Standards Department, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA

[2] Barry M. Cook, Data-Strobe Links and Virtual Channel Processors, in A. Bakkers editor “Parallel Programming and Java”, Proceedings of WoTUG-20, IOS Press, 1997, ISBN 90 5199 336 6

[3] European Cooperation on Space Standardization, ECSS-E-50-12A SpaceWire – Links, nodes, routers and networks (24 January 2003)

[4] Rosetta Mission Home Page, European Space Agency, `http://sci.esa.int/rosetta/`, last visited: August, 2006.
[5] Mars Express Home Page, European Space Agency, `http://mars.esa.int/`, last visited: August, 2006.
[6] Venus Express Home Page, European Space Agency, `http://sci.esa.int/venusexpress/`, last visited: August, 2006.
[7] Swift Gamma-Ray Burst Mission, NASA Goddard Space Flight Center, `http://swift.gsfc.nasa.gov/docs/swift/swiftsc.html`, last visited: August, 2006.
[8] James Webb Space Telescope, NASA, `http://www.jwst.nasa.gov/`, last visited: August, 2006.
[9] Lunar Reconnaissance Orbiter, NASA Goddard Space Flight Center, `http://lunar.gsfc.nasa.gov/missions/`, last visited: August, 2006.
[10] Geostationary Operational Environmental Satellite (GOES) R series, NASA, `http://science.hq.nasa.gov/missions/satellite_67.htm`, last visited: August, 2006.
[11] IEEE-1394-1996 High Speed Serial Bus, IEEE Standards Department, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA
[12] ST C104 Asynchronous Packet Switch, data sheet, SGS-Thomson Microelectronics
[13] Networks, Routers and Transputers, Ed. M D May, P W Thompson & P H Welch, IOS Press, ISBN 90 5199 129 0
[14] 4Links Home Page, `http://www.4links.co.uk`, last visited: August, 2006.
[15] IEEE 802.1D Standard for Local and Metropolitan Area Networks – Media Access Control (MAC) Bridges, IEEE Standards Department, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA
[16] Barry Cook, Reducing Time Code Jitter, International SpaceWire Seminar, ESTEC, November 2003 (available at www.4links.co.uk/reducing-time-code-jitter.pdf)Low jitter TC
[17] PID – Process Identifiers, working document of the SpaceWire working group
[18] RMAP – Remote Memory Access Protocol, working document of the SpaceWire working group
[19] C.A.R. Hoare, Communicating Sequential Processes, Prentice-Hall, London, UK, 1985.
[20] André L.R. Pouponnot, A Giga Instruction Architecture (GINA) for the future ESA microprocessor based on the LEON3 IP core, European Space Agency, ESTEC, Noordwijk, The Netherlands