The Operation and Uses of the SpaceWire Time-Code

International SpaceWire Seminar (ISWS 2003)

4-5 November 2003, ESTEC Noordwijk, The Netherlands

Steve Parkes [(1)]

[(1)] University of Dundee
Applied Computing, Dundee, DD1 4HN, Scotland, UK.
Email: sparkes@computing.dundee.ac.uk.

ABSTRACT

SpaceWire time-codes provide a means of distributing time information across a SpaceWire system [1][2]. Time can be distributed across a large network with relatively low jitter. This time information can be provided as ticks or as an incrementing value which may be synchronized to spacecraft time. Time-codes provide a mechanism for supporting distributed system synchronization. They may also be used to implement isochronous communication channels, complementary to the asynchronous channels provided naturally with SpaceWire.

A time-code comprises the SpaceWire ESC character followed by a single 8-bit data character. The data character contains six-bits of system time (time-field) and two control flags. Each SpaceWire node or router [3] contains a six-bit time counter. A node or router acts as the time-code master and is responsible for distributing time. The time-master interface has a "tick" input, which is asserted periodically (e.g. every millisecond) by its host system. When the time-master link interface receives a tick, it increments its time counter and then immediately sends out a time-code with the 6-bit time field set to the new value of the time counter.

When the node or router at the other end of the link receives the time-code it updates its internal time counter with the new time and asserts a tick output signal. The new time should be one more than the time counter's previous time-value – this fact can be used for checking on time validity. If a node or router receives a time-code that is equal to its current time value then it does not emit a tick output signal. This prevents repeated time-code propagation in a circular network.

When a router receives a time-code it checks that it is one more than the router's current time setting. It then increments the router's time-count and emits a tick signal. This tick signal propagates to all the output ports of the router so that they all emit the time-code. This time-code is the same value as that received by the router since the router time counter has been incremented. If there is a circular connection then the router will receive a time-code with the same time value as the router time counter. When this happens the time-code is ignored. In this way time flows forward through a network reaching all nodes but is suppressed if it flows back due to a circular connection.

With the provision of this basic time distribution function, application level protocols can be used to distribute specific time values at full resolution (not just 6-bits) and to issue time dependent commands etc. The two control flags that are distributed with the 6-bit time-code can be used to broadcast information to all nodes and routers on the network.

This paper describes time-code operation in detail. It explains what happens in the event of a fault where a time-code is lost and looks at time-code latency and jitter. Finally several ways in which time-codes may be used are presented.

TIME-CODES

The SpaceWire time-code is used to distribute system time information along with a pair of control-flags to all nodes and routers in a SpaceWire network. SpaceWire time-code distribution is normally performed periodically with successive time-codes separated by a time-interval determined by the particular application, e.g. 1 ms interval between time-codes.

Time-code structure

A SpaceWire time-code comprises the SpaceWire ESC character followed by a single 8-bit data character. The data character contains two control-flags and a six-bit time-count. The time-code is illustrated in Fig. 1. The six-bit time-count is held in the least-significant six-bits of the Time-Code (T0-T5) while the two most-significant bits (T6, T7) contain the two control-flags. The parity bit (P) in the middle of the time-code is set to one to give the correct parity.

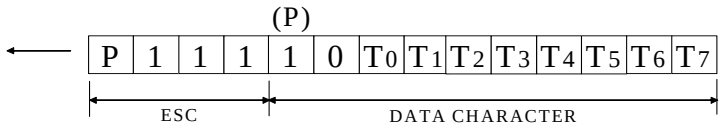

Fig. 1 SpaceWire Time-Code

The two control-flags are for general use, they do not have a specific function for time-code distribution.

Time-code interface

In order to be able to transmit and receive time-codes a SpaceWire CODEC has a time-code interface. The time-code interface comprises two signals, TICK_IN and TICK_OUT, together with an eight-bit time-code input-port and an eight-bit time-code output-port. The eight-bit input and output are split into two fields: a six-bit time field and two-bit control-flag field.

The TICK_IN signal is used to request the transmission of a time-code and the eight-bit input port provides the time-code value to be sent. Whenever TICK_IN is asserted the SpaceWire CODEC will send a time-code immediately after the character currently being transmitted has finished. Note that a SpaceWire CODEC can only send time-codes when it is up and running sending Nulls, FCTs and/or data (i.e. is in the *Run* state).

The TICK_OUT is used to indicate that a time-code has arrived at a SpaceWire interface. TICK_OUT is asserted whenever the link interface is in the *Run* state and the receiver receives a valid Time-Code. The eight-bit time-code output-port is set to reflect the contents of the time-code.

Time-counter

Each node and router contains a time-counter which is used to hold the current 6-bit time value from the time-code time. This counter is also used to validate an incoming time-code and to decide whether the time-code should be propagated. In a node, time-code propagation is up to the application (hardware or software). In a router it is to all the output ports of the router.

During normal operation the time value in a time-code increments from one time-code to the next, rolling round from 63 to 0 when it reaches the maximum possible time value. When a time-code arrives at a SpaceWire node or router its value should be one more than the current value of the time-counter at this node or router. The time value of an incoming time-code is compared to the value of the time-counter. If the time-code is one more than the current value of the time-counter then the time-code is deemed valid. The time-counter is then incremented to the value of the time-code and the arrival of a valid time-code is indicated.

If the time value of the time-code is not one more than the time-counter then the time-code is not valid. In this case the time-counter is set to the value of the newly arrived time-code but the arrival of the time-code is not flagged.

Time master

A single SpaceWire node or router in a system is responsible for generating time-codes. This “time-master” has an active TICK_IN signal which is asserted periodically (e.g. every millisecond) by its host system. Only the time-master should assert its TICK_IN signal, all other nodes must keep their TICK_IN signals de-asserted. To send out a time-code the time-master has to increment the time value on the SpaceWire CODEC time input-port and then assert the TICK_IN signal. The SpaceWire CODEC will then read in the new time-code value and transmit the requested time-code as soon as the current character has finished being sent.

The time-counter in the node or router acting as the time-master, may be used to generate the correct sequence of time-codes for transmission. The six-bit time-counter output is fed to the time-code input-port. When the next time-code is to

be transmitted, the time-counter is incremented and the TICK_IN signal asserted. This causes the SpaceWire CODEC to send the required time-code. The values of the two control-flags sent in the time-code are application dependent. Their values should be updated when the time-counter is incremented.

Time-codes are normally generated periodically by the time-master to provide a regular timing tick for the SpaceWire system. The frequency of time-code is called the tick rate and the period between time-codes is called the tick interval.

Time-codes across a link

When one end of a SpaceWire link is periodically sending out valid time-codes, the SpaceWire interface at the other end of the link receives the time-codes, checks their validity, updates the time-counter with the new time value and asserts the TICK_OUT signal. The host system attached to this end of the SpaceWire link, receives periodic TICK-OUT signals together with the six-bit time value and the two control-flags. The TICK-OUT signal can be used to raise an interrupt or event in a software-driven node.

If a SpaceWire interface receives a time-code that is not one more than the current value of its time-counter then the time-code is not valid and the interface does not emit a TICK_OUT signal.

Router action on receiving a time-code

A router contains a single time-counter. When a link interface on a router receives a time-code it checks that it is one more than the current value of the router's time-counter. It then increments the router's time-count and emits a TICK_OUT signal. This TICK_OUT signal propagates to the TICK_IN interfaces of all the router output ports so that they all emit the time-code. This time-code is the same value as that received by the router, since the router time-counter has been incremented. The time-code is not emitted by the link that first received the time-code.

If there is a circular connection then the router will receive a time-code with the same time value as the router time-counter. When this happens the time-code is ignored. In this way time flows forward through a network reaching all nodes but is suppressed if it flows back due to a circular connection.

TIME-CODE DISTRIBUTION ACROSS A NETWORK

In this section the distribution of time-codes across a SpaceWire network is described.

The initial state of a network is shown in Fig. 2. N1 and N2 are nodes. R1 and R2 are routers. The dotted lines represent the SpaceWire links between the nodes and routers. N1 is time master and sends out a time-code whenever its TICK_IN signal is asserted. The numbers in the node and router boxes represent the current values of their time-counters.

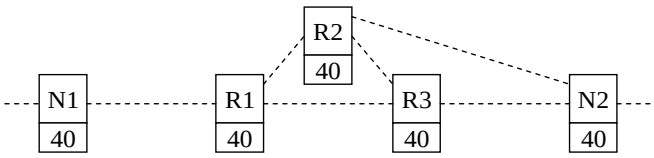

Fig. 2 Initial state of network

When TICK_IN is asserted in node N1, its time-counter is incremented from 40 to 41 and a time-code with time value 40 is transmitted. This is shown in Fig. 3.

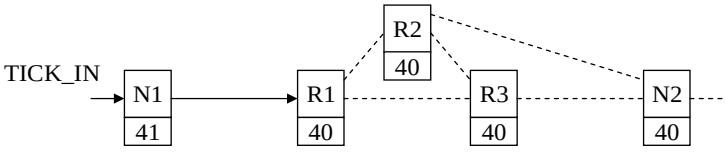

Fig. 3 TICK_IN asserted

The transmitted time-code is received by router R1 which checks that the time value is one more than current time counter value, and then increments its time-counter and sends out time-code on all its other links. See Fig. 4.

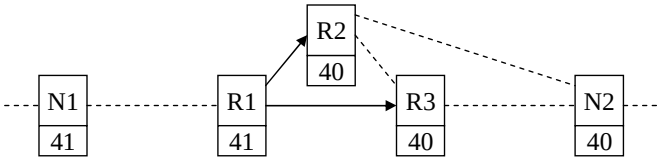

Fig. 4 Router R1 forwards time-codes

Routers R2 and R3 both receive time-codes sent from R1 and both of these routers send out time-code on all other links as shown in Fig. 5.

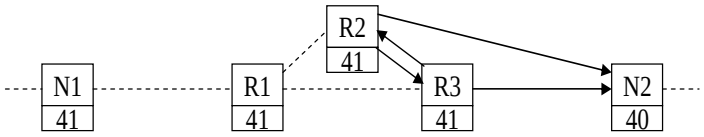

Fig. 5 Routers R2 and R3 forward time-codes

Node N2 receives the time-code from R3 and validates that the incoming time-code has a time value of one more than the node's time-counter. It then increments the time-counter and asserts TICK-OUT. See Fig. 6.

Time in N2 is updated.

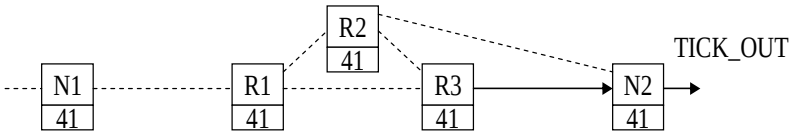

Fig. 6 N2 receives time-code and give TICK_OUT

R3 will also receive a time-code from R2, which is now not one more than R3's time-counter value, so is ignored as shown in Fig. 7. This prevents time-codes from going back through the network. Time-codes move forward through the network even when there are loops in the network.

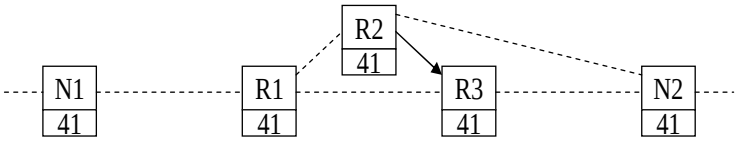

Fig. 7 Time-codes cannot go backwards

Node N2 will also receive a time-code from R2 which is not one more than N2's time-counter value, so is ignored. This prevents multiple triggering of the TICK_OUT signal as duplicate copies of the time-code which have taken different paths through the SpaceWire network arrive at the node. See Fig. 8.

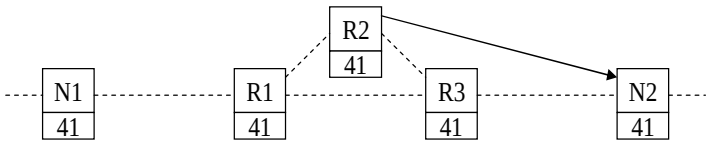

Fig. 8 Other time-codes received at N2 are not valid

LOST TIME-CODES

If a received time-code is not one more than (modulo 64) to the current time-count at the receiving link-interface, then either the time-code or the time-count shall be considered invalid. This can happen if a time-code is lost, or if a link is reset or restarted after a disconnect.

If the time-code is invalid then the time-count is updated to the new value but the time-code is not propagated in a router and TICK_OUT is not asserted in a node. This prevents propagation of invalid time-codes across a network. When the next time-code is received it is expected that the time-counter matches the time-code and normal operation resumes. Recovery from missing or invalid time-codes will now be considered.

Fig. 9 shows a SpaceWire network in which a time-code with a time-value of 20 is lost between R1 and R2.

Fig. 9 Lost time-code

On the next tick N1 sends out the time-code 21. R1 then forwards this time-code to R2. This is not same as, nor one more than time-counter of R2 so R2 updates its time-counter but does not emit the time-code, as shown in Fig. 10.

Fig. 10 R2 time-counter updated

On the next tick the time-code 22 is sent from N1 to R1, which forwards it on to R2. At R2 the time-count is now 21, so the incoming time-code is one more than the time-count hence the time-code is now valid and is propagated by the router and reaches N2. See Fig. 11. When the time-code reaches N2 it is not one more than N2's time-count (value 19) so the time-code is deemed invalid. N2 updates its time count to 22 but does not give a TICK_OUT.

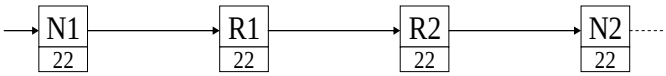

Fig. 11 N2 time-counter updated

The next tick will result in the time-code 23 propagating across the network and N2 will produce a TICK_OUT, as shown in Fig. 12.

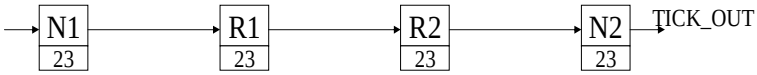

Fig. 12 N2 gets valid time-code

It takes several ticks to recover from initial error, depending upon the size of the network.

Note that if there is an alternative path from R1 to R2 the time-code may propagate successfully through the alternative path so that R2 gets a valid time-code even though one of the time-codes on its way to R2 gets lost. This provides a first level of fault tolerance for time-code distribution.

Nodes using the time-code distribution function can either use the TICK_OUT signal as a periodic timing signal or use the value of the time-count as an indication of the least-significant 6-bits of system time.

As a missing tick results in a timing discrepancy, the TICK_OUT signal should not be used to increment a counter with the expectation that this counter always corresponds to the system time. Rather a time-lock technique should be used where a free running local time-counter is updated to be an exact multiple of the system tick rate every time the

TICK_OUT signal is asserted. The reason for this is that when using the TICK_OUT signal as a periodic timing signal the time-code can be missed so that a TICK_OUT signal is missed. Having said this, SpaceWire signals running over 10m SpaceWire cable have a good eye diagram and are unlikely to give rise to any errors.

TIME-CODE LATENCY

The accuracy with which system time can be distributed is dependent upon the number of links over which it is distributed and the operating rate of each of those links. A delay of at least 14 bit-periods (ESC + data character = 4 + 10 bits) is encountered for each link that the Time-Code traverses, due to the time taken for each link-interface on the way to receive a Time-Code. This gives rise to a time-skew across a network of Tskew = 14.S/A where S is the number of SpaceWire links traversed and A is the average link operating-rate. Jitter is also introduced at each link interface due to the variation in time spent waiting for the transmitter to finish transmitting the current character or control code. At each link interface a delay of 0 to 10 bit-periods can be encountered. Across a network, this gives rise to a total jitter of Tjitter = 10.S/A. For an average rate of 100 Mbit/s and 10 links traversed, the time skew is 1.4 μs and the jitter 1.0 μs. The skew and jitter may be higher than indicated above depending on the implementation of the link-interface. A time accuracy across a network of significantly better than 10 μs may be difficult to achieve.

TIME-CODE APPLICATIONS

Time distribution

With the provision of this basic time distribution function, application level protocols can be used to distribute specific time values at full resolution (not just 6-bits) and to issue time dependent commands etc. The two control flags that are distributed with the 6-bit time code can be used to broadcast information to all nodes and routers on the network.

Nodes using the system time distribution function can either use the TICK_OUT signal as a periodic timing signal or use the value of the time-count as an indication of the least-significant 6-bits of system time.

Event signalling across a point-to-point link

Time-codes can also be used to signal events or to pass a high-priority byte of information across a point-to-point link. The time-code is sent transparently in the middle of the packet currently being transmitted. There is no need to terminate the current packet or to wait for its transmission to complete before sending the time-code. This technique should not be used when there are routers in the SpaceWire network.

Isochronous Communication

SpaceWire is fundamentally an asynchronous, packet switching network. However, using time-codes and the priority mechanism implemented in the SpaceWire routers, it is possible to provide an isochronous communication service.

Before describing the isochronous communication technique it is first necessary to recall how the priority mechanism works in a SpaceWire network. Priority is assigned in the SpaceWire router routing tables. When the routing tables are programmed, certain logical addresses are given high-priority while the others all have low-priority. If a destination node is to be able to receive both high and low priority packets then it must be given two separate addresses: one high-priority and the other low-priority. Nodes that wish to send this destination a packet will use one logical address for the low-priority packets and another logical address for the high-priority packets. When a packet arrives at a router with a particular logical address, the router output port(s) to use for that address are read from the routing table. At the same time the priority to give the packet is also read from the routing table. Each logical address has a priority level specified in the routing table (high or low).

When isochronous communication is required the data for asynchronous transfer is assigned low priority and can be sent by a node at any time. Data that has to be transferred isochronously is sent with high-priority but can only be sent at specific times.

Every time a time-code is sent out across the network, one node gets a chance to send an isochronous packet, i.e. a packet with a high-priority logical address. Since there are 64 possible values of time in the time-code, there are 64 possible slots for sending isochronous, high-priority packets. Nodes are assigned zero, one or more of these slots for sending data isochronously. When a node receives a time-code with a value corresponding to one of its allocated

isochronous slots, it can send out one packet with a high-priority logical address. This packet will propagate across the network, possibly waiting at each router to gain access to the output port that it needs to use, if that output port is currently being used to send a packet.

There is a possible delay, waiting for the current packet to finish being sent, at each router. The worst-case delay will depend upon the maximum size of the packet that can be transferred across a SpaceWire link and the rate of transfer. Since SpaceWire allows arbitrary length packets to be transferred with no upper bound on their length it is necessary to introduce a maximum packet length for all transfers both asynchronous and isochronous. If this maximum packet length is L and the effective data rate (approximately 80% of the transmission speed) is D then the maximum time that an isochronous, high-priority packet will have to wait at a router is P.

$$P = L/D.$$

Single Router Network

If the SpaceWire network comprises a single router (or redundant pair of routers) connected to a number of nodes then the maximum delay for an isochronous packet is simply the time spent waiting for a transmission slot (Tslot), plus the time spent waiting for the current packet being sent by the router to complete its transfer (P), plus the time to send the isochronous packet (I).

$$T = Tslot + P + I \quad = Tslot + L/D + L/D \quad = Tslot + 2L/D$$

The wait for the current packet to complete is only necessary if the router output port to the destination is currently being used.

The worst case situation is actually a little worse than this. It could be that the source node that wants to send the isochronous packet had just started to send an asynchronous packet to the same destination that the isochronous packet is to be transferred to. This asynchronous packet itself has to wait for any current packet from another node that is heading for the same destination to complete transmission. In this case the worst-case delay is

$$T = Tslot + (R-1)\ P + I \quad = Tslot + R.L/D$$

where R is the number of ports on the router.

The time spent waiting for an isochronous time slot (Tslot) is dependent upon the scheduling table that allocates time-slots to isochronous nodes. There are 64 time slots which can be allocated. One source node can be given several time slots if its required isochronous latency and bandwidth demand it.

Alternatively the available isochronous communication bandwidth can be shared between the nodes by allocating each node a certain number of packets (or bytes) that it can transmit in each and every time slot. In the single router case the maximum latency is then

$$T = Tslot + (R-1)P + NI \quad = Tslot + (N+R-1)L/D$$

where N is the total number of packets that can be sent in a single time slot summed over all nodes and N.L is the maximum total number of bytes that could possibly be sent by the nodes. The total number of bytes sent isochronously by all the nodes must be less than the total allocated bandwidth for the time slot and must be significantly less than the bandwidth available in between time slots i.e. the isochronous communications must not hog all the bandwidth or asynchronous communications will not get the opportunity to use the network.

For an eight node network with a single 8-port router running at 160 Mbits/s (200 Mbits/s baud rate) and a maximum packet length of 2000 bytes, the worst-case time delay is T = Tslot + 800 µs, for a single isochronous packet sent in a time slot.

Assuming that each node is given equal access to the isochronous slots which occur every 1 ms, the worst case delay is then T = 1 + 1.5 ms = 2.5 ms. With the same network and the same data-rate and packet size, if each node is allowed to transmit one isochronous packet at each time slot the maximum total time to transfer all the isochronous traffic is 800 µs, so a time slot interval of 2 ms may be necessary to give adequate bandwidth for the asynchronous traffic. In this case, however, the worst-case time delay is T = 2 + 1.5 ms = 3.5 ms.

Two Router Deep Network

If the SpaceWire network contains routes that pass through two routers between nodes then the situation for isochronous communications is rather different. It could be that the router closest to the destination is very busy and asynchronous packets are queued up on all its inputs waiting for the output port that goes to a particular destination. In the other router, closest to the source node, the situation is as before. Now in the worst case, the router at the destination end will give round-robin priority to the waiting asynchronous packets and service the input from the router near the source node last. That is a total delay at the router near the destination of

$$T_{dest} = (R-1)P$$

where R is the number of ports in the router. This delay occurs for every link except one on the second router giving a total delay of

$$T = T_{slot} + ((R-1)(R-1) + N)L/D$$

If each of the routers is an 8-port router then using the same data-rate and packet size as in the previous example, the total time delay is $T = 10 + (7x7 + 1).L/D = 10 + 5$ ms = 15 ms for the case of one packet per time slot and 10 ms time slot interval.

Many Router Deep Network

If the SpaceWire network has routes with three or more routers between nodes then the worst-case delay multiplies

$$T = T_{src} + 2.T_{dest} = T_{slot} + (N + (R-1)^{Q}).L/D$$

where Q is the number of routers on a path. For large networks the time delay rapidly becomes very large. However, if the isochronous traffic is restricted to paths with one, two or three routers then the worst-case delays may be acceptable.

Destination Stalling

The isochronous communications mechanism requires that packets are not allowed to stall i.e. the destination node must be ready to receive the packet and not cause the packet to wait while buffer space is freed in the destination node. This implies that no packet should be sent without buffer space in the destination node being reserved for it, otherwise the destination will stall and the network will get blocked unnecessarily. This gives rise to the need for a SpaceWire Transport layer which handles end-to-end packet flow control [4].

SUMMARY

SpaceWire time-codes extend the communication capabilities of a SpaceWire network considerably, enabling the high accuracy distribution of time and the broadcast of two-bits of control information. They also enable the construction of isochronous communication mechanisms within SpaceWire.

REFERENCES

[1] S.M. Parkes et al, "SpaceWire: Links, Nodes, Routers and Networks," *European Cooperation for Space Standardization, Standard No. ECSS-E50-12A,Issue 1, January 2003.*

[2] S.M. Parkes and J. Rosello "SpaceWire ECSS-E50-12A," *International SpaceWire Seminar, ESTEC Noordwijk, The Netherlands, November 2003*

[3] S.M. Parkes, C. McClements, G. Kempf, S. Fischer and A. Leon, "SpaceWire Router," *International SpaceWire Seminar, ESTEC Noordwijk, The Netherlands, November 2003*

[4] S. Mills and S.M. Parkes, "The SpaceWire Transport Protocol," *International SpaceWire Seminar, ESTEC Noordwijk, The Netherlands, November 2003*