High Accuracy Time Synchronization over SpaceWire Networks

Final Report

SPWCUC-REP-0004
Version 1.0
18 December 2013

Final Report

ESTEC Contract 400105931

EUROPEAN SPACE AGENCY CONTRACT REPORT

The work described in this report was done under ESA contract.
Responsibility for the contents resides in the organisation that prepared it.

Aeroflex Gaisler AB
Kungstatan 12
411 19 Göteborg
Sweden

www.aeroflex.com/gaisler

Table of contents

1 INTRODUCTION

1.1 Scope

This document establishes the Final Report for the *High Accuracy Time Synchronization over SpaceWire Networks* activity initiated by the European Space Agency.

1.2 Overview

This document is partitioned in the following main sections:

- Activity description
- Study logic
- Time Distribution Protocol
- Latency, Jitter and Drift mitigation
- Conclusions

1.3 Referenced documents

[1] Request for Full Proposal: RFQ 3-13363/11/NL/LvH – High Accuracy Time Synchronization over SpaceWire Networks, ESA

[2] SpaceWire – Links, Nodes Routers and Networks, ECSS-E-ST-50-12C

[3] SpaceWire Protocol Identification, ECSS-E-ST-50-51C

[4] Remote Memory Access Protocol (RMAP), ECSS-E-ST-50-52C

[5] GRLIB IP Library User's Manual, Aeroflex Gaisler, *http://www.gaisler.com/products/grlib/grlib.pdf*

[6] GRLIB IP Core User's Manual, Aeroflex Gaisler, *http://www.gaisler.com/products/grlib/grip.pdf*

[7] AMBA Specification, Rev 2.0, ARM IHI 0011A, ARM Limited

[8] Problem formulation: Jitter and drift of Time-Codes in SpaceWire networks, SPWCUC-REP-0002

[9] Time Distribution Protocol, SPWCUC-REP-0003

[10] S. Parkes, "The Operation and Uses of the SpaceWire Time-Code", *International SpaceWire Seminar (ISWS 2003)*

[11] B. van Leeuwen et al., "SpaceWire Network Simulation of System Time Precision", *International SpaceWire Conference 2011*

[12] F. Siegle, "Time-Code Jitter Measurements", ESA

[13] M. Suess et al., "SpaceWire Time Code Latency and Jitter", 5th International SpaceWire Conference 2013

[14] CCSDS 301.0-B-4 - Time Code Formats, Blue Book

[15] CCSDS IP Cores User's Manual, Aeroflex Gaisler, http://www.gaisler.com/doc/tmtc.pdf

1.4 Terms, definitions and abbreviated terms

AMBA	Advanced Microcontroller Bus Architecture
ASCII	CCSDS ASCII Calendar Segmented Code (ASCII)
CCS	CCSDS Calendar Segmented Time Code
CCSDS	Consultative Committee for Space Data Systems
CDS	CCSDS Day Segmented Time Code
CUC	CCSDS Unsegmented Time Code
CUCTP	SpaceWire - CCSDS Unsegmented Time Code Transfer Protocol
ECSS	European Cooperation on Space Standardization
ESA	European Space Agency
GRLIB	Aeroflex Gaisler's IP core Library
LSB	least significant bit
MSB	most significant bit
OBDH	On-Board Data Handling (legacy bus)
P-Field	preamble field
RASTA	Reference Avionics System Testbench Activity
RMAP	Remote Memory Access Protocol
SOW	Statement of Work
SPW	SpaceWire
SPWCUC	SpaceWire CUCTP IP core
T-Field	time field
TAI	International Atomic Time
TDP	SpaceWire - Time Distribution Protocol
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit

2 ACTIVITY DESCRIPTION

2.1 Introduction to the activity

The SpaceWire [2] concept is a network for space applications composed of nodes and routers interconnected through bi-directional high-speed digital serial links. The technology has grown organically from the needs of on-board processing applications. Starting off from simple point-to-point interconnections, the SpaceWire concept has grown into a complete network featuring routers and redundancy concepts. The SpaceWire network is made up of a number of nodes. The nodes can either be interconnected by means of point to point links, or via routing switches. Each node is the source or destination for SpaceWire packets.

Time synchronization in spacecraft is becoming increasingly important. For example instruments and navigational on-board resources can now be combined for establishing scientific observations and therefore need to be well synchronized in time.

Traditionally time synchronization has been done via dedicated signals or via deterministic on-board buses (e.g. MIL-STD-1553 or OBDH). With the advent of SpaceWire point-to-point links and router switches being used for critical control functions, the need for accurate time synchronization via this network has arisen.

The SpaceWire protocol provides rudimentary time-code transmission, but lacks support for automatic time message distribution and time synchronization. It has no means for handling latency (delays) and jitter caused by routing or drift caused by unstable oscillators.

This work supports communication in real-time constrained applications such as spacecraft navigation or control. The work supports the incoming SpaceWire-D (D as in Deterministic) standard, which relies on the availability of time synchronization in a SpaceWire network. This work has been based upon existing knowledge about SpaceWire nodes and networks.

The aim has been to investigate the feasibility and assess the performance of novel techniques to cope with the aforementioned issues. This has been done by means of analysis and experimental verification, as well as upgrading existing equipment for ESA’s RASTA avionics testbed in order to facilitate high-level prototyping and software development.

Accurate time synchronization through SpaceWire should enable and promote the use of SpaceWire for critical control functions onboard spacecraft. It will also allow unification (i.e. reduction of the number of on-board buses actually required) of the on-board bus systems.

2.2 Relation with other activities

Aeroflex Gaisler has previously developed in collaboration with the European Space Agency (ESA) an initial protocol for the transmission and synchronization of CCSDS Unsegmented Code (CUC) [14] time in SpaceWire networks. The working name of the protocol is "SpaceWire - CCSDS Unsegmented Code Transfer Protocol" (CUCTP).

Aeroflex Gaisler has also previously developed under indirect funding from the ESA a new IP core that implements the SpaceWire - CCSDS Unsegmented Code Transfer Protocol named SPWCUC [15], providing automatic SpaceWire Time-Code transmission and reception, and automatic CUCTP packet reception. It also provides support for CUCTP packet transmission.

The previous CUCTP protocol and its implementation was a first iteration to solve some of the time distribution problems that exist in SpaceWire networks. Additional work has been performed both on the specification side as well as on the implementation side to allow an ECSS standard protocol to be established in the future.

In this activity the new SpaceWire - Time Distribution Protocol (TDP) has been developed that replaces the SpaceWire - CCSDS Unsegmented Code Transfer Protocol (CUCTP).

2.3 Main project goals and challenges

According to the work statement [1], the objectives of the proposed activity have been to:

- Establish a time message distribution mechanism over SpaceWire
- Establish an offset correction mechanism between local times which is correcting for the time distribution latency via SpaceWire
- Establish a method for clock synchronisation by correcting the drift between the local clocks in a SpaceWire network
- Updated the existing RASTA equipment with the above time distribution, offset correction as well as clock synchronisation methods

The results from the activity have been the following:

- Well defined and proven technology (or methods) for highly accurate time distribution and synchronization over SpaceWire networks.
- A prototype implementation (VHDL code in non-flight quality) to future soft cores implementing the above technology in hardware, as well as initial software code (C-code in non-flight quality) for part of the algorithms that are better handled in software.
- Preparation of a future development in which additional new buses or existing buses are harmonized using the same above technology.

The targeted applications for the activity have been the following:

- SpaceWire networks for critical on-board control applications where current on-board buses such as Mil-Std-1553 and OBDH can be replaced. For this type of applications the time accuracy is important to allow implementation of isochronous communication over the inherently asynchronous SpaceWire network.
- SpaceWire networks for science payloads where time synchronization is an important factor. An example could be multiple distributed sensors in an antenna that communicate via SpaceWire and need by synchronized for coherent measurements, or when two instruments exchange data to correlate their results.
- The above applications also require ground test equipment to support time distribution throughout the development cycle.

2.4 Main implementation steps and schedule

The main implementation steps that have been performed consist of the following tasks:

- Time distribution method and specification
- Time distribution accuracy methods (latency, jitter and drift)
- Time distribution hardware implementation (synthesizable VHDL core)
- RASTA avionics testbed update
- RASTA software drivers and demonstration

The activity was originally proposed as a 4 month development, but was extended due to technical challenges with the jitter mitigation techniques and to accommodate feedback from the SpaceWire community. The activity was further extended to include the upgrading of multiple existing RASTA units with the latest available IP cores from Aeroflex Gaisler. The activity took in overall 26 months to complete.

The following milestones and corresponding review meetings have been held:

• Kick-off Meeting	2011-10
• Progress Meeting	2012-09
• Final Review Meeting	2013-12

2.5 Main output

The following items have been delivered within the activity:

TN-1	Time Distribution Specification
TN-2	VHDL IP core User’s Manual
TN-3	Updated RASTA User’s Manuals
FR	Final Report
ABS	Abstract (5th International SpaceWire Conference 2013 paper)

SW-1	VHDL IP core model
SW-2	FPGA programming bit files for RASTA boards
SW-3	Driver software (part of RTEMS on-line distribution)
SW-4	Protocol software, i.e. RMAP software stack (part of RTEMS on-line distribution)
SW-5	Test application software (part of RTEMS on-line distribution)

2.6 Main results

The following products have been developed and will be commercialized as a result of the work performed in this activity:

- SpaceWire - Time Distribution Protocol VHDL IP core and user’s manual
- RASTA FPGA programming files and user’s manuals
- RTEMS Cross Compilation System (RCC) driver updates (free of charge)

3 STUDY LOGIC

The development flow of the subject activity has been in line with the work statement, following a straight stream from definition and specification, design of hardware and software.

The work logic for each of the tasks that have been performed is presented in the following sections. These present an overview of the activities in each task objective. These are based on the statement of work [1]. The overall activity was contained in a single phase.

The work logic included the following main tasks, as further detailed in the remainder of the section:

- Develop a method for time distribution, offset correction and clock synchronisation
- Implementation of time distribution, offset correction and clock synchronisation
- Upgrade RASTA FPGA designs with time distribution, offset correction and clock synchronisation

3.1 Establish specification for time distribution & clock synchronisation

In this work package, the requirements for time distribution in SpaceWire networks have been established, based on trade-offs between different time message distribution protocols, characterisation of latency and jitter in current SpaceWire networks, investigation and benchmarking of different methods for time distribution latency correction, and finally investigation and benchmarking of different methods for time distribution jitter correction and local oscillator offset and drift correction.

The latency and jitter of SpaceWire Time-Codes were measured suing different equipment such as SpaceWire nodes and routers [8]. Amongst others the following equipment was used during the analysis:

- GRSPW2 IP core, 50 MHz local clock frequency
- GRSPWROUTER IP core, 50 MHz local clock frequency
- AT7910 ASIC, 30 MHz local clock frequency

Results from independent measurements conducted by ESA [12], [13] were also analysed in this activity. The actual measurement data were used as stimuli to verify the mitigation concepts and were also used in the VHDL IP core verification described in the next task.

The objective is to reduce the impact of jitter between the time events (i.e. the reception of Time-Code) in a remote node (i.e. Target), as well as to adapt to the frequency drift or wander between the *local clocks* in the originating (i.e. Initiator) and terminating nodes (i.e. Targets).

The instant jitter can be quantified by measuring the *Time-Code interval* [T'_{TICK}] between two consecutive time events in the terminating node using a *local clock*. The resolution of a single measurement would be limited by the period [T_{SYS}] of the *local clock*. The above measured *Time-Code intervals* [T'_{TICK}] can be stored locally in the terminating node, to build up statistics of the jitter.

Assuming that there is no frequency drift or wander, the *Time-Code interval* [T_{TICK}] of the originating node can be estimated by averaging *Time-Code intervals* [T'_{TICK}] as measured by the *local clock* in a terminating node. Based on this estimate an internally generated signal could be generated such that all events on this signal occur without any jitter. The resolution of the adjustment would be that of the local clock period [T_{SYS}].

Assuming that there would be no jitter present on the received SpaceWire Time-Codes, then the frequency drift or wander could be estimated comparing these time event to a locally generated signal which is based on the real-time clock keeping in the terminating node. The *real-time clock* in the terminating node can then be adjusted such that the locally generated signal follows the incoming SpaceWire Time-Code events. The resolution of the *real-time clock* adjustment would that of the local clock period [T_{SYS}].

Combining the two above, adjustment of both jitter and frequency drift or wander could be achieved. To summarize, to make the problem formulation simple, the *local clock* is used for:

- generating the *real-time clock*
- measuring the *time interval* between received Time-Codes
- correcting Time-Code jitter
- detecting and correcting frequency drift/wander

This task was concluded with a Progress Meeting.

3.2 Implementation of time distribution

In this work package, the time distribution specification has been implemented as a synthesizable VHDL model which has been verified by means of simulation. A user’s manual has been established for the VHDL model (also referred to as VHDL IP core).

The implementation of the jitter and drift mitigation is based on a simple time interval measurement of the incoming SpaceWire Time-Codes using the *local clock*, gathering statistical information which is then used to calculate an average correction value that is applied to a locally generated signal which is free from jitter and local drift. The variance of the corrected locally generated signal is limited to one period of the *local clock*.

A VHDL test bench was developed to verify the functionality of the VHDL IP core. Emphasis has been placed on the following areas:

- CCSDS Time Code transmission using RMAP [4] commands in SpaceWire packets
- Synchronization via Time-Codes
- Time-stamping of reception and transmission of Distributed Interrupts (as well as Time-Codes)
- Latency offset adjustment
- Jitter mitigation
- Drift mitigation

During simulation the real-world data collected during the independent ESA measurements [12] have been used as stimuli to validate the jitter and drift mitigation technique implemented in the VHDL IP core.
FPGA based rapid prototyping has been used during the development of the VHDL IP core. The VHDL IP core has been integrated with a LEON3 32-bit SPARC processor in a system-on-chip design to facilitate early validation of the IP core with software in the loop.

3.3 Upgrade of RASTA designs with time distribution

In this work package, the RASTA FPGA designs have been upgraded to include new VHDL IP core.

The following additional modifications have been performed:

- transition to the GRSPW2 IP core
- transition to the GRPCI2 IP core
- update of GR1553B IP core to the latest version
- update of PacketWire IP core to the latest version
- inclusion of the new SpaceWire - Time Distribution Protocol IP core

FPGA programming bit files for the following RASTA systems have been developed:

- RASTA-IO board for RASTA-101 (AT697 over PCI)
- RASTA-IO/CPU board for RASTA-105 (LEON3 integrated)
- RASTA-IO/CPU board for RASTA-106 (LEON2 integrated)
- RASTA-TMTC board

The RTEMS drivers have been extended to support the new VHDL IP cores.

Test software to execute the time protocol has been developed, both for the master in the system and the slaves. The example test application demonstrates the operation of the VHDL IP core and the software protocol (RMAP [4] software stack) and RTEMS drivers.

The RASTA FPGA and software has been verified in an in-house RASTA environment at Aeroflex Gaisler.

A final report and abstract have been established. The task was concluded with a Final Review conducted off-line.

4 TIME DISTRIBUTION PROTOCOL

The aim of the Time Distribution Protocol (TDP) is to synchronize time across a SpaceWire network. It does this by an initiator writing a CCSDS Time Code using an RMAP [4] command placed in a SpaceWire packet, transferring it across the SpaceWire network and then extracting the CCSDS Time Code at the target, and by means of SpaceWire time control codes (Time-Codes) used to convey the time instant at which the CCSDS Time Code becomes valid (synchronization event).

4.1 Protocol

The Time Distribution Protocol (TDP) provides the capability to transfer CCSDS Time Codes (i.e. time message) between onboard users of a SpaceWire network. The CCSDS Time Codes may be of variable length or fixed size at the discretion of the user and may be submitted for transmission at variable time intervals, providing a communication service.

The Time Distribution Protocol provides the capability to synchronize nodes in a SpaceWire network by using SpaceWire time control codes (Time-Codes), providing a timing service.
An Initiator is a SpaceWire node distributing CCSDS Time Codes and SpaceWire time-control codes (Time-Code). An Initiator is also an RMAP initiator, capable of transmitting RMAP commands and receiving RMAP replies. There is only one active Initiator in a SpaceWire network during a mission phase.

A Target is a SpaceWire node receiving CCSDS Time Codes and SpaceWire time-control codes (Time-Codes). A Target is also an RMAP target, capable of receiving RMAP commands and transmitting RMAP replies. There can be one or more Targets in a SpaceWire network.

The protocol also provides means for time-stamping of incoming and outgoing Distributed Interrupts in the Target and makes this information accessible to an Initiator by means of RMAP accesses. Note that Distributed Interrupts are currently being defined in the draft ECSS-E-ST-50-12C Rev.1 standard.

The protocol also provides means for transferring latency correction information (which can be calculated from the above time-stamp information) from an Initiator to a Target by means of RMAP accesses.

4.2 Operation

The Initiator and the Target maintain their own time locally, for which the implementation is independent of this standard. The Time Distribution Protocol provides the means for transferring the time of the Initiator to the Targets, and for providing a synchronization point in time.

The time is transferred by means of an RMAP write command carrying a CCSDS Time Code. The synchronization event is signalled by means of transferring a SpaceWire time control code (Time-Code). The transfer of the SpaceWire Time-Code is synchronized with the time maintained by the Initiator.

To distinguish which SpaceWire Time-Code is to be used for synchronization, the value of the SpaceWire Time-Code is transferred from the Initiator to the Target by means of an RMAP write command prior to the actual transmission of the SpaceWire Time-Code itself.

When there is more than a one Target, the CCSDS Time Code need be transferred to each individual Target separately (unless SpaceWire packet broadcast or multicast can be used). Only one transmission of the SpaceWire Time-Code is however need, although one can imagine systems where different SpaceWire Time-Code values are used for different Targets.

4.3 Services

The Time Distribution Protocol provides users with *communication services* based on RMAP service primitives and parameters, transferring amongst others CCSDS Time Codes.

The Time Distribution Protocol provides users with *timing service* based on SpaceWire time-control codes (Time-Codes).

The followings services are defined in this Standard:

- Configuration
- Status
- Command (CCSDS Time Code)
- Datation
- Timing (Initialisation/Synchronization)
- Time-Stamp (of SpaceWire time-control codes (Time-Codes))
- Latency

4.4 Publications

As an outcome of this activity a draft protocol specification has been established [9] and presented at various conferences and meetings:

- 4th International SpaceWire Conference, 2011
- 17th SpaceWire Working Group Meeting, 2011
- 18th SpaceWire Working Group Meeting, 2012
- 20th SpaceWire Working Group Meeting, 2013
- 5th International SpaceWire Conference, 2013

5 LATENCY, JITTER AND DRIFT MITIGATION

Due to the natural latency of transferring time control codes (Time-Codes) in a SpaceWire network, the clock state correction accuracy discussed in [11] will be limited by an offset difference between the initiator and the target (or between targets).

The proposed TPD protocol utilizes the new Distributed Interrupts defined in the draft ECSS-E-ST-50-12C Rev.1 standard which are distributed using similar methods as time-control codes and can therefore be used a means for measuring the propagation delay of the latter. Each Target can be configured by the Initiator to perform a time-stamp whenever a Distributed Interrupt with a specified value has been sent or received. The time-stamping is done with the time that is maintained by the Target.

The Initiator performs similar time-stamping at its end and then uses the time-stamps in both ends to calculate the latency or (propagation delay) in either direction. The calculated latency can be written by the initiator to a specific Target register which can be used for correcting the time maintained in the Target.

The proposed protocol thus provides a means for measuring latency between an Initiator and a Target, and provides means for communicating the result to the Target. The methods for how to send and receive Distributed Interrupts, to perform measurements, and to realize the correction in the Target are left to the implementers.

Statistical methods and regulation techniques can be used to mitigate the jitter seen on time control codes (Time-Codes) in a SpaceWire network, as discussed in [10]. Jitter and drift mitigation discussed hereafter could be combined.

Drift mitigation by means of clock rate correction [11] can be performed based on periodically received time control code (Time-Code) by a Target. The mean interval between received time control codes (Time-Codes) could be measured with the local time maintained by the Target as reference, and any long term variation could be fed back to the generation of this local time.

The proposed TDP protocol does not provide any means for jitter and drift mitigation, since this does not affect the protocol itself, being a problem to be solved locally in the Target. The draft specification could however be extended with implementation guidelines as part of an informative annex.

6 SPACEWIRE – TIME DISTRIBUTION PROTOCL (SPWTDP) IP CORE

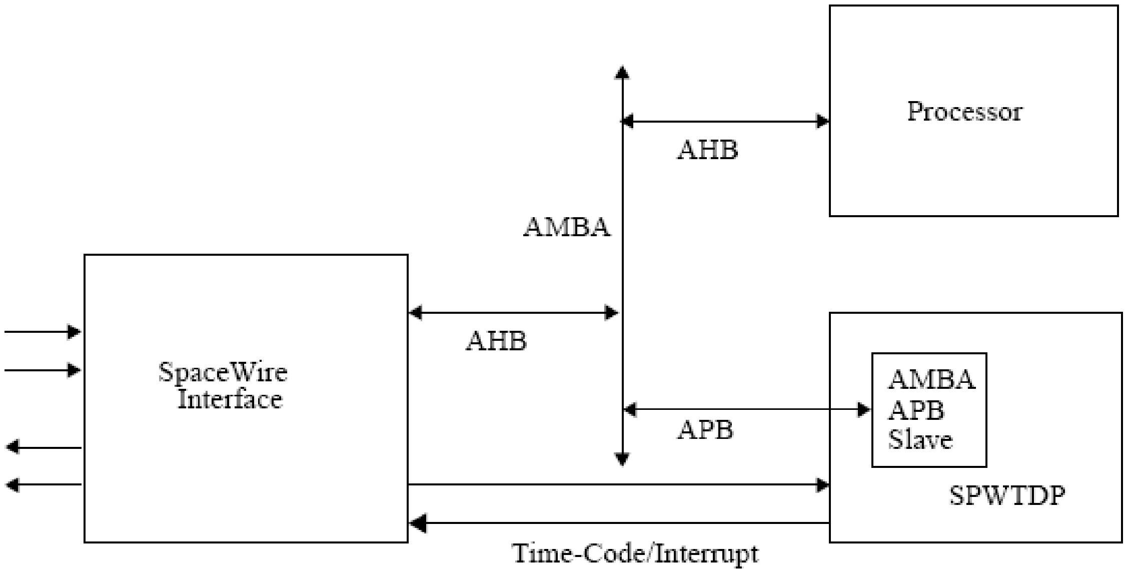
The VHDL IP core (named SPWTDP) that has been developed in this activity implements the draft Time Distribution Protocol (TDP). The IP core is interfaced via an AMBA Advanced Peripheral Bus (APB) slave interface, providing a register view that is compatible with the TDP. Since the TDP protocol is based on the RMAP protocol, it has been possible to reuse the RAMP target implementation in the Aeroflex Gaisler GRSPW2 SpaceWire node IP core and only implement the actual register view and functionality required for TDP.

Thus from the SpaceWire network side a device (e.g. RASTA) can be accessed through RMAP commands and RMAP replies, addressing the specific memory space on the AMBA bus where the SPWTDP IP core resides. This is compliant with the draft TDP protocol and is independent of Aeroflex Gaisler GRSPW2 implementation, since AMBA is the standardized on-chip bus used in most ESA activities. The SPWTDP IP core has one semi-custom interface towards the GRSPW2 IP core which allows receiving and sending SpaceWire Time-Codes and the new SpaceWire Distributed Interrupts. This interface is however very simple and easy to reproduce in other designs.

The SPWTDP IP core can both act as an Initiator and a Target as specified in the draft TDP protocol, being able to send and receive SpaceWire Time-Codes.

Note that the TDP Initiator requires that the SpaceWire link interface implements an RMAP Initiator, most commonly this is done in software.

Note that the TDP Target requires that the SpaceWire link interface implements an RMAP target either in hardware or in software.



7 CONCLUSIONS

This activity has resulted in a draft specification of a SpaceWire – Time Distribution Protocol. The daft protocol has been implemented as a VHDL IP core and integrated in the RASTA testbed environment. The IP core also supports rudimentary latency, jitter and drift mitigation.

The IP core will be included in future releases of the Aeroflex Gaisler GRLIB VHDL IP core library and in RASTA systems.

Information furnished by Aeroflex Gaisler is believed to be accurate and reliable. However, no responsibility is assumed by Aeroflex Gaisler for its use, or for any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Aeroflex Gaisler.

All information is provided as is. There is no warranty that it is correct or suitable for any purpose, neither implicit nor explicit.