SpaceWire *Features* and *SOIS* Services

Prof. ***Yuriy Sheynin***

St. Petersburg State University of Aerospace Instrumentation

Institute of High-Performance Computer and Network Technologies

Director, Doctor of Science

190 000 St. Petersburg
67, Bolshaya Morskaya str.
Fax: +7 812 3157778
E-mail: sheynin@online.ru

1. **PnP**

   Configuration space standardization for SpaceWire Nodes and Switches

2. **CCSDS SOIS Services**

   1. Low-latency real-time signals and interrupts in the Services at the Application Support Layer and at the SOIS Sub-Network Layer (a problem statement)

   2. CCSDS SOIS Services Prototyping,
      Plans in SOIS Services Prototyping and Analysis.

Network Notification Data Structures

- Plug-and-Play network service requires a unified data structures in network components
  - to unify network components identification procedures
  - to get important in network management procedures details on network components type, status

- PnP friendly architectures tend to standardize basic data structures

  (PCI Configuration space structure an example)

Router Identification

- Router Details:
  - Device Class=<*SpaceWire Router*> |
    <SpaceFibre-*SpW Router*> | …
  - Vendor ID
- Number of ports
- Router Parole = <Random Router identification code>
- Router Serial Number

Node Identification

- Node Details:
  - Device Class=<*SpaceWire Node*>
  - Device Subclass
  - Vendor ID
- Number of ports
- Node Serial Number

Data Structure in a Router

Router Configuration

- Ports' Status (unitary coding)
- Faulty ports (unitary coding)
- Terminal ports (unitary coding)
- Group adaptive routing (unitary coding)

Routing Table

Routes to NNM

- Serial identification NNM = <Node Identification> <Path Address Length > < Path Address>
- Regional NNM = <Node Identification> <Path Address Length >< Path Address>

- Port State (for each port)
  - Port Status
    - Run
    - Error
    - Terminal
    - GAR Include
    - Up (Un_Id)
    - Down (Un_Id)
    - …

- Port Connection (for each port)
  - <Port Connection>= <Own Port> <Neighbor port>
  - <Neighbor Unit Details> = <DeviceClass=Node/Router><Vendor ID>
  - <Neighbor Unit Parole> = Random identification code of Neighbor Unit
  - <Node Serial Number > Регистр GAR

Data Structure in a Node

- Port State (for each port)
  - Port Status
    - Run
    - Error
    - Terminal
    - GAR Include
    - Up (Un_Id)
    - Down (Un_Id)
    - …

Routes to NNM

a) <Serial identification NNM>= <Node Identification > < Path Address Length>< Path Address>

b) <Regional NNM List>=<List Length> {<Node Details> <Node Parole> <Node Serial Number > < Path Address Length >< Path Address>}*L

- Port Connection (for each port)
  - <Port Connection>= <Own Port> <Neighbor port>
  - <Neighbor Unit Details> = <DeviceClass=Node/Router><Vendor ID>
  - <Neighbor Unit Parole> = Random identification code of Neighbor Unit
  - <Node Serial Number > Регистр GAR

Low-latency real-time signals in CCSDS SOIS Services

- The Low-latency real-time signals - ***SpaceWire*** **Distributed Interrupts** possibilities are not presented at the Application Support Layer and at the SOIS Sub-Network Layer
- Interrupts as an example of low latency real-time signals are mentioned only in the **Cmd & Data Acquisition Services**

  " The DDPS (Device Data Pooling Service) will periodically sample the devices at a determined sampling rate **or cache state from devices that generate interrupts."**

  (page 30 in "Spacecraft onboard interface services// Draft informational report 850.0-G-0b ")
- To have an access to the low latency real-time signals distribution feature of the SpaceWire networks correspondent services and functions should be specified at the SOIS Application Support Layer and at the SOIS Sub-Network Layer

SOIS Services Analysis and Prototyping

Services of interest

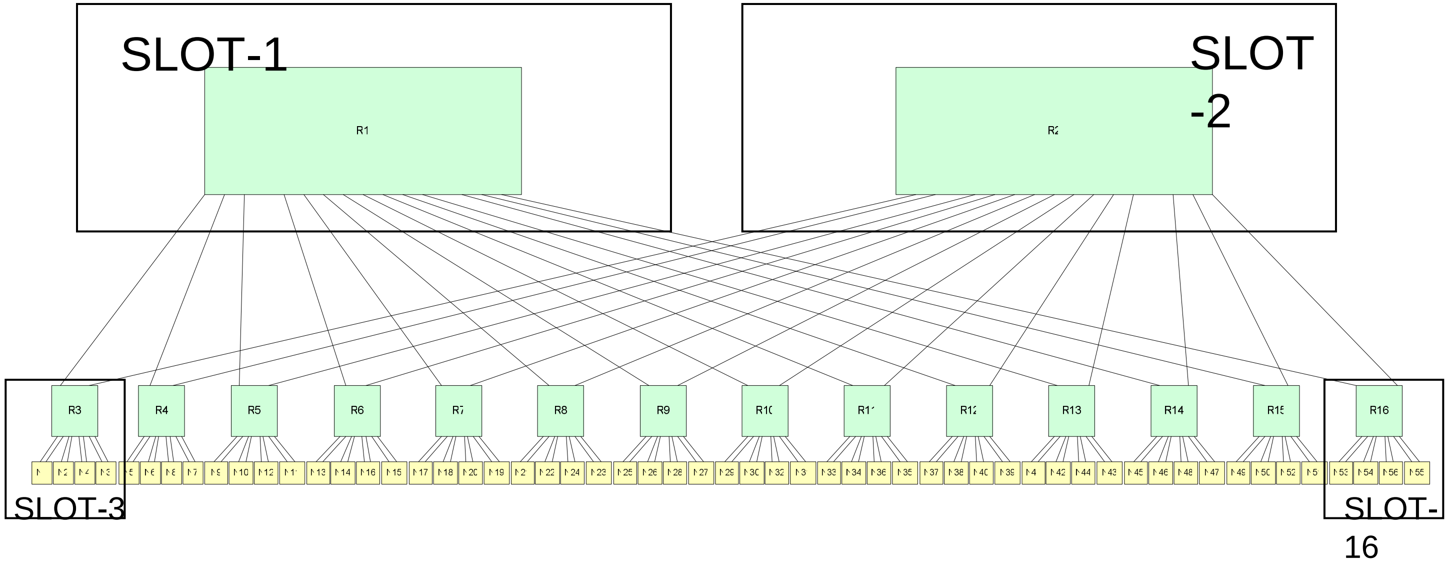
- Interrupts Distribution Services
- Time Distribution Services
- QoS Packet Services

Analysis and prototyping by

- Analytical models
- Simulation models
- Prototyping with HW modules

CCSDS SOIS Services Prototyping

- Configurable SpaceWire network models
  - Written in SystemC
  - Each type of network elements (node, link and routing switch) is implemented as an independent module. A network of required topology is be composed from this modules.
- Visual Network Assembling in Visio

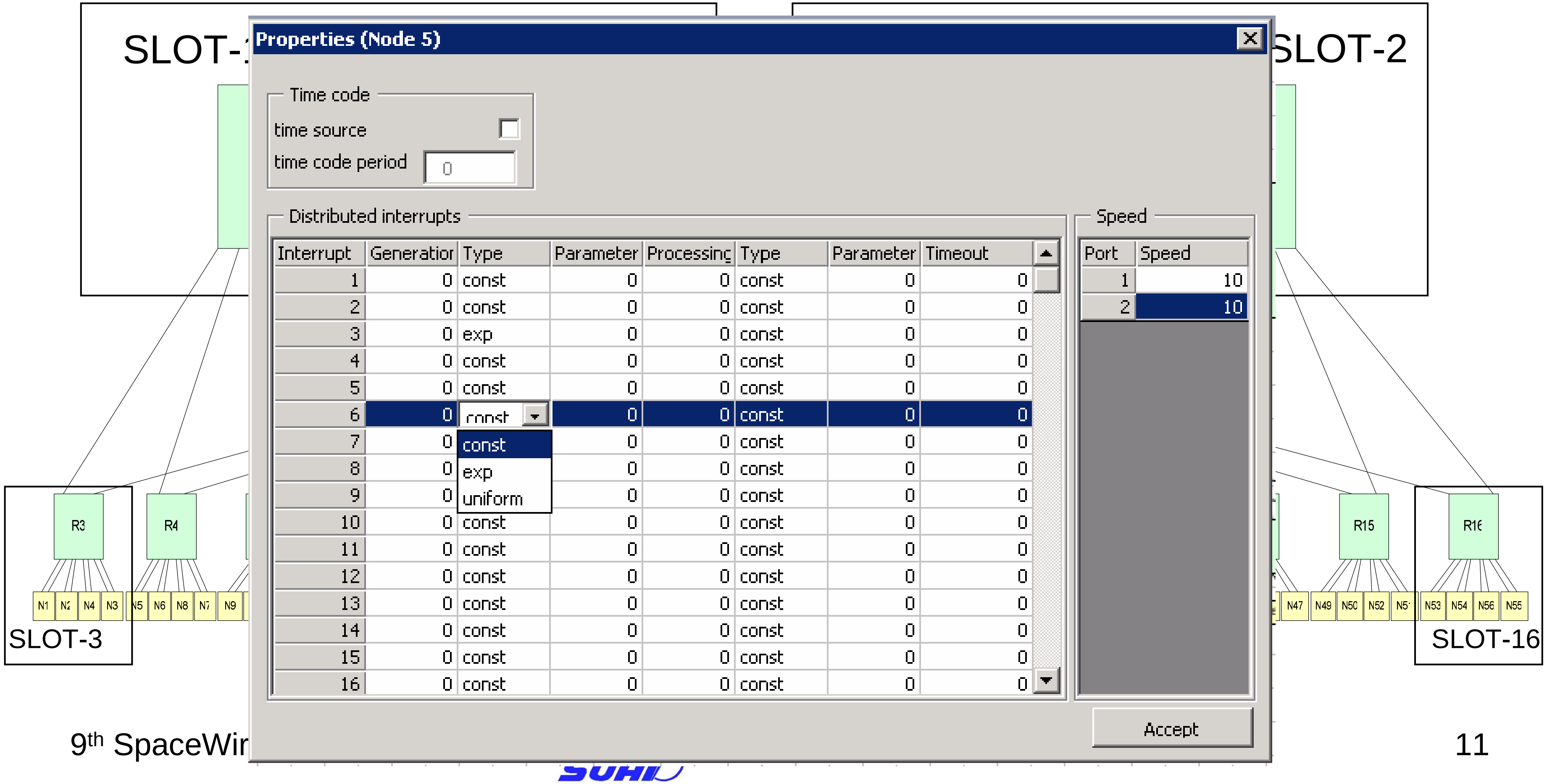


SpaceWire network example (MS Visio)

SpaceWire Network – double star (D=1)

with ATCA (Advanced Telecommunications Computing Architecture)

Parameters settings for Node 4



Time code

time source □

time code period 0

Distributed interrupts

Interrupt	Generation	Type	Parameter	Processing	Type	Parameter	Timeout
1	0	const	0	0	const	0	0
2	0	const	0	0	const	0	0
3	0	exp	0	0	const	0	0
4	0	const	0	0	const	0	0
5	0	const	0	0	const	0	0
6	0	const	0	0	const	0	0
7	0		0	0	const	0	0
8	0		0	0	const	0	0
9	0		0	0	const	0	0
10	0	const	0	0	const	0	0
11	0	const	0	0	const	0	0
12	0	const	0	0	const	0	0
13	0	const	0	0	const	0	0
14	0	const	0	0	const	0	0
15	0	const	0	0	const	0	0
16	0	const	0	0	const	0	0

Speed

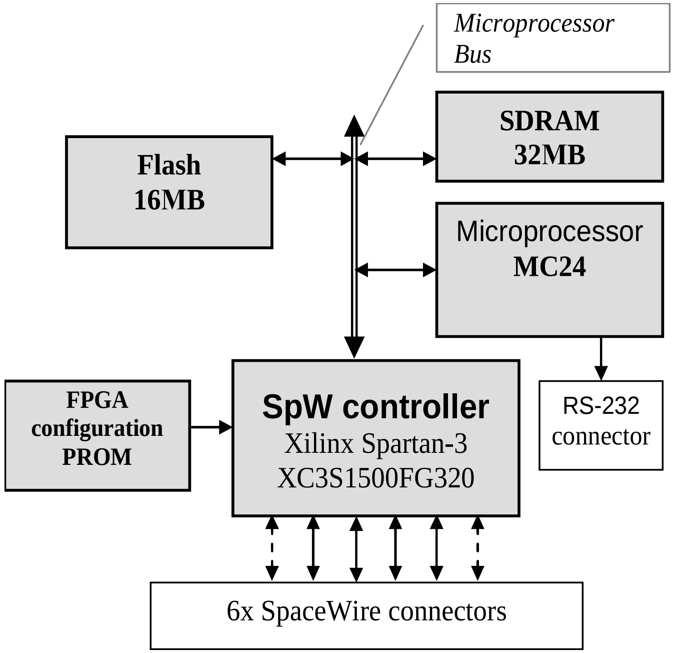
Port	Speed
1	10
2	10

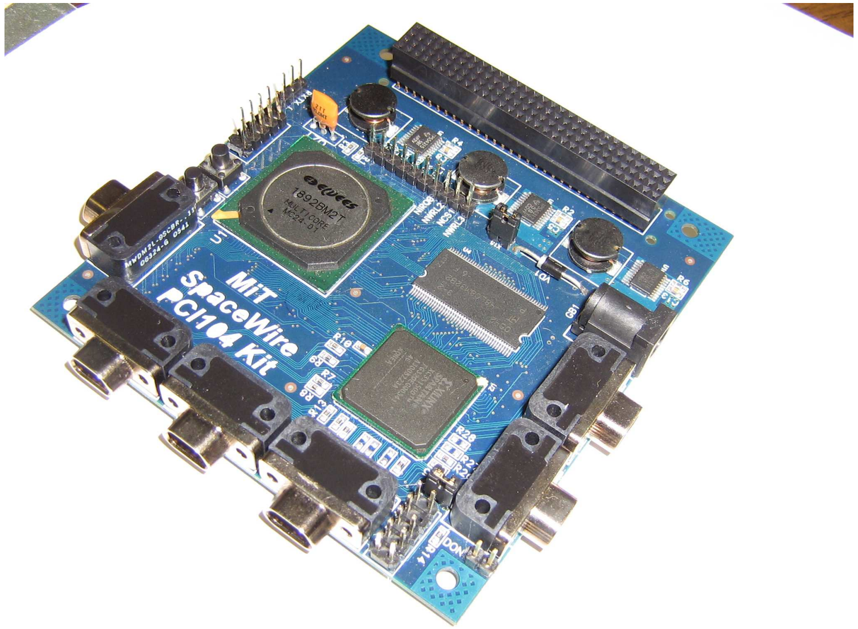
PCI104 SpaceWire Kit

The one-board high performance DSP module:
- a ready-made building block

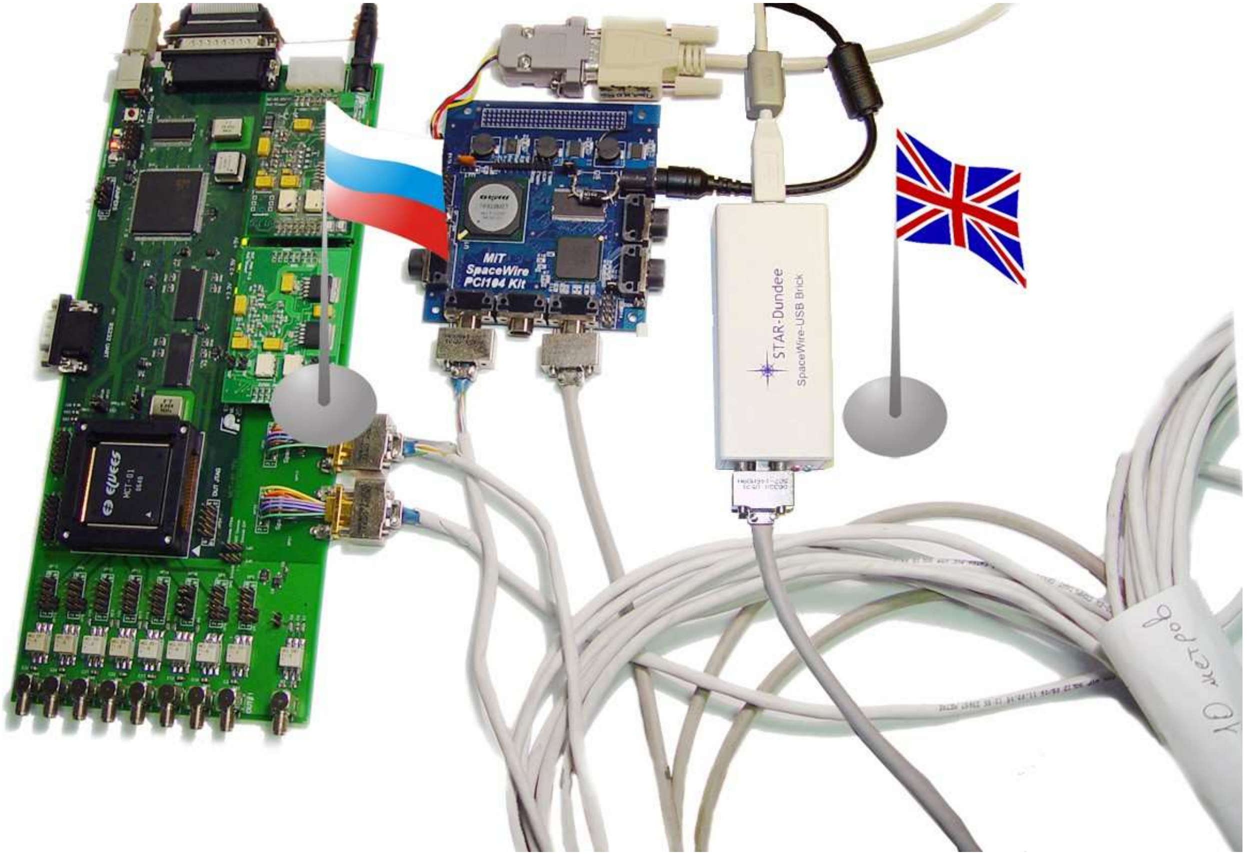
- for SpaceWire-compatible EGSE systems
- for on-board distributed & parallel space data systems prototypes with high-speed SpaceWire communication technologies.

PCI104 form-factor





Sample Trial Interconnection of SpaceWire implementations

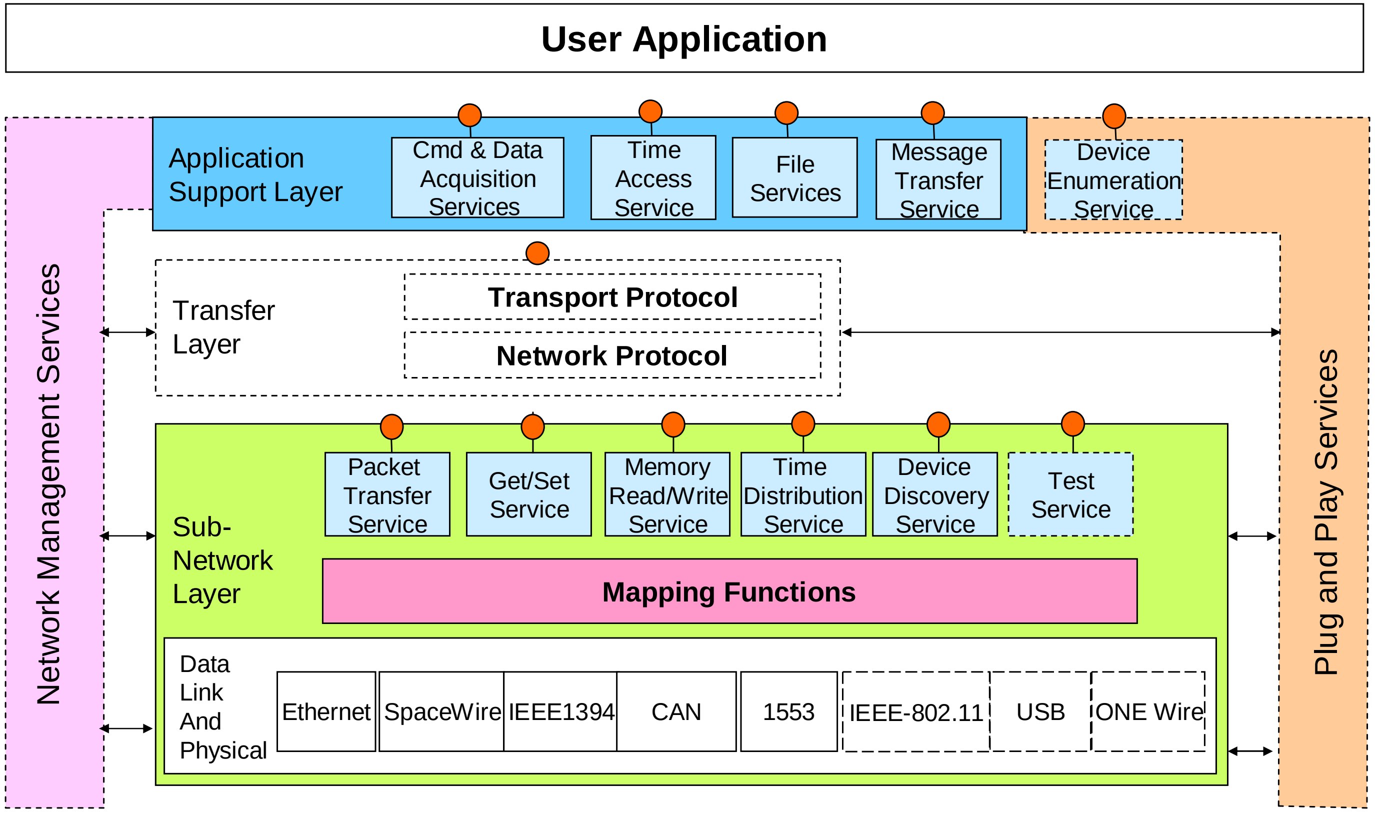


SOIS General Remarks

1. Services specification in SOIS
   - A wide range of qualitative requirements

     - ***Good*** !
   - ***But*** :
     Lack of quantitative parameters
     (a set of parameters, not particular values)
2. General SOIS Architecture
   - Network/Subnetwork in the SOIS multi layered structure

Network/Subnetwork in the SOIS multi layered structure



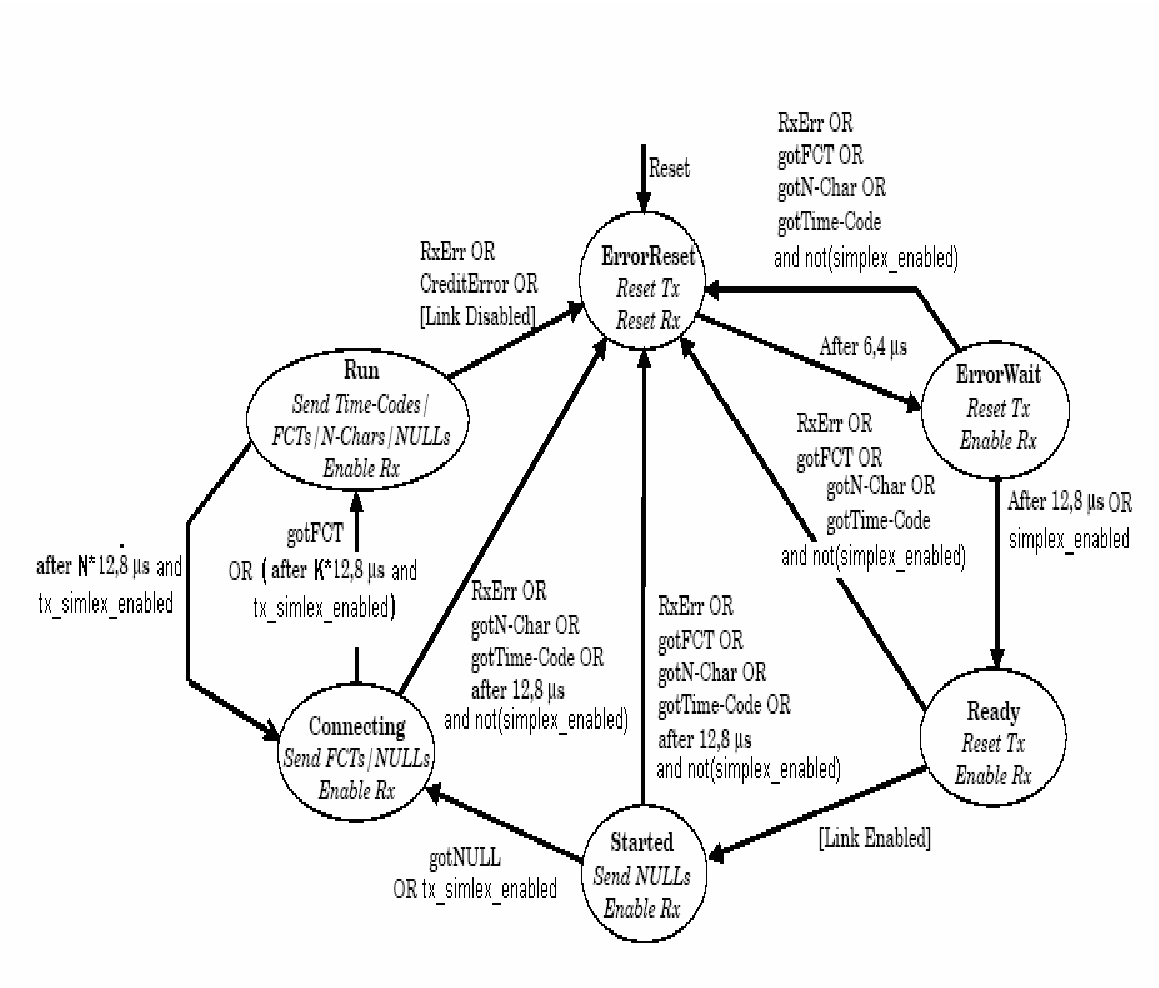
● Denotes service access point

Simplex SpaceWire links operation mode

- Simplex mode of SpaceWire links was claimed to be useful at some previous SpaceWire WG meeting
- Some SpaceWire potential users and applications would like to have simplex links
- One of the main reasons – to minimise cabling, number and weight of cable wires: 4 wires instead of 8. (Harness minimization)
- It is not hard to implement a Simplex mode of a SpaceWire link operation just switching of FCT-based flow control in a transmitter and a receiver
- The problem is in managing link disconnection:
  * the Receiver will be reset and wait for *NULLs* to restart the connection
  * the Transmitter will never know that link disconnection happened and will continue to send data, N-char and EOP, not *NULLs*
- Some form of “re-synchronization” between Transmitter and Receiver operation is required

Simplex SpaceWire links operation mode (2)

- The standard state machine should be slightly changed to deal with the problem
- There some variants to do it
- To do it in a consistent manner for various SpW controllers implementation – a subject for SpW WG standardization



Thank you !

Back up