Outline

1. Introduction
2. TraffiController\(^4\text{SpW}\) DVS
3. TraffiController\(^4\text{SpW}\) API
4. TraffiController\(^4\text{SpW}\) GUI
5. Conclusion and questions
3 Java software layers, with a gateway to C language

- Devices Virtualization Service
- Application Programming Interface
- Graphical User Interface

Multiple functions in one single software package

- SKYLAB test hardware management, data communication and control
- Integration and test, simulation and network dimensioning thanks to:
  - virtual routers,
  - RMAP support,
  - IP/Ethernet tunnels,
  - real, emulated or virtual nodes,
  - software implementation of SpW link analyzers,
  - communication statistics,
  - etc…
1. Introduction
2. TraffiiController\textsuperscript{4SpW} DVS
3. TraffiiController\textsuperscript{4SpW} API
4. TraffiiController\textsuperscript{4SpW} GUI
5. Conclusion and questions
**DVS main functions**

- Enumeration and update of hardware
- Management of SpW devices, nodes
- Driver(s) routines access service
- ID_DVS listing

**DVS object-oriented architecture**
import util.AbstractSpWDevice;
import util.AbstractSpWNode;
import dvs.Generic;

public class Test {
    public static Map<Integer, AbstractSpWDevice> listDevice = new TreeMap<Integer, AbstractSpWDevice>();
    public static void main(String[] args) {
        listDevice = Generic.scanList();
        boolean exist = false;
        System.out.println("|");
        System.out.println("|__[List of SMARTCABLE devices]|");
        for (AbstractSpWDevice device : listDevice.values()) {
            if (device.getType() == 1) {
                exist = true;
                System.out.printf("|   |__SMC %d : IDDVS [0x%04x] : Initialized [%b] : Configured [%b] |
                    device.getIdDriver(), device.getUID(), device.isInitialized(), device.isConfigured());
                if (device.getListnode() != null)
                    for (AbstractSpWNode node : device.getListnode())
                        System.out.printf("|   |    |__Node %d : IDDVS [0x%04x] : Initialized [%b] : Configured [%b] |
                            node.getNumNode(), node.getUidDvs(),node.isInitialized(), node.isConfigured());
            }
        }
    }
}
DVS console

[Test Device]
1. (c) Get Info
2. (x) Exit

Enter your choice:

[List of MMI cable devices]
- Device A: Identified [false]: Configured [false]
- Device B: Identified [true]: Configured [false]
- Device C: Identified [false]: Configured [false]
- Device D: Identified [false]: Configured [false]

[List of PCI devices]
- PCI 0: Identified [false]: Configured [false]
- PCI 1: Identified [false]: Configured [false]
- PCI 2: Identified [false]: Configured [false]
- PCI 3: Identified [false]: Configured [false]
- PCI 4: Identified [false]: Configured [false]

[List of PCI Express devices]
- PCI Express 0: Identified [false]: Configured [false]
- PCI Express 1: Identified [false]: Configured [false]
- PCI Express 2: Identified [false]: Configured [false]
- PCI Express 3: Identified [false]: Configured [false]

For which node do you want to set frequency? Node ID -> ID0001
12

---------------------------------------------- Result ----------------------------------------------
Enter SPI CODEC transmission frequency value between 250kHz and 200MHz for node 0x0012: 125MHz
Setup SPI node 0 frequency of PCI Board 3 to 125.00MHz.
No error
1. Introduction
2. TraffiController$^{4SpW}$ DVS
3. TraffiController$^{4SpW}$ API
4. TraffiController$^{4SpW}$ GUI
5. Conclusion and questions
Application Programming Interface (API) overview

- SpW networking emulation and command control based on the following objects:
  - Router, node, ports objects
  - RMAP engine, multiple instances
  - *Ethernet communication for API dialogues* engine, multiple instances
  - Link streams for Virtual protocol analyzer engine, multiple instances

API main functions

- SpW networking using virtual routers, and virtual, real or emulated nodes
- RMAP support (ECSS-E-ST-50-11C)
- IP/Ethernet tunneling
- Virtual protocol analyzer
API Router

- 2 to 31 ports Wormhole routing,
- configurable via RMAP port 0
- path and logical addressing using routing table
- 3 arbitration modes:
  - packet priority, last served, lowest port number
- implementation inspired from SpW-10x router

API RMAP module
API: IP/Ethernet tunneling

- API interconnexion through IP/Ethernet
- local PC / local area / intranet or internet networking modes
- using Java RMI. Tested with VPN and direct client/server connexion.

API: Protocol analyzer

- between real or emulated nodes (PCI/PCI Express, smartCable), router ports and virtual nodes
- activable (on/off) for each direction of any link
- 4 configuration types: debug, blocking buffered, unblocking buffered, continuous
- 2 outputs: display or archive
- simple setMode() and getData() routines for easy mode config and data retrieves
Outline

1. Introduction
2. traffiController4SpW DVS
3. traffiController4SpW API
4. traffiController GUI
5. Conclusion and questions
Graphical User Interface (overview)

Main working area panel with intuitive functions access (right click or keyboard shortcuts)

Rich Menu Toolbar

Local API Hardware resources

Traffic statistic

Eye diagrams

Work area dimensioning

Log

2 – traffiController4SpW
Graphical User Interface (basic functions)

- Project management
- Device access
- Node management
- Peripheral initialization
- Peripheral configuration (ex: PCI4SpW)
Graphical User Interface (SpW read/write functions)

Packet generator

SpW or SIP* Send/Receive

*SIP is SKYLAB SpaceWire Interpreted Protocol, a 32bit ‘large cargos’ efficient processing format.
Graphical User Interface (RMAP Write function)
Graphical User Interface (RMAP Read function)
2 – traffiController4SpW

Graphical User Interface (RMAP RMW function)
Graphical User Interface (Router functions)
Graphical User Interface (IP tunnelling function)
Graphical User Interface (virtual analyzer function)
Outline

1. Introduction
2. TraffikController\textsuperscript{4Sp\textsuperscript{W}} DVS
3. TraffikController\textsuperscript{4Sp\textsuperscript{W}} API
4. TraffikController\textsuperscript{4Sp\textsuperscript{W}} GUI
5. Conclusion and questions
traffiController\textsuperscript{4SpW}

- DVS
- API
- GUI

are the 3 levels of trafficController architecture, which implements extensive software features useful for network emulation, integration and test.

It is provided with user manual, javadoc, test use cases.
TrafficController is included all hardware equipment commercialized by SKYLAB.
It is also available in stand alone (without hardware, using emulated nodes).

There is a Skylab4SpW Googlegroup community for user feedback and support.

SKYLAB plans to open the platform to competitive hardwares if demand strong enough.
We would like to hear about your potential additional requirements to be implemented.

Questions?
Thank you for your attention!

SKYLAB Industries SAS

42, av. du general de Croutte
31100 Toulouse - France

Tel. +33 (0) 5.61.41.77.03
Fax +33 (0) 5.61.41.63.56

spacewire@skylab-corporate.com
www.skylab-corporate.com
Spare Slide 1

Same priority arbitration
Different priorities arbitration

Same priority arbitration
Different priorities arbitration