WR Testbencher

Description of the WR Testnecher's components, please read with the drawing of the WR Testbencher, at the end of the document, by hand:

Testbencher GUI and User App.

- The interface presents configuration of the system, tests and results.
- The user can load template of the tests based on the RFC 2544/2889 and WR test, stored in a data base.
- The user can modify and save new tests in the data base.
- Before be able to run a test, the App must check that the system configuration and the DUT fulfils the requirements of the test.
- Transmission of the test to the WR Testbencher.

Implementation

C/C++ and Qt

Test and Result Data Base

- Contains the tests template (RFC 2544/2889 and WR test) and stores the new test save by the user.
- Stores the results of the Tests.

Implementation

MySQL or other open data base.

Collector

- Receives the test configuration from the GUI&App
- Processes and shapes the test information

Length Frame	Payload
Frame Rate	N° Trials
Burst Size	Destination Address
Traffic Direction	Load per Port
Interframe gap	VLAN flags

- Transmits the processed test information to the Engine
- Gathers the results of a test and sends them to the data base
- Creation of pseudo-random/specific destination MAC address

- Creation of pseudo-random/specific payload
- Creates a destination port table for a test that send different traffic, rate etc... in every port.

Testbencher Engine

- Stores the result of a test in memory and sens to the Collector , once is finished the test.
- Organizes the parameters of the test (length frame, payload etc..) and uses them to load, trigger and stop the Frame Generator
- Receives from the Frame Detector information of the test or DUT and stores the relevant information in memory.
- Modifies on-the-fly parameters of the running test, if the information received from DUT fulfils pre-defined conditions.

Frame Generator

- Generates frames with parameters: legnth, rate etc.. received from the Engine
- Indicates to the Frame2Port Allocator the destination port of the frames

Implementation

Maciej has developed a frame generator for testing the FEC, we could reuse it.

Frame Detector

- Detects pre-defined frames sensible for the test, either from the DUT or from the traffic's test
- Propagate the detected frames to the Engine, or to the WRPTP/RSTP dameon.

Frame2Port Allocator

- Allocates a frame in the queue of a destination port in the corresponding level of priority according to the information provided from the Frame Generator.

Implementation

It is quite similar to the part of the swcore_spec, from (what is so call in the swcore_spec) Input block to the Output block.

CoS Queues and Queue Manager

- Hosts accordingly the frames in seven (or less) queues coming from Frame2Port Allocator
- The Queue Manager transmits to its port the frames from the queues accordingly an algorithm defined to respect a different levels of priority.

Implementation

The Queue Manager algorithm is being researched in GSI in the frame of a Master Thesis.
The CoS Queues is a similar to the Output Block in the swcore