HaRTES Meeting

January 28, 2010, Aveiro, Portugal

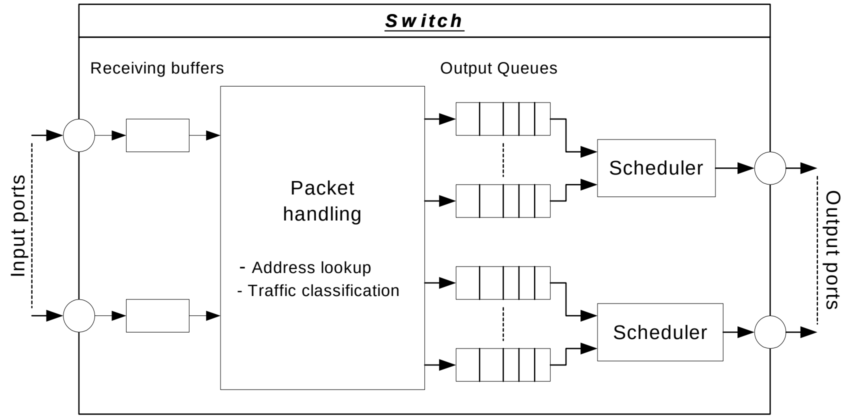
Enhanced Ethernet Switching Technology for Adaptive Hard Real-Time Applications

Rui Santos, rsantos@ua.pt, UA

SUMMARY

MOTIVATION

- **Switched Ethernet became common in real-time communications**
  - Some interesting properties
    - Large bandwidth
    - Cheap network controllers
    - Micro-segmentation
      - Collisions are eliminated
    - Multiple parallel forwarding paths
    - High availability



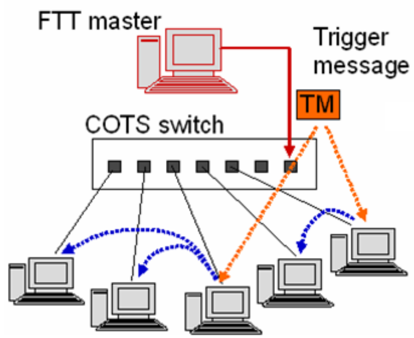
- **But there are still limitations**
  - FIFO queues
  - Limited number of priorities
  - Memory overflows

SOLUTIONS

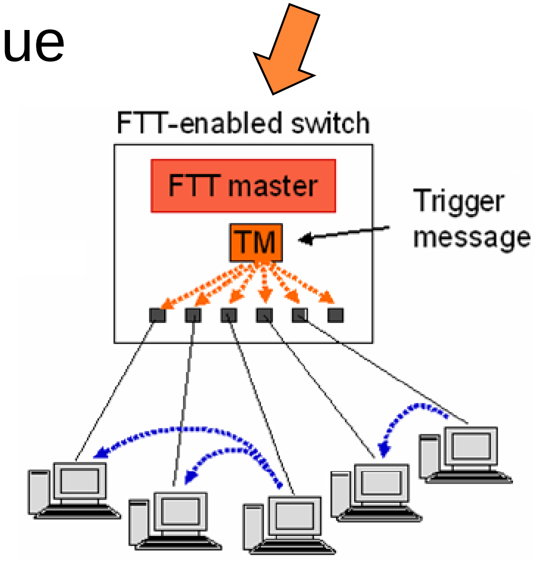
- **Commercial Of-The-Shelf Ethernet Switches**
  - Limiting the generated traffic by the application design
  - Traffic shaping
  - Master-Slave protocols (FTT-SE, …)
- **Customized Ethernert Switches**
  - TTEthernet
  - Profinet-IRT
    - Static pre-defined configuration
    - Online admission control is not generally available
    - Miss on-line adaptation
  - **FTT-Enabled Switch (our solution)**

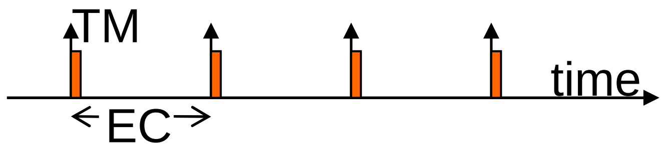
FTT-SE

FTT-Enabled Switch (Mechanism)



- Based on **Flexible Time-Triggered Paradigm**
- **Master-slave** transmission control technique
- Communication occurs in fixed slots (**Elementary Cycles – Ecs**)
  - **ECs** are organized in **synchronous** and **asynchronous** windows
  - Supports **synchronous**, **asynchronous** and **non real-time** traffic, with strict temporal isolation



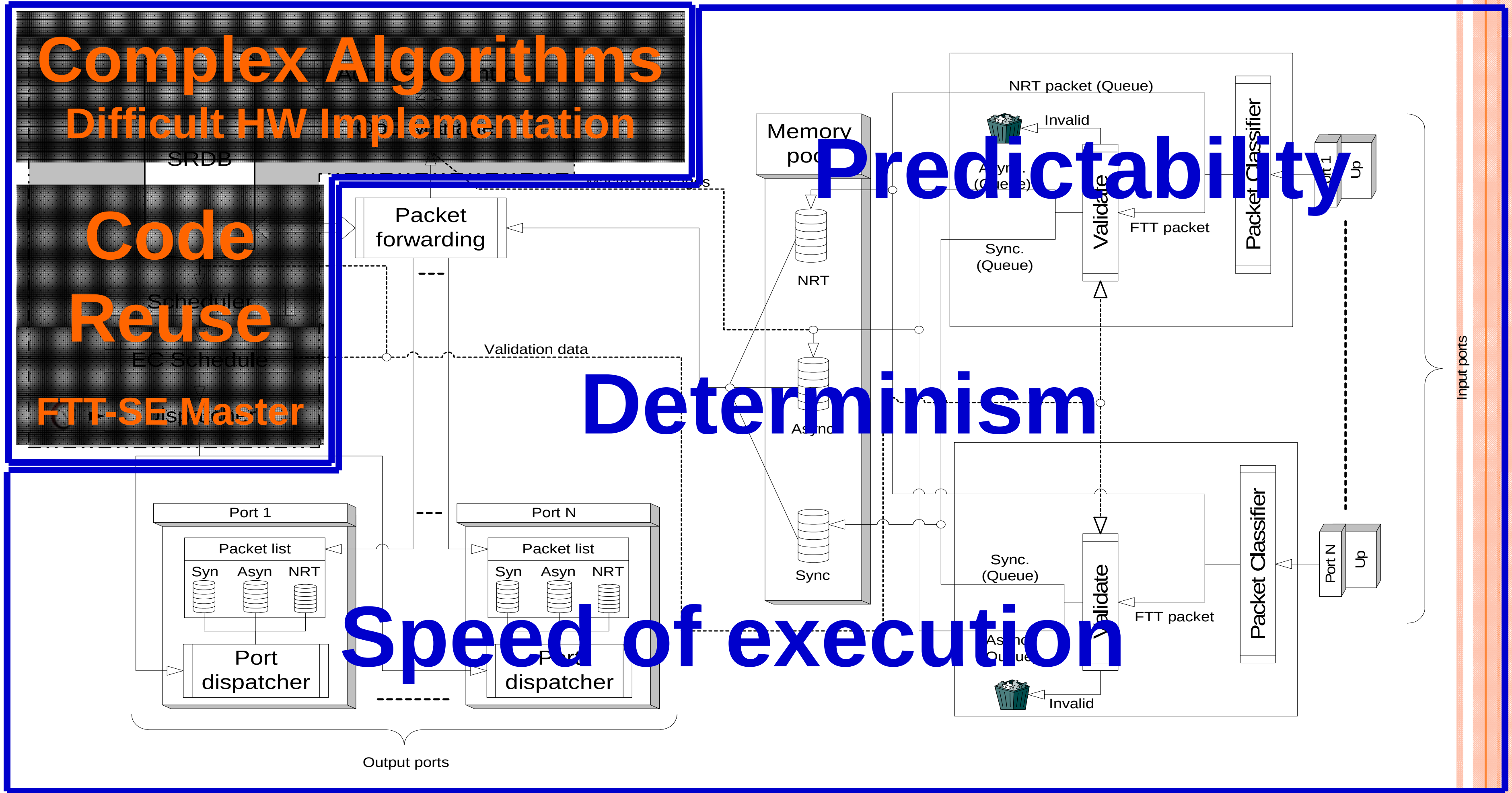


- The **ECs** start with a **Trigger Message (TM)** sent by the Master (switch), that contains the schedule for each EC

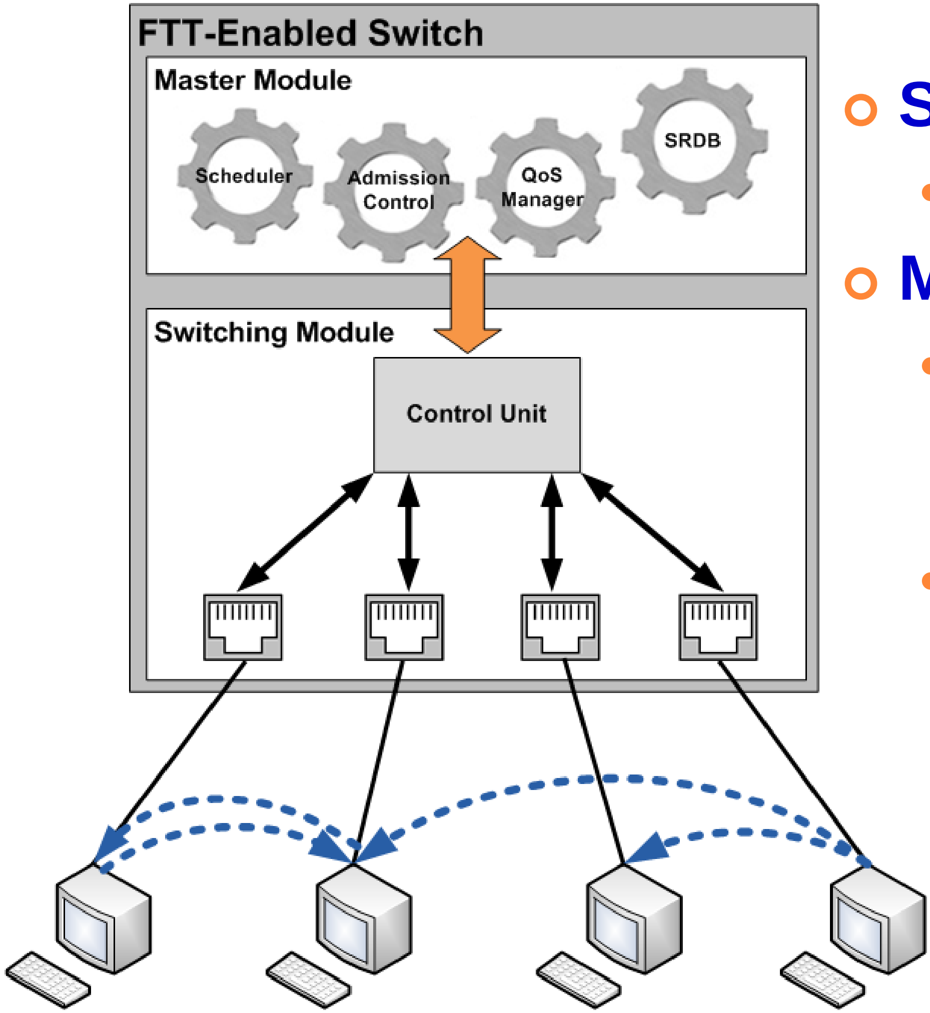
FTT-ENABLED SWITCH (PROPERTIES)

- **Traffic scheduling and management**
  - Old • **Global traffic coordination** in a common timeline
  - Old • Supports **online admission** control and **dynamic QoS management**
  - Old • Allows **arbitrary traffic scheduling** policies
- **Traffic classification, confinement and policing**
  - New • **Seamless integration** of standard **non-FTT-compliant** nodes without jeopardizing the real-time services
  - New • Asynchronous traffic is **autonomously triggered** by the nodes
  - New • **Unauthorized transmissions** can be **readily blocked** at the switch input ports, thus not interfering with the rest of the system

How can we implemet that?
FTT-Enabled Switch (Architecture)

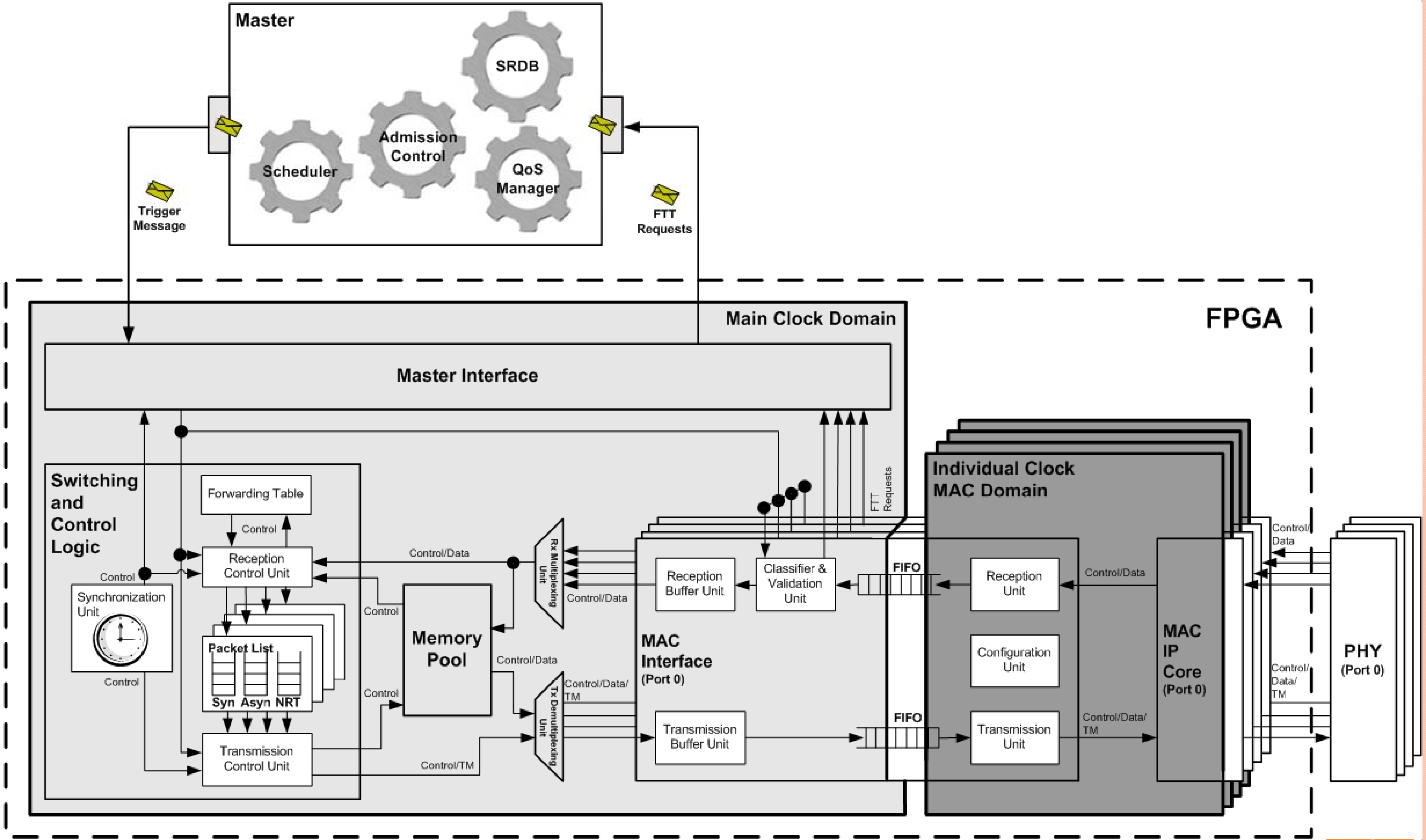


FTT-Enabled Switch (Implementation)



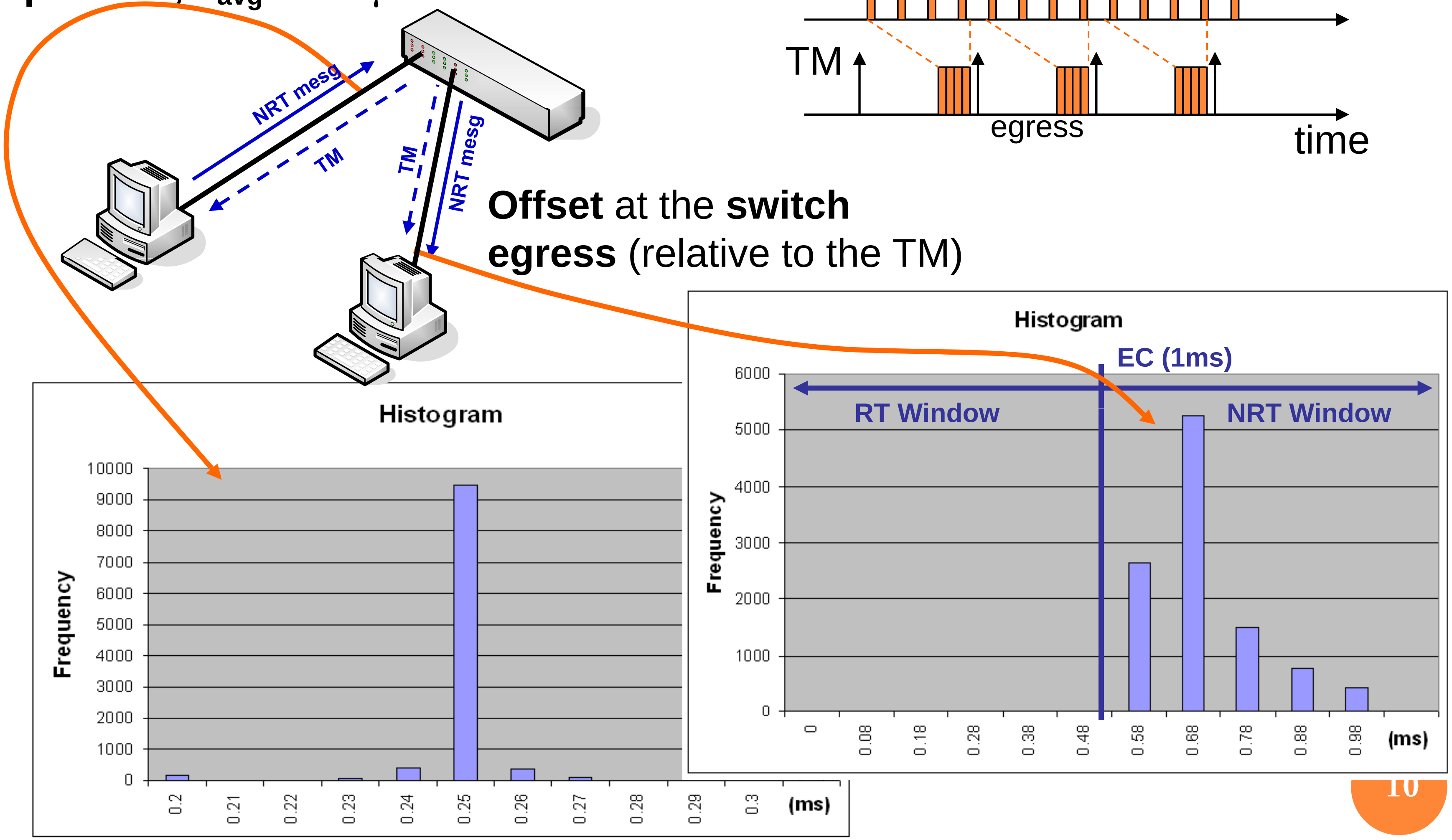
- **Switching Module**
  - Implemented in hardware
- **Master Module**
  - Implemented in FPGA embedded processor (Syntesizable or Hardwire)
  - Utilization of a CPU – communication with the FPGA is carried out by the conventional interface (Ethernet, USB, PCI, …)

FTT-ENABLED SWITCH (IMPLEMENTATION)

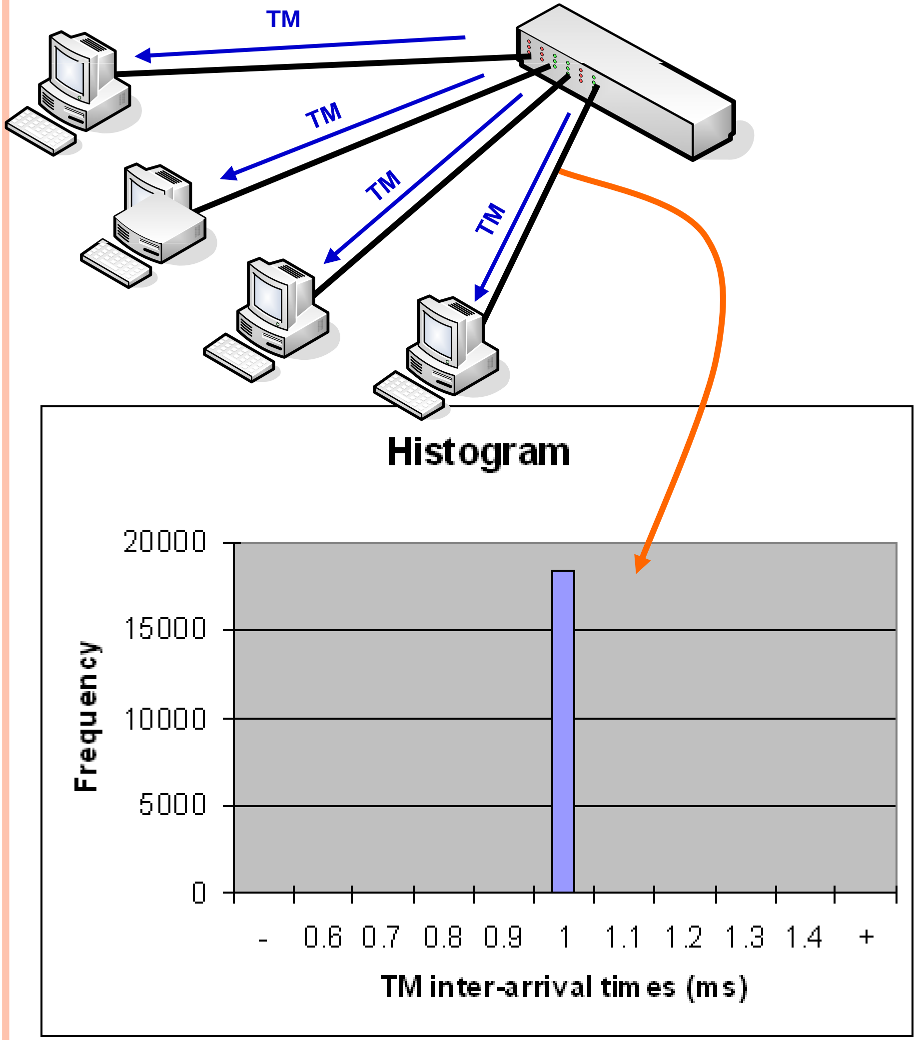


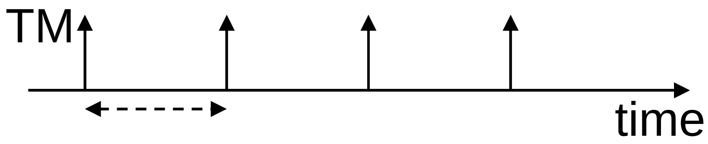
FTT-Enabled Switch (Experimental Results) Traffic Confinement

Submitted traffic
1kB packets, $T_{avg} = 250\mu s$



FTT-Enabled Switch (Experimental Results) Regularity of the TM



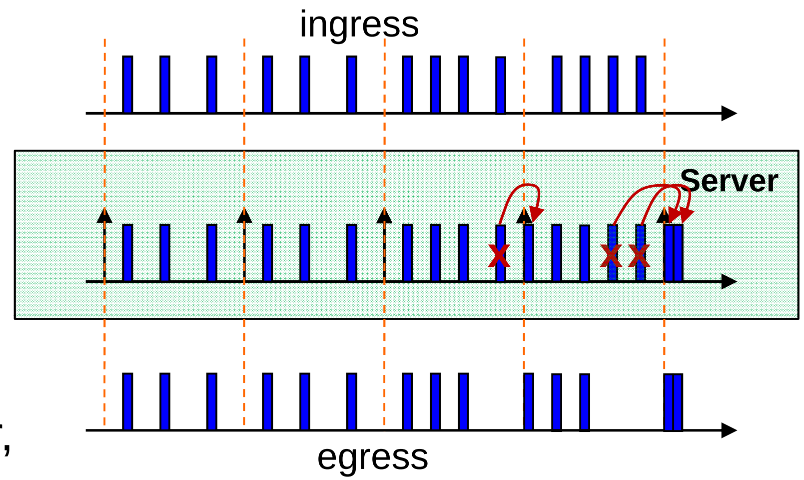


- Measures:
  - T_TM_{avg} = 1,000ms
  - T_TM_{max} = 1,0003ms
  - T_TM_{min} = 0,99998ms
  - STD_TM = 138ns

Jitter purely from the switch

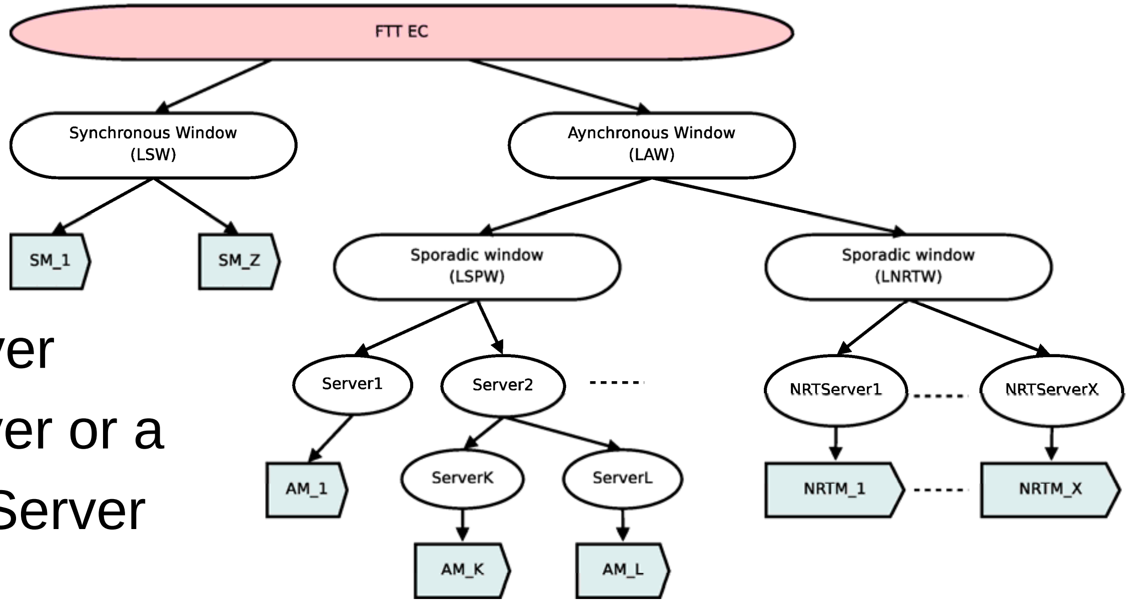
SERVER-BASED TRAFFIC SCHEDULING

- **Motivation**
  - Address the growing NES requirements to:
    - support streams with arbitrary arrival patterns
    - provide QoS guarantees .
- **Solution**
  - We propose to integrate CPU based server policy in the FTT-Enabled Switch
    - Polling Server, Deferrable Server, Sporadic Server



  - Providing hierarchical composition, reconfigurability and adaptability
    - Online creation, deletion and adaptation of servers

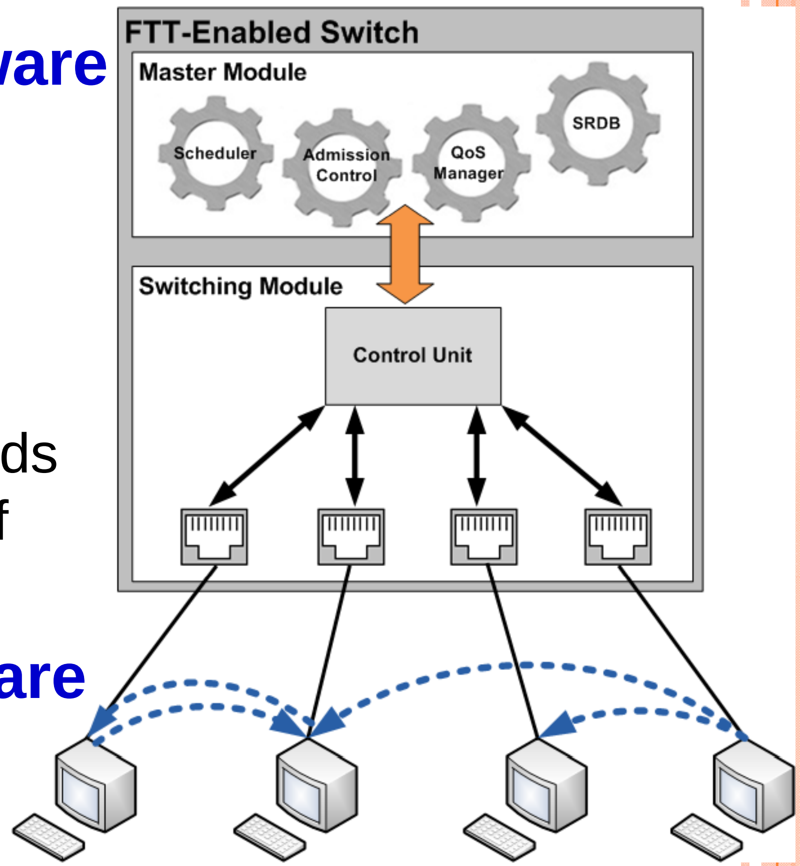
Server-based Traffic Scheduling (Integration)



- **First Level**
  - SW – Polling Server
  - AW – Polling Server or a Deferrable Server
- **Second Level**
  - Manages the sporadic and the NRT traffic inside the AW
- **Third Level**
  - Implements specific servers, *virtual channels*

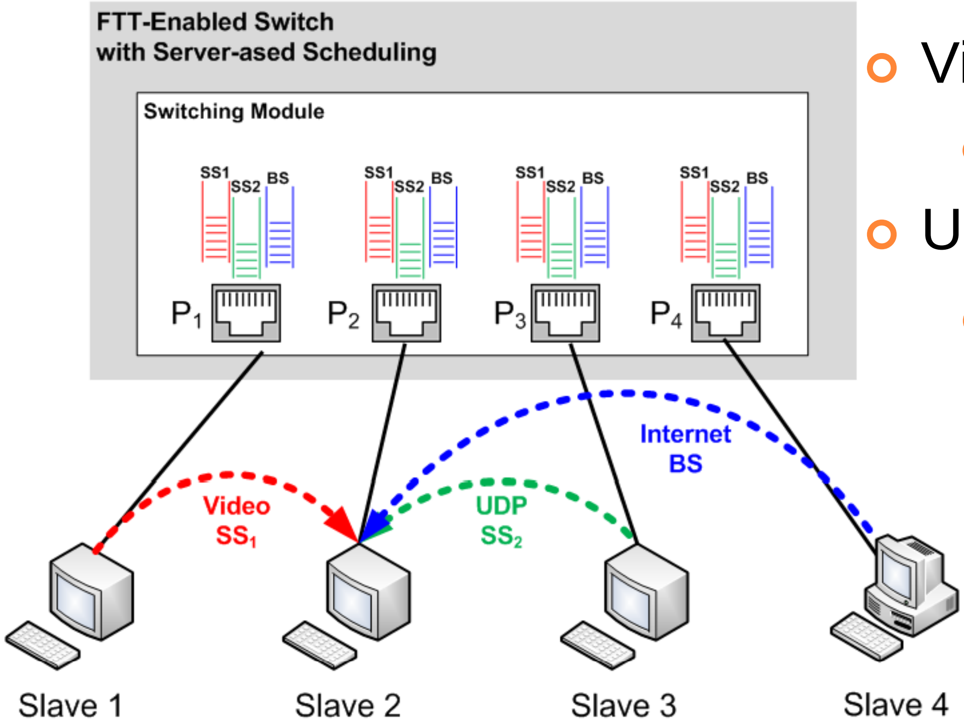
SERVER-BASED TRAFFIC SCHEDULING (IMPLEMENTATION)

- **Servers implemented in Hardware (Switching Module)**
  - High reactivity
  - Less flexibility (the number of the servers is fixed)
  - Complex server scheduling methods can require a significant amount of hardware resources.
- **Servers implemented in Software (Master Module)**
  - High flexibility
  - The server latency is relatively large

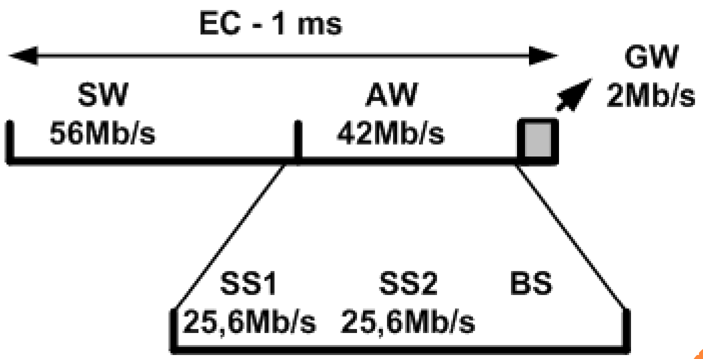


SERVER-BASED TRAFFIC SCHEDULING (EXPERIMENTAL RESULTS)

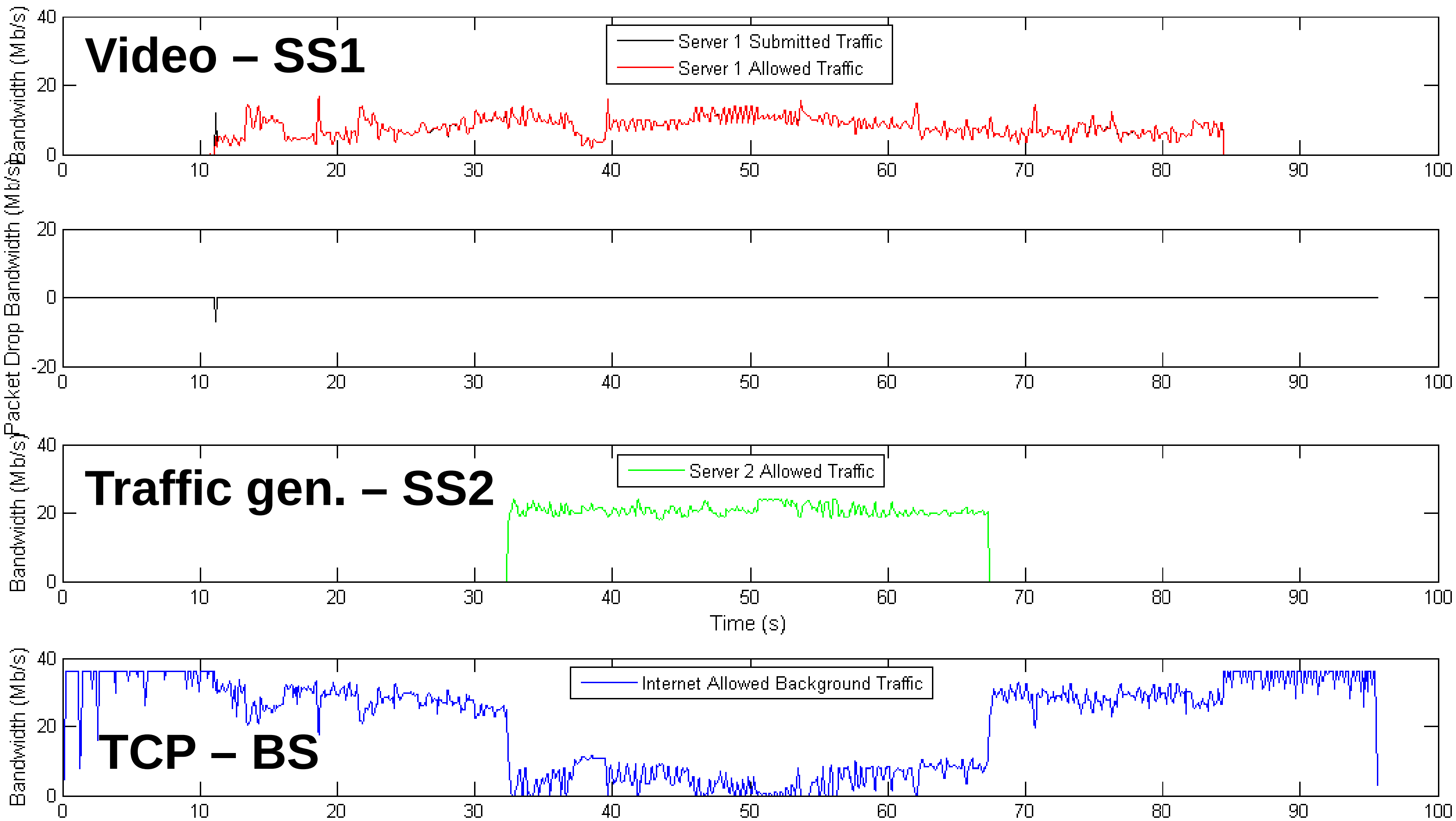
- Elemantary Cycle = 1ms; Asynchronous Window = 42%
- SS1, SS2 – **sporadic servers** with C=3200B and T=1ms
- BS – **backgound server** uses the remaining bandwidth



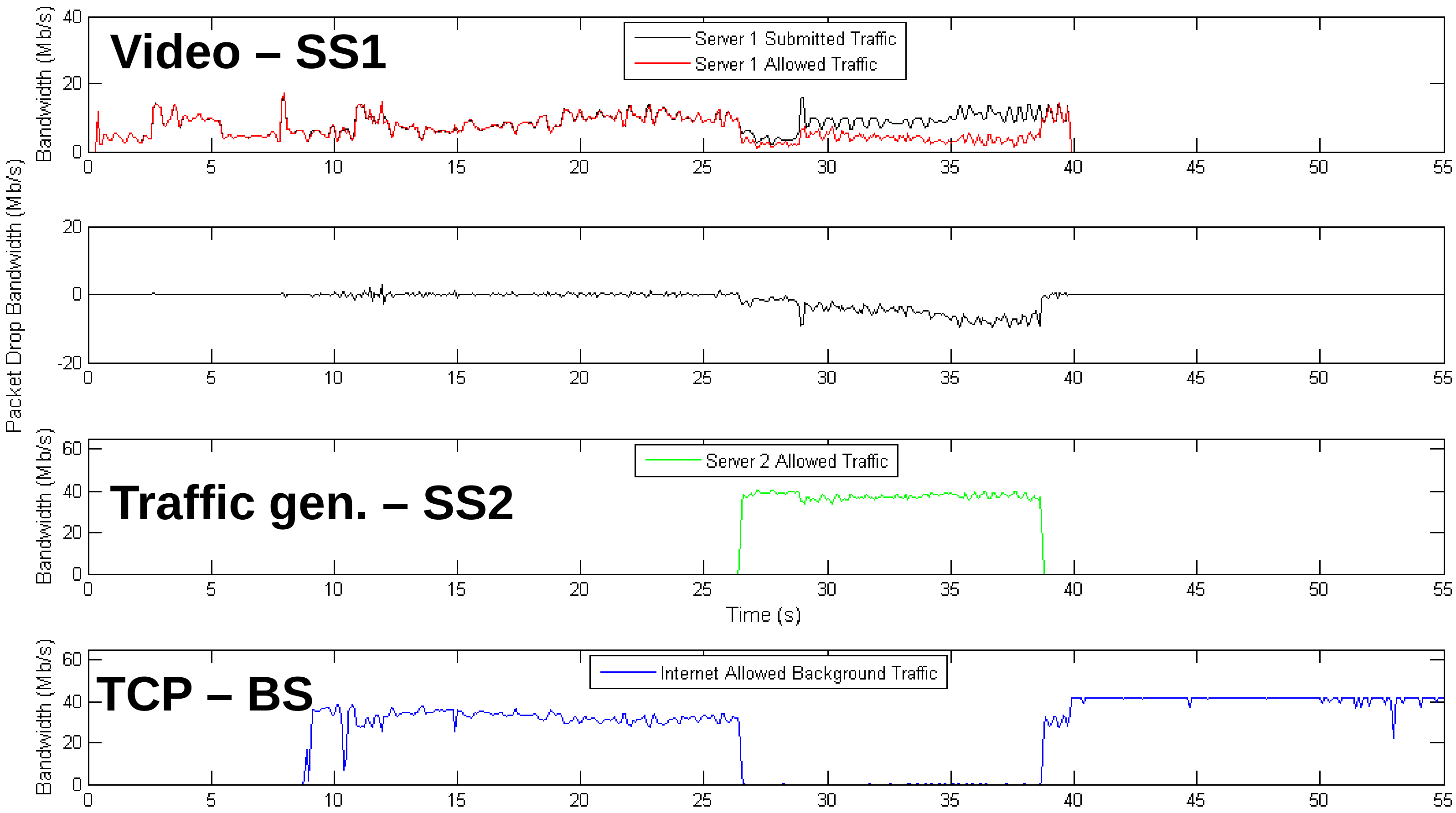
- Video SS1
  - Peak load = 21.9 Mbps
- UDP SS2
  - Average load = 99.9Mbps



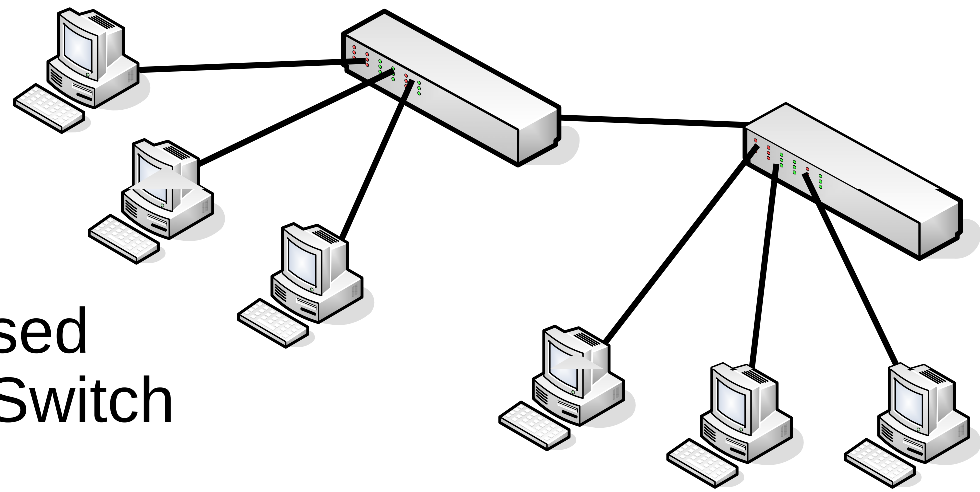
SERVER-BASED TRAFFIC SCHEDULING (EXPERIMENTAL RESULTS)



THE SAME EXPERIMENT WITH A NORMAL SWITCH!

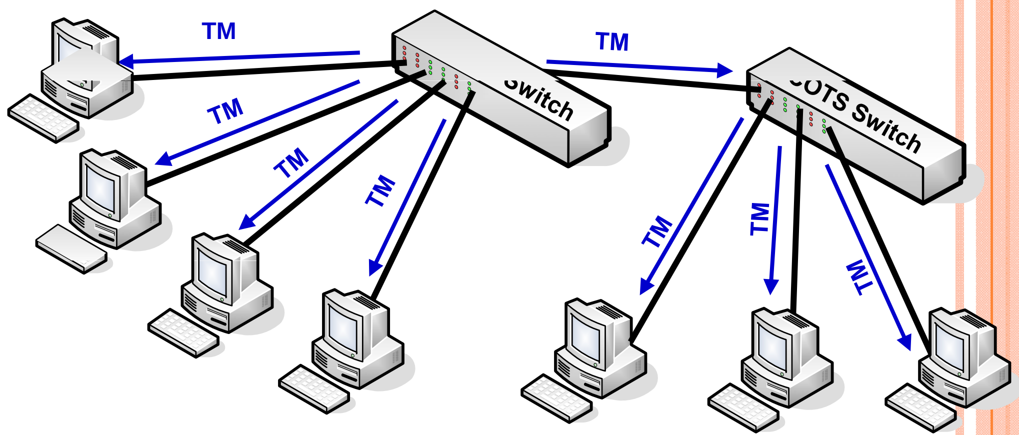


FTT-ENABLED SWITCH (MULTIPLE SWITCHES)



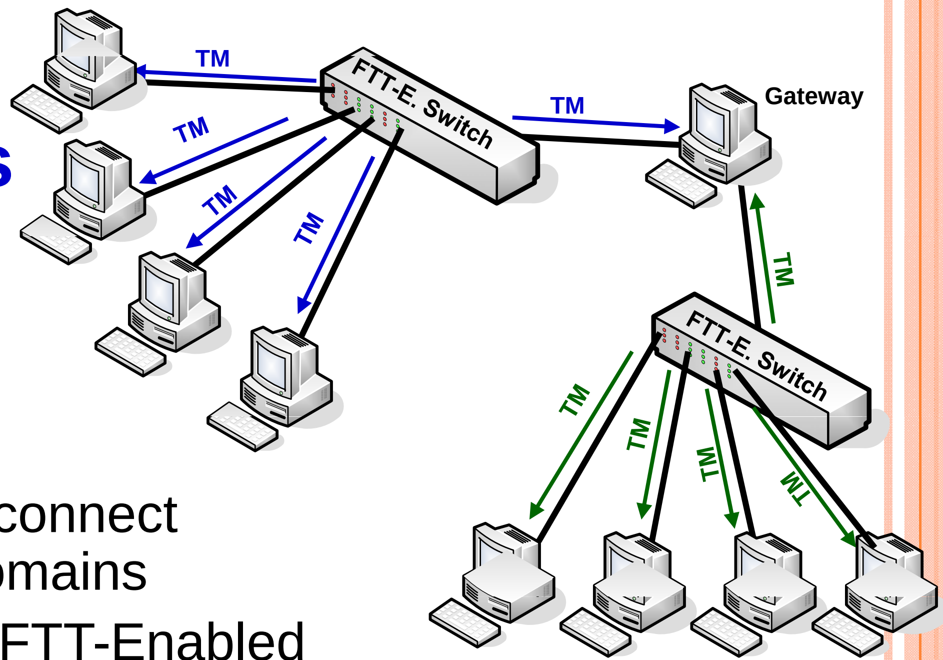
- **Problem**
  - How to create a network with multiple switches, where the communication is based on the FTT-Enabled Switch (HaRTES)?
- **Solutions**
  - **Network with one FTT-Enabled Switch and multiple COTS switches**
  - **Network with multiple FTT-Enabled Switches**

FTT-ENABLED SWITCH (MULTIPLE SWITCHES)



- **Network with one FTT-Enabled Switch and multiple COTS switches**
  - **Properties**
    - Trigger Messages are generated by FTT-Enabled Switch and disseminated by the others switches
  - **Advantages and Disadvantages**
    - ✔ Solution compatible with common networks
    - 🚫 COTS switches don't perform traffic policing
    - 🚫 The Trigger Message latency can generate problems of synchronization

FTT-Enabled Switch (Multiple Switches)



- **Network with multiple FTT-Enabled Switches**
  - **Properties**
    - Each FTT-Enabled Switch creates its own **synchronization domain**
    - It needs a **gateway** to interconnect different synchronization domains
    - Gateway can be avoided if FTT-Enabled Switches are slaves to each other
  - **Advantages and Disadvantages**
    - ✔ Whole network is covered by the traffic policing
    - 🚫 It needs a gateway

FTT-Enabled Switch (Current Status)

- **HaRTES/B**
  - Basic switching — Executed
  - Capability to separate different traffic classes — Executed
  - On-line scheduling — Executed
- **HaRTES/S**
  - Error detection
  - Traffic policing

  Partially executed
- **HaRTES/Q**
  - Dynamic QoS management capabilities — Partially executed

CONCLUSIONS

- The growing availability of FPGAs, associated tools and communication IP cores opens the way to build customizable devices with properties that are tuned to specific application domains
- We propose an enhanced Ethernet switch that:
  - Provides **seamless integration of** any (kind of) **nodes without causing any interference**
  - Provides **filtering of unauthorized transmissions**
  - Allows **arbitrary synchronous traffic scheduling policies**
  - Allows **arbitrary server scheduling** and **hierarchical composition**
  - Provides **dynamic creation** and **adaptation of servers**

On Going and Future Work

- Finish the proposed work in the project
- Study and integrate multiple switch architecture
  - Adapt the enhanced switch to allow integration in architectures with multiple synchronization domains
- Replicate the Master
- Study over the schedulability analysis of the server-based traffic scheduling

THANK YOU