An Integrated Hardware Solution for Mac Address Translation, MPLS-UNI, and Traffic Management in Access Networks

Harald Widiger¹, Stephan Kubisch¹, Daniel Duchow², Thomas Bahls², Dirk Timmermann¹

¹University of Rostock, Germany
²Siemens Networks Greifswald, Germany
Outline

- Access Network Architecture
- MATMUNI – Functional Elements
  - The MAC Address Translation (MAT)
  - The Multi Protocol Label Switching-User Network Interface (MPLS-UNI)
  - The Traffic Management (TM)
- Architecture of FEs
- Functional Integration (MATMUNI)
- Conclusion
Access Network Environment

- Need for Differential Services, increased QoS
- MATMUNI as a combined solution targeting MAT, MPLS-UNI, and TM
MAC Address Translation - MATMUNI

Customer Side | Provider Side
---|---
DST MAC | DST MAC
Customer SRC MAC | Provider SRC MAC
VLAN | VLAN
Ethertype | Ethertype
Data & Padding | Data & Padding

Upstream

Customer DST MAC | Provider DST MAC
SRC MAC | SRC MAC
VLAN | VLAN
Ethertype | Ethertype
Data & Padding | Data & Padding

Downstream
MAC Address Translation

- Scalability, flexibility
  - 1:1, n:1 and partial assignment of Provider-MAC
  - White list (just forward), black list (block)

- Protection against ARP spoofing
  - Attacks base on dynamic address table updates → but we have static address tables
  - DSL/Ethernet-flatrates are always on → „static“ address assignments

- Standards compliance
  - Frame size & header structure unchanged
  - Feasible with standard switching hardware
MPLS-Encapsulation

- MPLS Label Stack usually between layer 2 and layer 3 header
- We use encapsulation scheme by Martini
MPLS-User Network Interface (MPLS-UNI) - MATMUNI

- MPLS Label Stack container to carry information
- No complete LER implementation with an LDP running is necessary
  - Possibility to implement the whole system in Hardware
- Primary Functionality:
  - Upstream direction → insert an MPLS Label Stack
  - Downstream direction → remove MPLS Label Stacks
Traffic Manager - MATMUNI

- Link Layer Header
- MPLS Label
- Experimental Bits
- End of Label Stack
- TTL (derived from IP)
- Data & Padding

Upstream

- Link Layer Header
- MPLS Label
- Frame Color
- End of Label Stack
- TTL (derived from IP)
- Data & Padding

Key and No. of Bytes

Counter Logic & Color Information Tables

Color Information
Traffic Manager

![Diagram showing traffic manager counter values over time with labels for CIR-Counter, BIR-Counter, CIR, and BIR]
Traffic Manager

- Metering/congestion control already in access area
- Operates at wire speed
- Grants committed BW to customers in over-subscribed networks
- Fair
- Performs policing tasks by discarding red frames
Module Architecture – Single Module
Memory Arbitration
Architecture MATMUNI
Implementation Results  
(Xilinx Virtex 4 FX20-11)

<table>
<thead>
<tr>
<th>Hardware Module</th>
<th>Speed in MHz</th>
<th>Area min/max</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATMUNI</td>
<td>140</td>
<td>2300/4300</td>
</tr>
<tr>
<td>MAT (UP &amp; Downstream)</td>
<td>220</td>
<td>2x210</td>
</tr>
<tr>
<td>TM (UP &amp; Downstream)</td>
<td>190</td>
<td>2x240</td>
</tr>
<tr>
<td>MPLS-Labeler</td>
<td>180</td>
<td>129</td>
</tr>
<tr>
<td>MPLS-Delabeler</td>
<td>320</td>
<td>101</td>
</tr>
<tr>
<td>Memory Arbiter</td>
<td>160</td>
<td>282/1023</td>
</tr>
<tr>
<td>CPU Arbiter</td>
<td>160</td>
<td>640</td>
</tr>
<tr>
<td>Key Parser &amp; Framebuffer</td>
<td>150</td>
<td>336/723</td>
</tr>
<tr>
<td>Framebuffer</td>
<td>180</td>
<td>205</td>
</tr>
</tbody>
</table>
Simulation Results

- 8 independent GbE channels
- Artificial traffic
  - only minimal frames
- Realistic traffic
  - 35 % minimal
  - 11 % average
  - 10 % maximal
  - 44 % random
  - → no packet losses
- Average latency 130 cycles (≈1 us for GbE)
Conclusion

- Powerful and cost-effective solution to integrate different functionalities into the Access Network area
- @125 MHz, 4 Gbps can be handled
- Size of the system can be minimized considering the actual tasks
- Functional spectrum can be broadened, due to reconfigurable HW