Optical Ethernet – Flexible Optical Metro Networks

Wolfram Lautenschlaeger, Nihel Benzaoui, Fred Buchali, Lars Dembeck, Roman Dischler, Bernd Franz, Ulrich Gebhard, Jens Milbrandt, Yvan Pointurier, *Senior Member, IEEE*, Detlef Roesener, Laurent Schmalen, *Senior Member, IEEE*, and Andreas Leven, *Senior Member, IEEE*

***Abstract*—Enhanced flexibility in optical transport networks is a key requirement to support dynamic traffic load in packet based networks. Today, flexibility is achieved by packet switches linked by static point-to-point transport connections. Wide-stretched synchronization patterns, line coding schemes, and forward error correction (FEC) frames prohibit flexibility right at the transport layer. We introduce a new optical transport concept that combines packet aggregation with a *multipoint-to-point* line coding and FEC processing. This concept avoids the quadratic full mesh scalability problem of other aggregated switching technologies like, e.g. wavelength switching. It combines the flexibility of a distributed Ethernet switch and the performance of a leading edge optical transport system.**

***Index Terms*—Ethernet, forward error correction, metro networks, modulation, packet transport, signal processing, traffic grooming, latency**

I. Introduction

The integration of telecommunication networks and IT infrastructure is proceeding relentlessly. Large powerful data centers (DC) are one of the most important building blocks of the Internet. These DCs host business as well as private data and processing capabilities, i.e. data is stored, processed, transferred, and analyzed. Current DCs are characterized by huge computing power, storage capacity and performance based on centralized data. Due to their large geographical distance to customers, transport latency is rather high leading to slow response times. Moreover, flexibility to place computing and storage resources in one of these large DCs is currently often limited by the capacity and flexibility of the conventional interconnecting networks.

New applications of the digital society, for instance like Industry 4.0, Internet of Things (IoT), mobile objects (e.g. end-user device, cars, etc.) and, in particular, 5G will lead to increased data traffic with very different traffic patterns. A number of these applications require very short response times, e.g. 1ms for some 5G applications. Future network technologies, including 5G, will make heavy use of virtual and pooled network resources.

Virtualization and low latency requirements will lead to a new network architecture based on smaller, distributed DCs in order to provide essential data and functionality closer to the customer. The reduced storage and processing capacity of smaller distributed DCs will lead to an increased data exchange between DCs. As a consequence, traffic is expected to be far more dynamic in nature, and traffic volume is expected to grow much faster in the metro than in core networks. Clearly, today's big DCs continue to be indispensable and have to be integrated into this new network architecture.

Optical Ethernet (OE) introduces a novel Ethernet-like transport on optical layer bypassing power hungry electrical packet processing. This concept allows for statistical multiplexing and can adapt to changing traffic demands instantaneously without reservation, similar to the behavior of packet networks. Our architecture reuses ideas of burst and frame switching architectures that have been investigated in the past [16], [17], [18], [19], but utilizes opaque nodes with optical-electrical-optical (OEO) conversion and preserves a synchronous physical transport layer (i.e. uniform bit clock along the whole bus). Similar combinations have been investigated before, e.g. [24], but with different focus. In our case we essentially combine frame aggregation ideas with transport layer framing. Compared to the earlier publications, we optimize our architecture strictly for low latency and we resolve well known scalability limitations of frame aggregation by a novel *multipoint-to-point* aggregation scheme. The term "Optical Ethernet" should not be confused with the same term used sometimes by Metro Ethernet Forum (MEF) which describes a plain Ethernet architecture equipped with optical interfaces.

OE achieves significant savings with respect to switching capacity since the switching matrix per node needs only to be dimensioned for add/drop traffic. In addition, OE features an increased spectral efficiency due to signal regeneration in each

Manuscript received July 27, 2016; revised December 16, 2016; accepted January 24, 2017. Date of current version January 30, 2017.

F. Buchali, L. Dembeck, R. Dischler, B. Franz, U. Gebhard, W. Lautenschlaeger, A. Leven, J. Milbrandt, D. Roesener and L. Schmalen are with Nokia Bell Labs, Stuttgart, Germany (e-mail: {first.last}@nokia-bell-labs.com). N. Benzaoui and Y. Pointurier are with Nokia Bell Labs, Nozay, France (e-mail: nihel_djoher.benzaoui@nokia-bell-labs.com)

The research leading to these results has partly received funding from German and French Governments in frame of the CELTIC+/BMBF/DGE project SENDATE-TANDEM.

Copyright © 2017 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

node and provides a good traffic scaling.

The Optical Ethernet concept can be used in a wide range of applications, from access to core networks. In this paper, we focus on OE as an alternative for current metro area network (MAN) technologies, i.e. with a capacity of > 1 Tbit/s and a transmission distance of < 1500 km. From a network architecture point of view, future smaller DCs are envisaged to be connected to OE nodes in metro networks whereas big DCs are intended to reside next to head-end nodes which on their part are forming the backbone network.

The paper is organized as follows. We motivate our proposal in Sec. II. In Sec. III, the overall concept of OE is described. In Sec. IV we introduce the structure of the OE transport container and outline corresponding Operation and Maintenance (OAM) functions. Section V explains the traffic aggregation process and highlights buffer requirements. The underlying optical transmission system is introduced in Sec. VI, followed by a detailed discussion of signal processing, modulation formats (Sec. VII) and Forward Error Correction (Sec. VIII). We verify our design decisions by feasibility experiments with respect to power budget, optical signal to noise ratio (OSNR), and error statistics in Sec. IX. And we benchmark the network performance of our architecture compared to state-of-the-art switching technologies by packet level simulation in Sec. X. Finally, Sec. XI concludes the paper.

II. Motivation

In this paper, we introduce a new framing format in a field, where well established standards like OTH or Ethernet exist. We do so because we feel that some well-proved certainties in the field of networking will change in the near future with the upcoming trends towards Tbit/s links, stringent latency requirements in the sub-milliseconds range, and data and/or processor mobility, respectively.

The closest state-of-the-art technology fulfilling these requirements and our reference architecture is a chain of Ethernet switches constituting a bus or ring in a metro area, where both ends are rooted in an operator's core network. If we assume that the node interconnections will be in the Tbit/s range, then these links will need a spectrally efficient modulation format with high gain Forward Error Correction (FEC) even for moderate node distances. The most commonly employed high coding gain FEC schemes require large contiguous data blocks of 100,000 bit and more to be effective. While FEC schemes are feasible to implement at a Tbit/s scale, its processing pipeline and the usually iterative decoder introduces a considerable delay in the range of several microseconds (extending to the millisecond range with some high-performing FEC codes, like, e.g., staircase codes [5]) *per node*. In the chain of Ethernet switches, this FEC processing delay accumulates on a hop-by-hop basis. The alternative, an end-to-end FEC processing, is impossible for Ethernet, since the smallest Ethernet packets (512 bit) are too small for a self-contained high-gain FEC. Additionally, subsequent ingress packets cannot share a common FEC frame since they typically travel towards different destinations. The OE framework proposed in this paper enables end-to-end FEC processing and thus avoids the hop-by-hop accumulation of the processing delay, see Sec. X.C, Fig. 15.

On the network bus or ring, it is obvious that for a particular node the amount of transit traffic is large compared to the local contribution. In an Ethernet switch, this transit traffic in the Tbit/s range needs to pass all steps of packet processing (en-/decapsulation, routing, queueing, crossbar, etc.). This is possible, however, it is neither simple, small, or cheap, nor results in low delay or low power. For example, a CMOS ASIC and memory is typically clocked at less than 1GHz. Thus, the throughput of 1 Tbit/s requires parallel processing of more than 1000 bit, which eventually means fetching and individual routing of *two* Ethernet packets *per clock cycle*. Our OE framework largely avoids the transit traffic processing, essentially only a one-to-one forwarding of the received symbols is performed. The transit traffic is not queued (cut-through) and local traffic is added/dropped in chunks not smaller than 1600 bit (sub-containers, Sec. V).

Bypassing techniques are frequently used to relax the transit traffic load on switches, for example OTH or wavelength switching, which would be another state-of-the-art alternative to OE. These techniques are efficient in scenarios with stable traffic matrixes (plain collector/ distribution networks). But they lose their benefits in case of a highly volatile traffic matrix, e.g., due to mobility of data agents (processing in distributed DCs, edge clouds). The $O(N^2)$ complexity prevents full mesh bypassing. Therefore, variable or adaptive bypassing techniques have been proposed, e.g., Optical Burst Switching (OBS) [21], its generalizations to Light Trails [22], or Optical Packet Circuit Integrated Networks (OPCI) [23]. These concepts typically suffer from the complicated and time critical interplay of bypass instantiation and the associated routing change in the layer on top. Our OE framework bypasses the packet processing in transit without any assumption on the traffic matrix. It presents full mesh reachability to its clients, like a big distributed Ethernet switch. Even isolated packets on previously inactive source-destination pairs are delivered within 50µs, s. Sec. X.B, Fig. 14.

III. General concept of Optical Ethernet

Optical Ethernet is a connectionless transport technology which integrates a switching function into the physical layer. Positioned at or slightly below the layer 2 of the OSI reference model, it is meant to be implemented on a line card plugged into a packet switch or a router. The basic architecture is shown in Fig. 1.

The architecture is based on two counter-directional buses each carrying a synchronous chain of fixed-length transport containers, which are shared by all nodes on the bus. Client data, i.e. IP or Ethernet packets, are mapped back-to-back into these transport containers without inter-packet gaps. Empty containers arriving at a node are filled with client traffic and forwarded, or are forwarded unchanged to be used by downstream nodes. When loaded with content, a container is dedicated to a single destination node indicated by an address

field in the container header. Upon arrival at the destination, the content is extracted and the container slot is released for further use. An empty container can be claimed by any node and be used for any destination. The key differentiating property of OE is that intermediate nodes only examine the container header for distinguishing between transit traffic and traffic destined for the local node, while transit containers are forwarded in a cut-through-like manner without adding queuing delay.

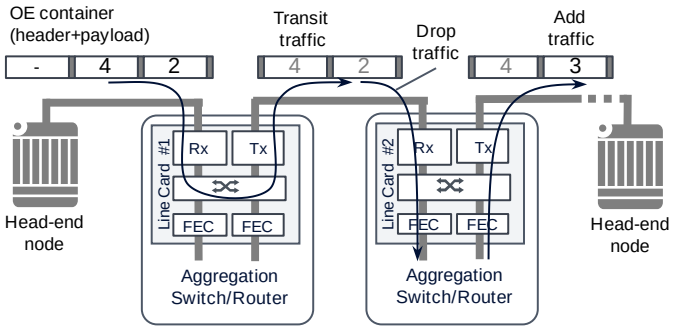

Fig. 1 Metro network architecture

In keeping with the example of a metro application scenario, the bus signal is modulated onto one or more wavelengths at net rates of 400Gbit/s and more, altogether targeting a bus capacity above 1Tbit/s. To be able to make a forwarding decision at line rate, an independent forward error correction (FEC) encoding and modulation scheme is used for the header which is robust and easy to detect. For the container body a spectrally efficient but processing-intensive modulation format is utilized (details in Secs. VI-VIII), which also uses a different FEC encoding scheme. This differentiation enables fast forwarding in transit nodes, where the container body is only regenerated but not decoded, and defers resource-consuming operations like FEC and higher-layer packet processing to the final destination node.

We assume roughly a 1:10 ratio of client interface rate to bus rate. So, given a bus rate of 1 Tbit/s, we are considering OE nodes that are able to add/drop packet traffic at a rate in the range of 100Gbit/s. As a consequence, a single node is not able to claim the whole bus. Free access to any of the empty containers enables the transmission of any combination of traffic relations in any variation over time as long as the given bus capacity is not exceeded. For cases of bus congestion, we envisage a fairness mechanism that equalizes the capacity share between the nodes, see Section IV.

In a technical implementation, the OE node can be conceived as a line card in a switch or router forwarding its outgoing and pre-selecting its incoming traffic. This pre-selection prevents that the transit traffic (which is the majority) needs to cross the backplane for higher-layer processing, thereby saving significant router capacity in contrast to a plain Ethernet solution.

The aggregation of client packets into larger transport containers is motivated by the transport-layer FEC, which should operate on larger blocks than the length of a single packet in order to allow for high net coding gains close to the theoretical capacity limits. In contrast to existing point-to-point transport technologies, the FEC information in a connectionless transport network like OE cannot be conveyed in a continuous stream across container boundaries, because containers subsequently arriving at a destination node can be from different sources. Streams would need to be separated by source node and precautions against starvation would be necessary. Thus, we chose containers with a self-contained FEC. The container size is a tradeoff between FEC coding gains, which demands for larger containers, and container aggregation delay, which increases with container size. Our current assumption is a container body of 102,656 bit or 12,832 bytes.

At low load, the accumulation of one full container could take exceptionally long time. Therefore, we limit the accumulation time by a reasonably short timeout, currently set to 50µs. After that time, the container is delivered to the bus, even if not full. Naturally, the empty space in the container wastes bus capacity and this waste increases with the square of the number of nodes – a full mesh scalability problem.

To decrease the waste of capacity, we introduce the new concept of *multipoint-to-point* containers. It enables intermediate nodes to add client packets to transiting containers on-the-fly if they have the same destination address and if the remaining container space permits. The idea is built on a specific *multipoint-to-point* FEC and exploits the significant asymmetry of complexity between FEC encoding and decoding. Using a linear code, encoding can be implemented with minimum logic circuitry at Tbit/s speeds. (For more details see Sec. VIII.)

IV. Container format and media access control

The format of the OE container is depicted in Fig. 2. It consists of a 120-bit header and 102,656 bit (12,832 byte) of gross payload subdivided into 80 sub-containers. The header uses a separate line encoding so that it can be easily inspected in every node. It carries mainly the container's destination node address and the number of occupied sub-containers. Each of the sub-containers carries 127 byte net payload, a 32-byte portion of the FEC parity bits, and a 1-byte sub-container source address, or, more precisely, the 7-bit OE address of the node which added the sub-container to the container. This scheme of destination address per container, but source address per sub-container reflects our *multipoint-to-point* architecture. The last sub-container is followed by another 32-byte parity block that terminates the FEC chain. The specific size of sub-containers is a trade-off between several performance factors like, e.g. memory organization, scheduling granularity, protocol overhead, and partitioning of the FEC processing.

The detailed format of the OE header is shown in Fig. 3. The meaning of the fields is as follows:

- "Busy" indicates whether the container is empty (0) or occupied (1). Partially filled containers are marked as

occupied, because their destination address is already set. Stations which have nothing to send, transmit or forward empty containers which can be occupied by downstream stations.

- DA, SA (Destination/Source Address) carry the OE address of destination and source node, respectively. A SC/MC bit decides whether DA is a singlecast or a multicast address. SA refers to the address that spawned the initial container.
- LEN (Length) carries the number of occupied payload sub-containers. If space permits, subsequent nodes can add further sub-containers, with same destination, and update the LEN field accordingly.
- CTRL indicates the presence of a control sub-container, if any, at the end of the container.
- FR (Fair Rate) and FR-Src carry the fair rate and its computing source, respectively.

Further fields are defined for operation and maintenance purposes (OAM).

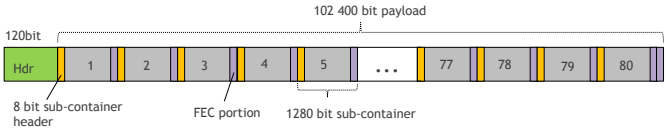

Fig. 2 Optical Ethernet container format

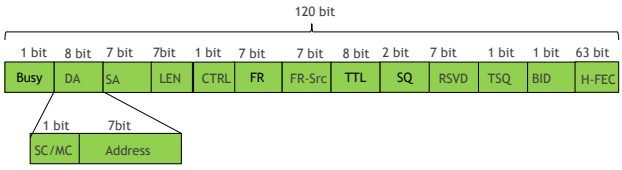

Fig. 3 Optical Ethernet header format

The header in the current design comprises only 57 information bits which are complemented by a strong FEC of 63 bits to a total of 120 bits. The shortness of the header, which is key for the OE architecture, inhibits the use of native Ethernet MAC addresses for transport containers. Therefore a mapping table is needed, which maps higher-layer addresses to OE addresses, and which is populated by a learning mechanism or a routing protocol. In one scenario, OE replaces Ethernet for IP transport in the metro area and an ARP table maps IP addresses directly to OE addresses. In another scenario, the OE client is Ethernet encapsulating IP, and the mapping table maps MAC addresses to OE addresses. Bootstrapping of address tables and other status negotiation between nodes is done by an in-band control channel, cf. the header bit CTRL.

OE implements a distributed media access control (MAC) which borrows elements from the Resilient Packet Ring (RPR) protocol [1] to signal backpressure to upstream nodes in case of congestion. As long as the capacity on the bus is sufficient to absorb all client traffic, nodes have unrestricted access to transport containers. When a node becomes congested (local add traffic exceeds the free transmit opportunities on the bus), it signals a fair rate to upstream nodes, to which excessive nodes are throttled. The fair rate is computed as the bus capacity divided by the number of nodes competing for bandwidth on the congested link (header fields FR, FR-Src). For a detailed description of the fairness algorithm we refer the interested reader to [1]. Please note that the fairness mechanism only prevents sustaining overload in the milliseconds range (round trip delay), while OE queueing and scheduling still has to cope with overload effects in the microsecond scale.

V. Packet Transport Model

In this section, we describe the forwarding of containers along the bus, insertion of client packets into containers and sub-containers, and their extraction at the OE destination node.

A. Traffic aggregation into containers

Containers are traveling along the bus in a continuous chain. They are not stored in intermediate nodes. Only the header is decoded, whereas the body continues as a stream of FEC encoded transport symbols (e.g. DP-32QAM symbols). If the header indicates that the whole container or at least some of its sub-containers are free, client traffic may be inserted on-the-fly into these vacancies. To enable this in due time, the client traffic is processed in advance and stored as encoded sub-containers. Here, processing includes per-destination classification and queueing, concatenation and segmentation, adding FEC overhead, and encoding into transport symbols. As a result, sub-containers are the smallest (atomic) transport entities. It is worth mentioning that client packets are mapped seamlessly into sub-containers across sub-container and container boundaries. Fragmented packets are reassembled at the OE destination node.

Classification and per destination queueing is comparable to virtual output queueing (VOQ) [15] in a router or switch, with the difference that the routing table indicates the destination node on the bus instead of the outgoing port of the switch. The queues are stored in a shared packet buffer of at least 256 KB and up to several GB (see Sec. V.C), shown on the left side of Fig. 4. A subsequent framer performs the remaining processing steps and stores the encoded sub-containers into a playout buffer. This second buffer stage is much smaller in size (on-chip memory), but it is organized in larger chunks than the packet buffer. It enables the rapid delivery of sub-containers at bus speed.

Insertion of containers into the bus is controlled by a hybrid round robin scheduler according to following rules: The scheduler iterates over the set of nonempty queues. On any transmit opportunity it selects the next queue that either (i) holds content of at least one full container size or (ii) holds some content of age beyond a 50µs timeout, regardless of size. If, however, a partially filled container arrives from upstream (as a result of rule (ii) in an upstream node), then (iii) any sub-container for the same destination can join as the remaining space permits. The round robin iteration according to rules (i) and (ii) progresses independent of intermediate firing of rule (iii).

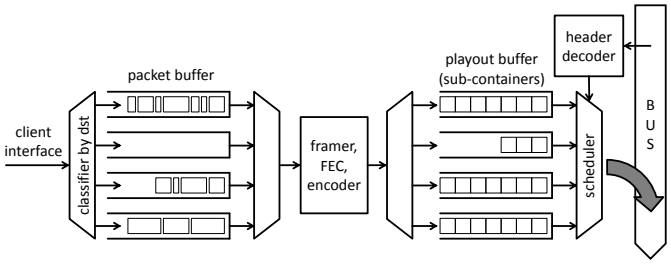

Fig. 4 Container aggregation at source node

Rule (i) is the dominating operating principle for our system. Rule (ii) is required to prevent starvation of isolated packets in low load situations. In our current considerations, we limit this waiting time to a timeout value of 50µs. As previously mentioned, the resulting partially filled container wastes bus capacity that cannot be used by any other traffic. At first sight, this looks like a problem of little importance since we are talking about a low load scenario. This is true for small node numbers but the wasted capacity scales with the square of the number of nodes and quickly reaches values in the range of the whole bus capacity. To avoid this, we introduced rule (iii). It allows not only to reuse the empty space in partially filled containers, but much more importantly, it drastically reduces the frequency of timeout events, since sole packets can easily hitchhike on partially filled containers from other upstream nodes with the same destination, long before their timeout expires.

The resulting *multipoint-to-point* containers include sub-containers from different sources but all with the same destination. The concept breaks the O(N^2) scalability problem of container aggregation and is built on specific FEC features as elaborated in Sec. VIII.B.

B. Unloading the containers at the destination

An OE container is dedicated to one particular destination node on the bus. As soon as the header decoder detects the appropriate address, the container body is dropped into a receive buffer, whereas the container slot is marked as empty for further use on the bus (cf. Fig. 5). The receive buffer holds a single first-in-first-out (FIFO) queue of containers or sub-containers. The FEC decoder operates on whole containers, so it can be placed before or after the receive buffer. Next, the content of sub-containers is concatenated *per originating source* and the resulting byte streams are split into the original client packets. This re-assembly stage needs to store packet fragments, but altogether not more than one maximum packet size per source. Finally, packets from different sources are multiplexed into the outgoing client interface.

C. Buffer sizing

In general, the packet loss in packet-oriented networks is due to congestion and related to buffer overflow. It is inevitable if too many sources are heading for the same destination (port, node, etc.) at the same time. Temporal overload can be absorbed by buffers upstream of the congested resource. Nevertheless, these buffers do overflow in case of sustaining congestion. Sizing of the buffers is a topic of controversial disputes ranging from few dozens of packets for a Poisson packet arrival process [13], up to gigabytes according to the bandwidth delay product rule for TCP traffic, and all compromises in between [14]. In a switch node, the affected buffer logically belongs to an output port. For reasons of switch matrix performance, however, it is distributed and physically co-located with input ports. The congestion signals are propagated back from output port across the switch matrix to the virtual output queues at the input ports. The back pressure concept can be extended to a hierarchy of switch nodes or even down to the end systems, e.g. by Quantified Congestion Notification (QCN) [20], but it is inherently limited to short distances by the signal propagation delay.

In our case, we have two congestion points – one at the entrance to the bus (sending node) and one at the outgoing client port at the receiving node. Bus congestion first affects the playout buffer but it can be propagated back across the framer onto the shared packet buffer. For this buffer, all dimensioning rules hold as for ordinary packet switches or routers [13], [14].

The playout buffer in the transmitter has to be able to store at least one container per destination. Otherwise, the on-the-fly insertion into the bus could not work. The playout buffer cannot lose packets. If it is full it sends back pressure signals into the packet buffer to throttle refilling. Nevertheless, if it is too small, it can lose transmit opportunities. If the bus is loaded at the limit, transmit opportunities are rare events that arrive independent of each other as a Poisson arrival process. If by chance several transmit opportunities arrive close to each other at bus speed, they quickly drain the playout buffer, but the framer cannot refill on time even though packets are still waiting in the packet buffer. Analytical calculations have shown that in the worst case, when the average rate of transmit opportunities equals the client rate and the playout buffer is just one container, we are losing 20% throughput. This worst case throughput degradation can be reduced to less than 5% with a buffer space of 3 containers per destination and an accelerated refill rate of 20% above the client rate.

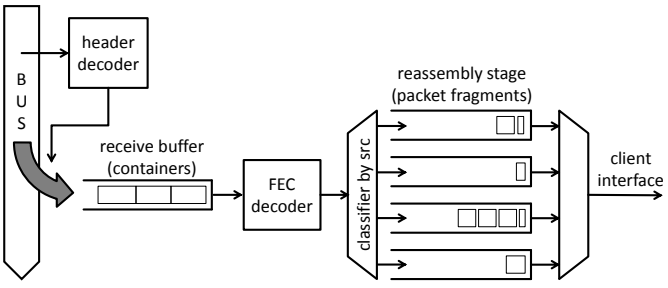

Fig. 5 Container reception at destination node

Congestion at the outgoing client port (receiving node) could be propagated back across reassembly stage and receive buffer down to the queues in the sending nodes on the bus, like it would be done in a compact node. In our case, however, the propagation delay would be too large due to the distribution of queues across metro distances. The bus wide fairness mechanism of Sec. IV can only prevent sustaining

overload. The receive buffer has to cope with the short term congestion.

If we look at a single bus direction and assume a 10-fold higher speed on the bus than on the client interface, we expect to *send* little less than one container per 10 container slots on the bus. At the *receiving* node, we are able to process one container during 10 container slots on the bus – no need to buffer at all. With two nodes sending one container in 20 slots each, independent of each other, the two containers could jointly arrive next to each other in a 20 slots period – for sequential processing, we have to buffer at least one of them. Hence, the required number of buffer slots initially grows linearly with the number of upstream nodes but finally converges at around 80-100 slots for an infinite number of nodes. The latter case is well known from literature as M/D/1 queue.

VI. Optical transmission system

The optical transmission system is determined by the targeted total capacity of the network beyond 1 Tbit/s and a maximum reach of up to 1500 km. The fundamental architecture of the system is shown in Fig. 1, where the opaque head-end to head-end system is divided into distinct segments from node to node. A state-of-the-art optical transmission approach is considered for this point-to-point connections with optical amplification by EDFAs for compensation of fiber attenuation and losses. To meet the considered requirements of the network's physical layer, we use coherent reception, higher order QAM modulation, and dual polarization transmission. Capacities beyond 400 Gbit/s per wavelength are feasible today [10]. To meet the capacity needs of more than 1 Tbit/s, we are targeting superchannel transmission with up to 4 wavelengths on the same fiber and a maximum of 400 Gbit/s per wavelength.

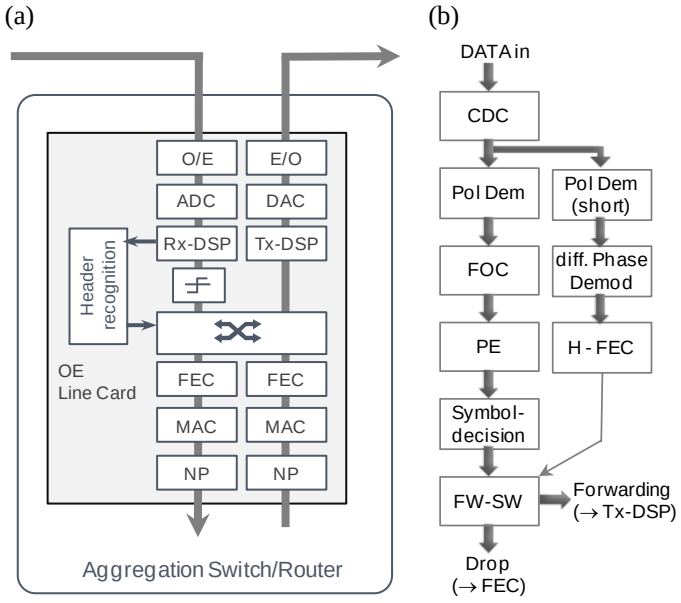

Fig. 6 (a) Connection of receiver and transmitter processing path after hard decision of symbols, (b) parallel processing of payload and header

The data transmission is organized in synchronous time slots, further denoted as containers. A short header at the beginning of each container carries the information about the respective container for the appropriate action within each node. During data transmission over several segments, every intermediate node has to extract and decode at least the information of the header; hence fast and error free header regeneration is mandatory. Coherent reception for the payload part inevitably requires an extensive digital signal processor (DSP). Contrary to the header, error free regeneration for the payload is not required because the data is protected by a strong FEC, which is performed in the sending and receiving OE nodes only.

VII. Signal Processing

The signal processing within an OE transponder must be adapted to the mentioned requirements. In the proposed system, a separation of header and payload modulation and processing is envisaged. For payload processing we require a scheme delivering an optimum performance with high spectral efficiency in the end-to-end transmission; beside the suppression of noise and signal distortions, we target a low latency as well, because each container may pass multiple intermediate nodes. As described earlier, we propose an end-to-end FEC for the payload, while in the intermediate nodes we do not correct errors in the payload.

Typically, coherent transponders incorporate two independent signal paths to process the transmitted and received data signals. If forwarding of traffic is required for the payload, a coupling of both paths is needed (Fig. 6 (a)). At that point we have to consider that the payload symbols are processed in the receiver part, but still suffer from noise and distortions. In a first feasibility experiment, we investigated the forwarding of regenerated symbols and found that a hard symbol decision leads to the best performance [11]. Hard decision performs a noise compression for the majority of symbols, whereas for very few symbols the decision may be wrong. Thus symbol decision errors are introduced and might accumulate during the transmission of a container, however the reduction of optical noise leads to an improvement of the system performance, if the accumulated error rate over several segments is considered. After hard decision, payload signals may pass the transmit path which conditions the signals for transmission over the subsequent segment.

Next we consider the header, which is transmitted at the beginning of each container. The header needs to be regenerated including an FEC decoder to derive the correct decision whether the payload has to be forwarded or dropped. (cf. Fig. 6 (a)). For optimum timing within the transponder, it is mandatory to have the header information already available when the payload leaves the receive DSP block. Otherwise the payload has to be stored on-chip. For the modulation of the header information we use a differential quadrature phase shift keying (DQPSK) in a dual polarization scheme to be fully compatible with the payload modulation format. In general, the header may be processed using the payload receiver DSP with some additional header decoding. However this approach

does not comply with the timing requirements. In more detail the receiver DSP consists of the chromatic dispersion compensation (CDC), the polarization demultiplex filter (Pol Dem) including channel compensation, the frequency offset compensation (FOC) and the phase estimation (PE).

We propose to apply a partial independent header processing (Fig. 6 (b)). The copy of the header is demultiplexed after compensating the chromatic dispersion. For polarization demultiplexing and channel compensation (Pol Dem) we propose an independent block of FIR filters of short length in a fully parallel architecture. The adaptation of the filter taps is performed periodically using the actual settings of the filters of the payload path which are permanently tracking the state of polarization of the optical signal.

After polarization demultiplexing, a differential phase decision suitable for the DQPSK modulated header format is applied followed by a header FEC decoder (H-FEC) adapted for a 120 bit code word. The timing of the shortened header processing ensures that the forward switch (FW-SW) control information is available if the first symbols of the data block leaves the payload DSP path.

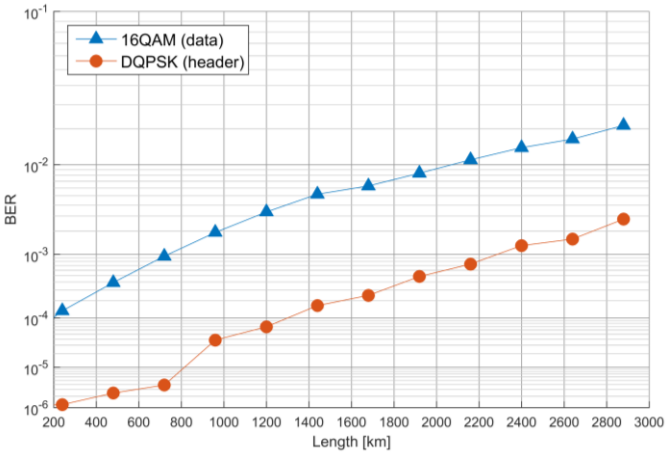

Fig. 7 Measured BER vs. transmission length for joint transmission of DP-DQPSK and DP-16 QAM

We have implemented the processing of 16 QAM payload and DQPSK header information in an experimental hardware setup to study the impact of the joint transmission of the two modulation formats over an optical fiber [25]. We chose a symbol rate of 32GBd to resemble a contemporary optical transmission system for 200Gb/s data rates. The digital signal processing for transmitter and receiver was performed offline on recorded data. Fig. 7 shows the measured BER vs. transmission length for a single channel experiment in an optical recirculating loop setup. The amplitudes of header and data symbol blocks were normalized to the same mean power value. Although the DQPSK signal passes only a simplified processing omitting frequency offset compensation and phase correction, the measured BER of the header information is always by a factor 10 below the respective BER of the 16QAM payload. After 2800 km, the BER of 16 QAM reaches the FEC limit. This corresponds to an OSNR of 17 dB. At this point we measure an SNR for DQPSK of 11 dB. Taking into account that the line width of the used lasers is about 100 kHz, this provides more than 3dB margin for an error free header transmission as shown in Sec. IX.

VIII. Forward Error Correction

In this section, we highlight in depth the forward error correction (FEC) of Optical Ethernet. As discussed previously, we require two types of FEC: The header needs to be reliably decoded with very low latency to be able to make a forwarding/drop decision of the payload. Additionally, we require a strong, yet computationally efficient FEC for the payload that can correct a large fraction of errors to achieve high transmission distances.

A. Proposed header forward error correction

The header needs to be reliably protected as it contains the routing information for the container payload. For this reason we cannot resort to inline protection of the header but require separate coding. As a decoding error in the header will lead to a complete container loss, we need a reliable coding scheme for the header. Furthermore, as the header size is less than 100 bit, we need a very strong code with high overhead. Unfortunately, FEC codes with block sizes in the range of 100 bit can require up to 7 dB higher SNRs than codes with the same overhead and longer block lengths (in the range of several 10,000 bit), as highlighted in Fig. 4 of [2]. As such a large SNR difference is difficult to achieve by purely changing the overhead, unless very low code rates sacrificing spectral efficiency can be tolerated, we resort to a combination of a robust modulation formats – highlighted in Sec. VII – and strong and short forward error correction for headers.

The header consists of 57 net bits which shall be encoded into 120 bits. This corresponds to an overhead of ≈110%. As the modulation format is based on differential coding with simple decoding, the errors at the output of the decoder are correlated. Such a correlation has been considered in [3], where Leong *et al.* utilize a bivariate binomial distribution to model the correlated error events. Using proper interleaving of the bits, we model the errors by the Markov model shown in Fig. 8. The state ε_0 indicates that a bit has been correctly received and the state ε_1 indicates an erroneous bit. The transitions $p_{ij} = \Pr(e_t = j | e_{t-1} = i)$ describe the bit error correlation where e_t denotes the error sequence at time index t. The probabilities p_{ij} can easily be estimated from measurements or simulations. The average pre-FEC error rate is then given by the steady state probability $\frac{1-p_{00}}{1-p_{00}+p_{10}}$.

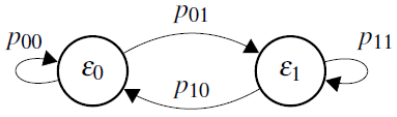

Fig. 8 Markov state model describing correlated error events

For encoding the header, we use a shortened 10-error correcting (120,57) BCH code. The header decoding failure P_F is then given by the probability that in a sequence of 120

state transitions, state ε_1 is visited more than 10 times. The random variable $T_n = \sum_{i=1}^{n} \mathbb{I}_{\{S_i=\varepsilon_1\}}$, where $\mathbb{I}_{\{\cdot\}}$ is the indicator function and S_i is the state visited after transition i, describes the number of visits to state ε_1. By marginalization, we get

$$P(T_n = \tau) = P(T_n = \tau, S_n = \varepsilon_0) + P(T_n = \tau, S_n = \varepsilon_1)$$

with

$$P(T_n = \tau, S_n = \varepsilon_i) = p_{0i} P(T_{n-1} = \tau - i, S_{n-1} = \varepsilon_0) + p_{1i} P(T_{n-1} = \tau - i, S_{n-1} = \varepsilon_1).$$

With the initial conditions $P(T_0 = 0, S_0 = \varepsilon_0) = 1$ and $P(T_0 = 0, S_0 = \varepsilon_1) = 0$, we can recursively compute $P(T_{120} = \tau)$ and the header decoding failure is then given by $P_F = 1 - \sum_{\tau=0}^{10} P(T_{120} = \tau)$. We illustrate the necessity of this model by a simulation example. We consider an AWGN channel additionally affected by Wiener phase noise modeling a receiver local oscillator laser with line width β and a baud rate of 32 GBaud. This very simple toy model is suited to illustrate the difference between errors caused by phase noise (single error after differential detection) and those by additive noise (correlated errors in two consecutive symbols). The simulation results are shown in Fig. 9. For comparison, we use a simple binomial model *not* taking into account the correlation between error events and estimate P_F as $P_F = 1 - \sum_{\tau=0}^{10} \binom{120}{\tau} P_b^{\tau} (1 - P_b)^{120-\tau}$ where P_b is the (empirically determined) input error rate. From the results, we can clearly see that the binomial model underestimates the failure probability, especially for the cases with little phase noise, and that the Markov model gives more accurate estimates, which we have verified by carrying out actual decoding trials. This model is important to get estimates on the required SNR needed for reliable transmission of the header and hence also of the payload.

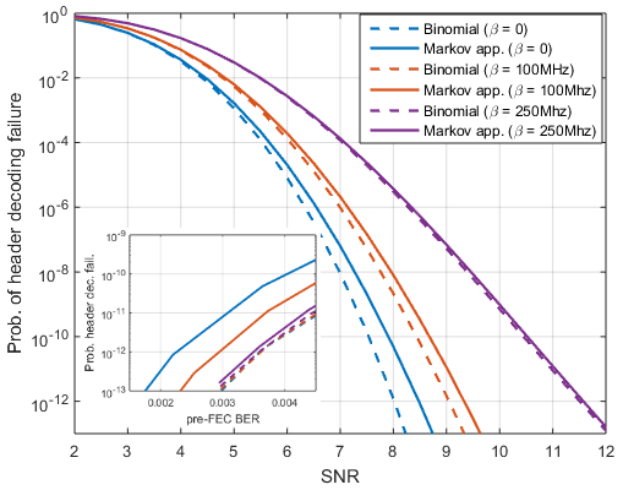

Fig. 9 Probability of erroneously decoding the header with DQPSK transmission over a Wiener phase noise channel with additive noise. Header failure is estimated using a simple binomial model (dashed lines) and a Markov model modeling correlated error events (solid lines)

B. Proposed payload forward error correction

In order to protect the payload, we need a high-performing FEC scheme that protects the payload data against transmission errors, leads to very low residual bit error rates, has low encoding and decoding complexity and inherently supports the concept of *multipoint-to-point* communications. To keep the cost of implementing receiver chips manageable we avoid the use of soft-decision decoding due to its increased complexity and decoder data flows [5]. Recently, several capacity-approaching coding schemes tailored for low-complexity hard-decision decoding have been proposed, like, e.g., the class of staircase codes [5]. These codes suffer however from the disadvantage that relatively large block lengths are required incurring a latency that cannot be accepted in all scenarios. The staircase codes of [5][6] for instance suffer from the drawback that every block consists of around 250,000 bits and the utilized windowed decoder requires joint processing of at least a few of such blocks. As the targeted container size is less than the size of even a single staircase code block, it becomes quickly obvious that we require a different type of coding unless we allow for coding across different containers. Coding across containers incurs however many difficulties in protocol design, buffering, and signaling as we have to make sure that always a well-defined number of containers is available at the receiver for joint decoding. We therefore opt for a coding scheme where the code words are confined within a container hence allowing for independent container decoding, suited for hard-decision decoding and promising high coding gains.

A coding scheme that fulfills these requirements is based upon spatially coupled LDPC codes [2][4]. We select a code from the class weakly coupled (unit memory) LDPC codes due to the excellent asymptotic properties, the low error floors and their low rate losses for this class of codes. A unit-memory spatially coupled LDPC code is a code that is defined by a *sparse* parity check matrix that has the form

$$\boldsymbol{H} = \begin{pmatrix} \boldsymbol{H}_0 & & & \\ \boldsymbol{H}_1 & \boldsymbol{H}_0 & & \\ & \boldsymbol{H}_1 & \ddots & \\ & & \ddots & \boldsymbol{H}_0 \\ & & & \boldsymbol{H}_1 \end{pmatrix}$$

of size $\dim \boldsymbol{H} = (L+1)m \times Ln$ and $\dim \boldsymbol{H}_i = m \times n$, i.e., the matrix consists of L vertically stacked and shifted copies of sub-matrices $(\boldsymbol{H}_0^T \quad \boldsymbol{H}_1^T)^T$. Every such stack is denoted *spatial position*, i.e., there are L spatial positions. A container $= (x_1 \quad \cdots \quad x_{Ln})^T$, consisting of Ln bits, which can be subdivided into sub-containers $\boldsymbol{x}_i$ of n bits each, has to fulfill the condition $\boldsymbol{H}\boldsymbol{x} = 0$, where the operations are carried out in the field of binary numbers $GF(2)$. As the matrix $\boldsymbol{H}$ contains $(L+1)m$ rows, the system of equations $\boldsymbol{H}\boldsymbol{x} = 0$ ideally defines $(L+1)m$ parity bits. Due to the particular banded structure of $\boldsymbol{H}$, solving the system of equations (i.e., encoding) can be easily accomplished, especially by selecting an appropriate $\boldsymbol{H}_0$. We distribute the $(L+1)m$ parity bits across the L spatial positions, where every spatial position corresponds to a sub-container (e.g. $L = 80$, if 80 sub-containers are used). We assign m parity bits to each of the first $L-1$ spatial positions and $2m$ parity bits to the last spatial position. By shortening, we ensure that the number of information bits per spatial position is constant. This parity bit

assignment is also highlighted in Fig. 2.

This code can be efficiently used for distributed *multipoint-to-point* encoding. If an intermediate networking node would only like to add a few sub-containers (i.e., spatial positions) to the container, it generates a codeword that only contains information in those spatial positions and zeros elsewhere (which is facilitated by the particular structure of $\boldsymbol{H}$) and generates parity for those spatial positions and the ones succeeding the last filled spatial position. We can limit the amount of generated parity-bits by using techniques similar to the ones we presented in [5] avoiding the effect of parity ringing. Due to the linearity of the code, the final container is filled by the modulo-2 sum (XOR) of the current container (where the sub-containers to be filled are empty) and the newly generated codeword. This process is displayed in Fig. 10.

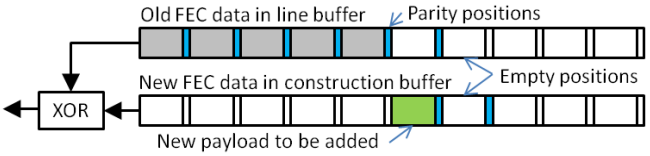

Fig. 10 Generation of a new container by addition of payload information and XOR of parity positions

At the receiver side, we employ hard decision decoding of the code. There exist multiple options for hard decision decoding and we employ Gallager's decoding algorithm B [7]. We can estimate the pre-FEC BER threshold using tracking of the error probability in the decoder assuming an asymptotic code performance. The performance on the code will depend on the design parameters and hence the sparsity of the parity-check matrix. For example, we may fix the number of "1"s in each column of $\boldsymbol{H}$ to be equal to d_v (and assume also a fixed number of "1"s in rows $m+1$ to Lm and then carry out the prediction over different realizations of the matrix. The predicted results are shown in Fig. 11, where the stars show the asymptotic performance of conventional LDPC codes under hard decision decoding and the squares represent the performance of the proposed unit-memory spatially coupled LDPC codes with different design parameters and 25% coding overhead. We can see that we can improve the decoding threshold by about 20% with the proposed codes. Additionally, the proposed structure greatly simplifies multipoint-to-point encoding and decoding. An additional advantage is the fact that we can deliberately leave some sub-containers (i.e., spatial positions) empty. As the empty positions are known at the receiver via the header, we can remove the errors from these positions prior to decoding and the spatial coupling of sub-containers allows us to beneficially use this information to improve the overall performance. At the transmitter, we can thus deliberately leave some sub-containers empty if some links suffer from bad performance or in the case of a node failure when a link length is increased. This enables an implicit rate versus reach tradeoff.

In the proposed OE scheme, latency plays a crucial role. Due to the fact that we carry out encoding and decoding only whenever required, we can significantly save upon latency with the proposed method by eliminating FEC re-encoding at intermediate nodes. Latency is dominated mostly by encoding and decoding latency. Encoding latency is negligible, as due to the spatial structure of the code, sub-containers are encoded individually and due to their small size, latency is well below 1µs (usually a few 10 clock cycles on an ASIC). The decoding latency is governed by two contributions: waiting for the decoding buffer to be filled (the so-called structural latency) and the time it takes to decode the codeword (the algorithmic latency). The former is in the range of 1µs while the latter can be in the range of a few µs of up to 15µs, depending on the decoder architecture, the implementation and the amount of parallelization that is used.

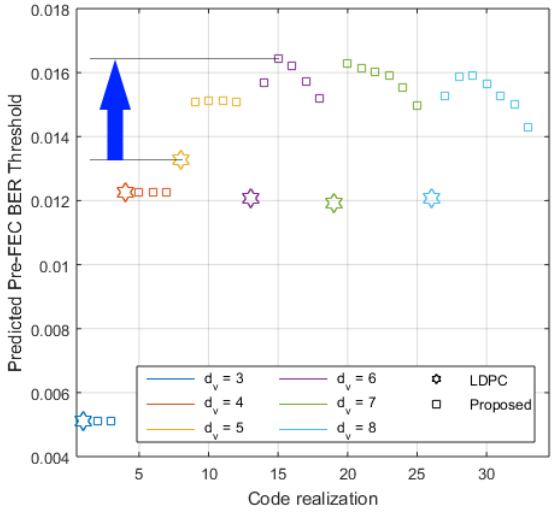

Fig. 11 Predicted Performance of proposed unit memory spatially coupled codes with 25% overhead under Gallager B decoding

IX. Transmission Performance

In the following section we want to point out the impact of experimentally evaluated constraints of state-of-the-art coherent transmission systems on the topology of OE networks.

Tab. 1 Properties of DP-16QAM and DP-32QAM formats. Netrate consideres 25% overhead for FEC

Mod.-format	Symbolrate (GBd)	Linerate (Gbit/s)	Netrate (Gbit/s)	Grid (GHz)
DP-16 QAM	32	256	200	37.5
	44	352	275	50.0
	54	432	338	62.5
	64	512	400	75.0
DP-32 QAM	32	320	250	37.5
	44	440	344	50.0
	54	540	422	62.5

For the data transmission we consider spectrally efficient modulation formats allowing 200 to 400 Gbit/s per wavelength. Tab. 1 summaries the properties of some candidates as reported in [10]. We propose to use dual polarization (DP) 16 QAM and 32 QAM formats as compromise of high spectral efficiency and low implementation penalties.

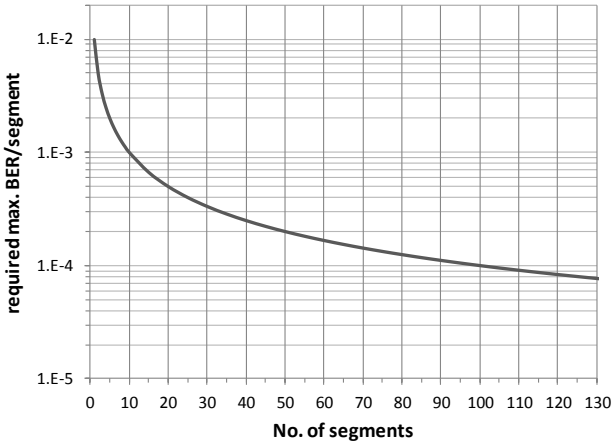

Fig. 12 Required max. BER per segment, targeting BER=1e-2 for cumulated system BER after concatenating N segments

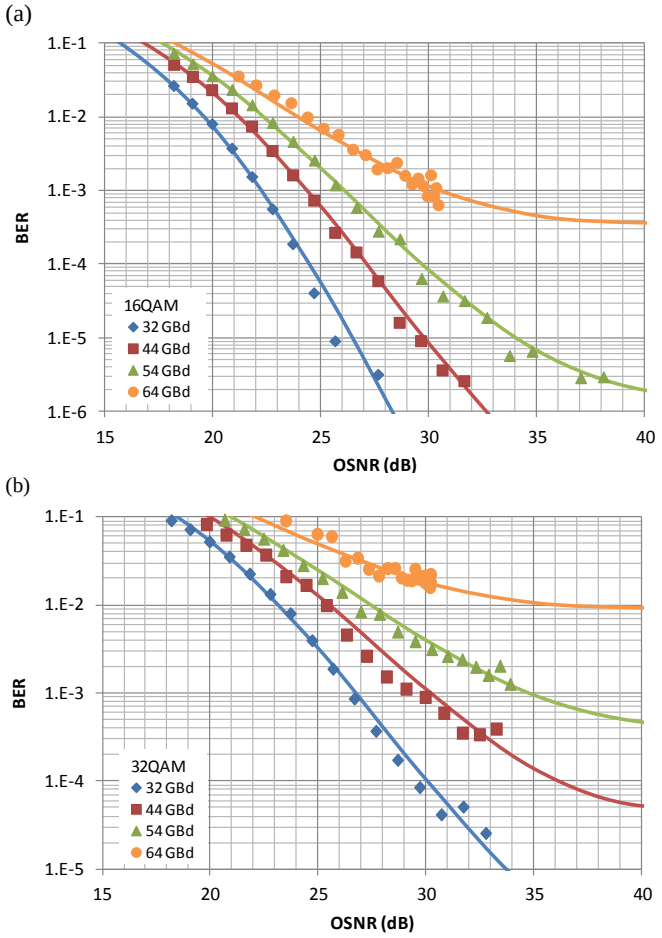

Fig. 13 Measured system performance (dots) in back-to-back measurement for DP-16QAM (a) and DP-32QAM (b) with various symbol rates extrapolated (solid lines) to high OSNR

The physical layer of OE has to cover a wide variety of possible segment lengths, depending on the distance of two OE-nodes. In addition, the concatenation of many OE-segments with multiple hard decision symbol regenerators without FEC has to be taken into account, which leads to additional decision errors. A good approximation for the cumulated system BER is the sum of the respective individual BERs in each OE-segment, when the latter are small. Fig. 12 shows the required BER of a single segment versus the total number of concatenated OE-nodes, if we are targeting a cumulated BER of 10^{-2} under the assumption of equally distribute OE-nodes.

From the performance measurements in back-to-back configuration (Fig. 13), the required OSNR for a given number of OE-nodes can be evaluated. The extrapolation towards high OSNR values takes into account that limitations of components, like electrical bandwidth or resolution of converters, generate additional implementation penalties, which finally restrict the performance of the system for high symbol rates. In the next step, the maximum achievable transmission distance can be estimated if fiber loss, amplifier noise figure and launched power are known. The results of this evaluation are summarized in Tab. 2 for two scenarios, considering a total number of 10 and 100 OE-nodes in the system.

Optical transmission for OE is adapted to the topology of the targeted system by the choice of modulation format and symbol rate. For short node distances below 100 km, DP-32 QAM can be used, allowing up to 1.6 Tbit/s OE-bus rate using a superchannel with 4 wavelengths. By cascading 100 nodes, a total distance of up to 8000 km can be reached. However, for larger node distances of up to 400 km, a more robust but less efficient modulation format like DP-16QAM is preferable. However, the achievable total distance in the two scenarios outperforms the targeted maximum system reach of 1500 km for OE.

Tab. 2 Required OSNR for BER=1e-3 (1e-4) representing a system with 10 (100) OE-nodes. Achievable node distance considers 80 km amplifier spans with 25 dB loss

		10 OE nodes		100 OE nodes	
Mod.-format	Symbol-rate (GBd)	req. OSNR (dB) 1e-3	max. node dist (km)	req. OSNR (dB) 1e-4	max. node dist (km)
DP-16 QAM	32	22.3	400	24.5	320
	44	24.4	320	27.3	160
	54	26.1	240	29.8	80
	64	30.5	80	n.a.	-
DP-32 QAM	32	27.8	160	30	80
	44	30.2	80	36	(< 80)
	54	34.8	80	n.a.	-

Today, the implementation of higher order modulation formats like 64 QAM at high symbol rates still suffers from imperfections of the available hardware. Further improvements of electronic converters and amplifiers will lower the BER floor at high OSNR values, thus also decreases the required OSNR for bit error rates around 10^{-3} and 10^{-4}, which are needed for the cascaded regeneration on the OE-bus with many nodes.

X. Network Performance

We investigated crucial performance and scalability aspects of our architecture by packet level simulation based on the ns-3 network simulator [12]. In this section, we report simulations that highlight the load performance of our architecture up to reasonably large node counts. Furthermore, we show how the inevitable container aggregation delay scales, if compared to a hypothetical hop-by-hop packet processing technology.

A. Simulation scenario

For performance evaluation, we load the network with a uniform full mesh traffic matrix, i.e. each node sends and receives an equal amount *A* of data to and from each other node on the bus. Since the architecture of the bus itself is unidirectional, we assume in a real system two counter propagating buses, each of them carrying a triangular part of the traffic matrix as shown in Tab. 3.

Tab. 3 Triangular traffic matrix

from/to node	1	2	3	…	N-1	N
1	0	A	A	…	A	A
2	0	0	A	…	A	A
…	…	…	…	…	…	…
N-1	0	0	0	…	0	A
N	0	0	0	…	0	0

In the simulation, the point-to-point load *A* is varied starting from zero and up to the load limit, where the packet loss ratio (PLR) on the most congested link exceeds the threshold of 10^{-3}. The traffic is generated as Poisson packet arrival process. Packet size is selected randomly according to a bimodal distribution of 40% 64 byte packets, 40% 1500 byte packets, and 20% uniformly distributed between both extremes. We assume packet buffers of 60 maximum sized packets at the client input ports.

B. Impact of container aggregation

In the simulation results shown in Fig. 14, we show the impact of container aggregation on the end-to-end delay. The simulation includes the waiting times to fill a container and the waiting times to find an empty space on the bus. It does not include processing times or fiber propagation delays along the bus. Finally, we take into account that with the uniform traffic matrix, the bus load is *not* uniform. On the contrary, the central link is the most congested and overload problems appear preferably in the node right in front of that central link. As a consequence, we show the delay and loss limit of the traffic that enters through the client interface at the most utilized node in the center of the bus.

Fig. 14 shows results for 10, 50, and 100 nodes and according to the container scheduling rules of Sec. V.A, that is, container aggregation alone (i), with timeout (ii), and with *multipoint-to-point* containers, i.e. sub-container addition to containers with common destination (iii). It is obvious that the container forwarding delay grows when the traffic load is approaching the bus capacity limit. But at the other end, the aggregation delay grows with *decreasing* load, since it takes more time to fill a container. The effect is amplified with *increasing* node count, where a given load is spread over more source-destination pairs.

The waiting time limitation safely avoids the low load problem. In Fig. 14 (center), with 50 nodes on the bus, the delay stays well below the 50µs timeout limit for a broad range of load values. Unfortunately, as an undesired side effect, the packet loss limit remains low only up until 80% of the bus capacity. The empty spaces in partially filled containers occupy the remaining 20%. The effect literally explodes with a larger node count, see Fig. 14 (right). Right from the beginning, the bus is crowded by containers that are occupied but almost empty. Empty slots on the bus become available only where upstream traffic leaves the bus. As a result, the timeout is essentially ignored. The containers have to wait for longer times, until they find a transmit opportunity. During that excess time, they collect more content, which partially compensates the capacity waste but at the price of increased latency.

The capacity waste problem disappears with the introduction of *multipoint-to-point* containers. It resolves not only the scalability problem for large node counts (Fig. 14 right), but it also reduces the delay for medium-sized buses (Fig. 14 center), because most sole packets find a transmit opportunity long before their timeout expires.

C. Benchmark with hop-by-hop packet processing

The introduction of transport containers was driven by the idea of an end-to-end FEC instead of a more conservative hop-by-hop FEC processing for the full bus capacity. The latter solution would resemble a chain of Ethernet switches that are connected by FEC protected point-to-point connections. It is obvious that the full packet processing of the transiting bus traffic in every node requires a higher effort than our solution, which is, however, not easy to quantify. At the other hand, our solution does not come for free. We are introducing additional delays due to the container aggregation time. To that end, we decided to limit our benchmark to the question whether we are equivalent in terms of throughput and delay.

To get a fair comparison, we assume hypothetical hop-by-hop processing nodes with separate output queueing on packet level, one queue for the transit traffic, and one queue for the local add traffic. The queues are served under strict priority for the transit traffic. We account for the *additional* processing delay by a fixed amount of 4µs per node which we derived from state-of-the-art technology for the OTN. This value is mainly attributed to the per hop FEC processing, which in our OE architecture occurs only once, namely at the final destination. As before, we investigate the traffic which is added right in front of the most congested link in the center of the bus. Due to the uniform traffic matrix, it still has to cross on average one half of the remaining nodes down the bus, which sums up to $25 \cdot 4\mu s = 100\mu s$ of additional processing delay for the 100 nodes case.

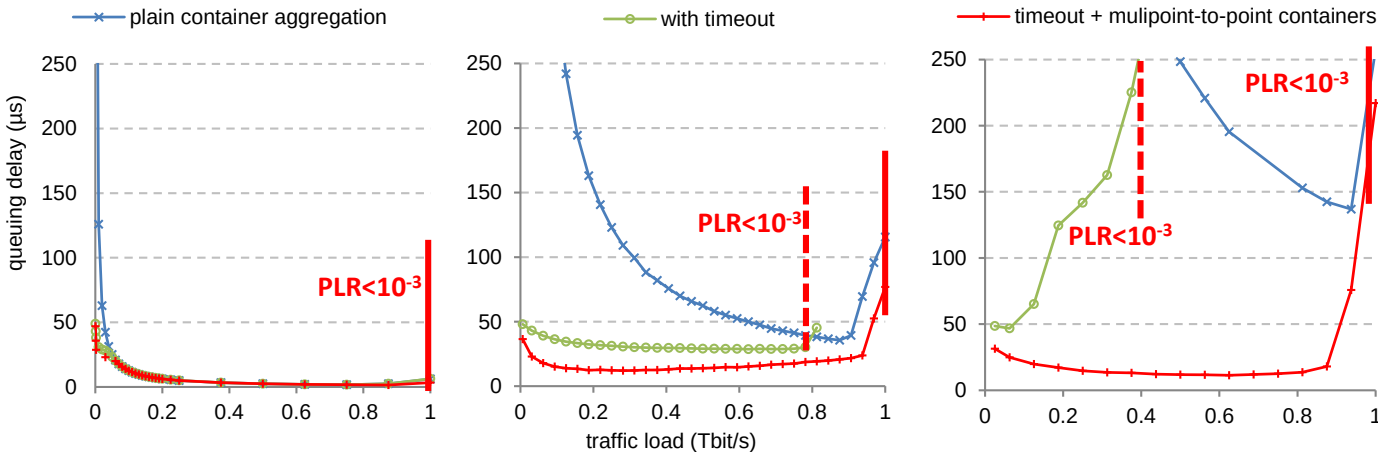

Fig. 14 Mean queueing delay as function of bus load for a 10, 50 and 100 nodes case (left to right).

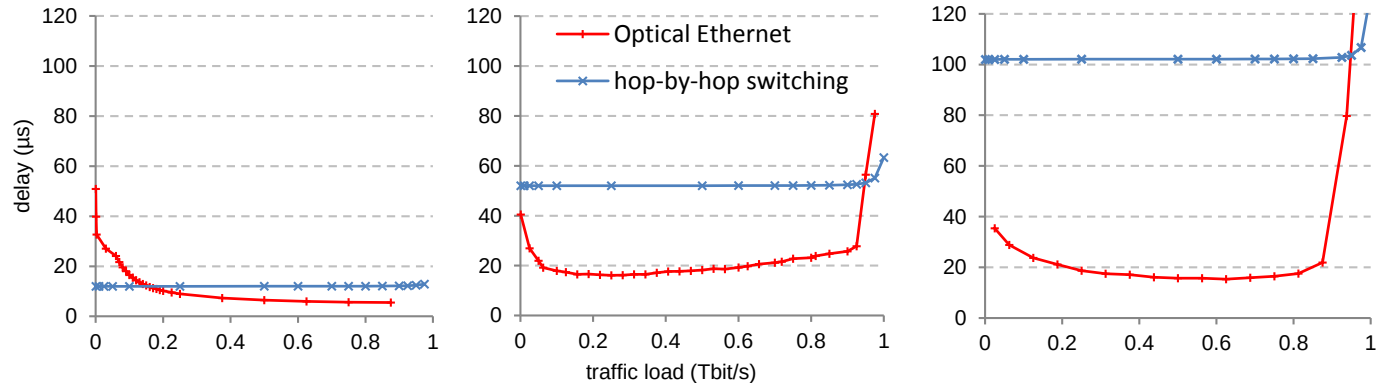

Fig. 15 Comparison between Optical Ethernet (OE) and hop-by-hop packet switching for a 10, 50 and 100 nodes case (left to right)

Fig. 15 shows the comparison of end-to-end delays for different network sizes. As expected, the additional processing delay of the hop-by-hop network outweighs the container aggregation times of our architecture. Only close to the capacity limit, the increase in queuing delay is slightly smaller. For a realistic interpretation of the results, it must be understood that we ignored the fiber propagation delay, because it is equal for both architectures. In absolute values, however, it is in the range of hundreds of microseconds for typical metropolitan area networks. It is obvious that our converged architecture, with its highly reduced processing effort in both transmission and switching parts, is at least equivalent in terms of delay and throughput if compared to a hypothetical chain of high capacity packet switches connected by series of high speed transmission sections.

XI. Conclusion

In this paper, we present Optical Ethernet (OE) as a converged optical transmission and switching architecture. It combines the flexibility of a packet switched network with leading edge optical transport performance. OE is intended to cover a metropolitan area by a linear bus with both ends attached to core network nodes. Logically, OE operates as a distributed packet switch. We replace costly hop-by-hop error correction by an end-to-end FEC, with only symbol regeneration in intermediate nodes. Our specific *multipoint-to-point* container aggregation fulfills the FEC requirement for large data chunks. At the same time, it preserves the switching granularity of source traffic down to 64 byte packets without sacrificing throughput, latency, or node count scalability. In addition, the architectural complexity of OE is limited to that of a typical transmission line card instead of a high capacity packet switch. We investigated and designed the whole range of functionalities, starting from modulation formats, signal processing, and forward error correction, up to packet segmentation/reassembly, queueing, routing, and congestion control. For verification of our design decisions, we performed analytical studies, simulations at different levels of abstraction, and lab experiments. We could show that with our design, typical metro requirements of up to 1500km at capacities of at least 1Tbit/s can be reached, and that the packet aggregation or traffic grooming process is able to utilize the capacity at its limit. We designed our architecture strictly for flexible, low latency packet transport.

References

[1] IEEE Resilient packet ring (RPR) access method and physical layer specifications, IEEE Standard 802.17, 2011.

[2] A. Leven and L. Schmalen, "Status and recent advances on forward error correction technologies for lightwave systems," *Journal of Lightwave Technology*, vol. 32, no. 16, pp. 2735-2750, Aug. 2014

[3] M.Y. Leong, K. J. Larsen, G. Jacobsen, S. Popov, D. Zibar, S. Sergeyev, "Dimensioning BCH codes for coherent DQPSK systems with laser phase noise and cycle slips," *Journal of Lightwave Technology*, vol. 32, no. 21, pp. 3446-3450, Nov. 2014

[4] L. Schmalen, V. Aref, J. Cho, D. Suikat, D. Rösener and A. Leven, "Spatially coupled soft-decision error correction for future lightwave systems," *Journal of Lightwave Technology*, vol. 33, no. 5, pp. 1109-1116, Mar. 2015

[5] B. P. Smith, A. Farhood, A. Hunt, F. R. Kschischang, and J. Lodge, "Staircase Codes: FEC for 100 Gb/s OTN," ," *Journal of Lightwave Technology*, vol. 30, no. 1, pp. 110-117, Jan. 2012

[6] L. Zhang and L. Schmalen, "Feed-Forward Staircase Codes," *Proc. International ITG Conference on Systems, Communications and Coding (SCC)*, Hamburg, Germany, Feb. 2017, preprint available online at https://arxiv.org/abs/1604.06574

[7] R. G. Gallager, *Low-Density Parity-Check Codes*, MIT Press, 1963

[8] SDH, https://www.itu.int/ITU-T/2001-2004/com15/otn/SDH-rec.html

[9] OTN, https://www.itu.int/ITU-T/studygroups/com15/otn/overview.html

[10] W. Idler et al., "Experimental Study of Symbol-Rates and MQAM Formats for Single Carrier 400 Gb/s and Few Carrier 1 Tb/s", in Proc. of OFC 2016, paper Tu3A.7

[11] F. Buchali et al., "Low Latency Digital Regenerator for Dual Polarization QAM Signals", in Proc. ECOC 2015, Valencia, Spain, paper Mo.3.3.5.

[12] NS3 documentation, online: https://www.nsnam.org/documentation/

[13] M.G. Hluchyj, M.J. Karol, "Queueing in High-Performance Packet Switching," IEEE JSAC, vol.6, no.9, December 1988

[14] G. Appenzeller, I. Keslassy, and N. McKeown, "Sizing router buffers," Proc. SIGCOMM. New York, NY, USA, 2004.

[15] Y. Tamir, G. L. Frazier, High-Performance Multi-Queue Buffers for VLSI Communication Switches, 15th Annual International Symposium on Computer Architecture, Honolulu, Hawaii, pp. 343-354, May 1988.

[16] N. Benzaoui, Y. Pointurier, T. Bonald, J.-C. Antona, Impact of the Electronic Architecture of Optical Slot Switching Nodes on Latency in Ring Networks, . J. Opt. Comm. Netw., 6(8), Aug. 2014

[17] G. Eilenberger, S. Bunse, L. Dembeck, U. Gebhard, F. Ilchmann, W. Lautenschlaeger, J. Milbrand, Energy-efficient transport for the future Internet, Bell Labs Technical Journal 15(2):147-167 · September 2010

[18] Zitian Zhang, Weiqiang Sun, Hao He, Weisheng Hu, Frame Assembly and Scheduling on Edge Routers in Fixed-Size Frame-Switching Networks, J. Opt. Comm. Netw. 5(1): 13-22, Jan 2013

[19] W. Lautenschlaeger, A. Mutter, S. Gunreben, Frame Assembly in Packet Core Networks – Overview and Experimental Results, Proc. 10. ITG-Fachtagung Photonische Netze, Leipzig, Germany, 2009

[20] M. Alizadeh, B. Atikoglu, A. Kabbani, A. Lakshmikantha, R. Pan, B. Prabhakar, and M. Seaman, Data center transport mechanisms: Congestion control theory and IEEE standardization. In Communication, Control, and Computing, 2008, 46th Annual Allerton Conference on

[21] Dolzer, K.; Gauger, C.; Späth, J.; Bodamer, S.: Evaluation of Reservation Mechanisms in Optical Burst Switching. AEÜ International Journal of Electronics and Communications. Vol. 55, No. 1, 2001.

[22] I. Chlamtac and A. Gumaste, "Light-trails: A solution to IP centric communication in the optical domain," Online publication: February 17, 2003, Springer-Verlag Berlin Heidelberg 2003

[23] H. Harai, H. Furukawa, K. Fujikawa, T. Miyazawa, N. Wada, Optical Packet and Circuit Integrated Networks and Software Defined Networking Extension, J. Lightwave Technology, Vol. 32 No. 16, 2014

[24] R. Veisllari, S. Bjornstad, J.P. Braute, K. Bozorgebrahimi, C. Raffaelli, Field-Trial Demonstration of Cost Efficient Sub-Wavelength Service Through Integrated Packet/Ciruit Hybrid Network, J. Optical Communication, Vol. 7, No. 3, 2015

[25] R. Dischler et al, "Embedded In-Band DQPSK Signaling within n-QAM Data Transmission," in Proc. ECOC 2016, Düsseldorf, Germany, paper W.4.P1.SC3.5