Kamchevska, V., Cristofori, V., Da Ros, F., Guo, B., Jackson, C., Fagertun, A. M., ... Galili, M. (2017). Synchronization in a Random Length Ring Network for SDN-Controlled Optical TDM Switching. *IEEE/OSA Journal of Optical Communications and Networking*, *9*(1), A26-A34. [A26]. https://doi.org/10.1364/JOCN.9.000A26

Synchronization in a Random Length Ring Network for SDN-Controlled Optical TDM Switching [Invited]

Valerija Kamchevska, Valentina Cristofori, Francesco Da Ros, Bingli Guo, Chris Jackson, Anna M. Fagertun, Sarah Ruepp, Reza Nejabati, Dimitra Simeonidou, Lars Dittmann, Michael Berger, Leif K. Oxenløwe and Michael Galili

***Abstract*—In this paper we focus on optical TDM switching and its main distinguishing characteristics compared to other optical subwavelength switching technologies. We review and discuss in details the synchronization requirements that allow for proper switching operation. In addition, we propose a novel synchronization algorithm that enables automatic synchronization of SDN-controlled all-optical TDM switching nodes connected in a ring network. Besides providing synchronization, the algorithm can also facilitate dynamic slot size change and failure detection. We experimentally validate the algorithm behavior and achieve correct operation for three different ring lengths. Moreover, we experimentally demonstrate data plane connectivity in a ring network composed of three nodes and show successful WDM-SDM transmission and switching of data bursts when using the proposed algorithm to provide synchronization.**

***Index Terms*—Synchronization, optical switches, time division multiplexing, software defined networking.**

I. Introduction

Traffic in data center networks (DCNs) is growing at a fast pace [1] and gaining increasing attention over the past few years. For modern DCNs that run cloud-based applications, bandwidth utilization and energy efficiency are becoming crucial for sustainable growth. For these reasons, optical DCN architectures have been proposed on several occasions. In [2,3] we proposed time division multiplexed (TDM) circuit switching in the optical domain to be deployed as one of the switching dimensions of multidimensional switching nodes connected in a ring topology. The main motivation behind the use of optical TDM is the ability to accommodate bursty traffic by using a single wavelength shared among different connections, thus providing better resource utilization. Unlike optical packet switching (OPS) and optical burst switching (OBS), TDM switching allows for circuit-based transmissions, thus avoiding contention and eliminating the need for buffering within the network. As time information is used to control the switches, precise synchronization among the nodes is necessary in order to perform accurate switching, In ring networks an additional constraint is introduced if *NxN* switches are used to perform the subwavelength switching i.e. simultaneous add and drop can be performed at each node only if the propagation delay along the ring is an integer multiple of the time slot duration.

Moreover, synchronization is a critical aspect for any type of optical subwavelength switching using *NxN* switches, as it allows that data arriving on the different switch inputs is aligned to the discrete switching time of the switch. Although crucial, in most demonstrations so far the problem of synchronization is approached by either using synchronizers or pre-engineered fiber links. In order to address this problem, in [4], we proposed a novel algorithm that allows for automatic synchronization in a ring network composed of software defined networking (SDN) controlled optical TDM switches that simultaneously add and drop bursts of data. The algorithm works by estimating the propagation delay along the ring and deciding on a slot size.

In this paper, we elaborate on the optical subwavelength technology envisioned in our previous work and its synchronization requirements. Based on this, we present the detailed implementation of the proposed synchronization algorithm and discuss the provided benefits. We verify the algorithm behavior and additionally experimentally investigate the performance in the data plane when the proposed algorithm is used for synchronization. Successful SDN-enabled switching is achieved with an extended OpenDaylight controller that

Manuscript received May 23, 2016.

V. Kamchevska, V. Cristofori, F. Da Ros, L. K. Oxenløwe and M. Galili are with the High Speed Optical Communications Group, Department of Photonics Engineering, Technical University of Denmark, Ørsteds Plads, B343, Lyngby, 2800, Denmark (e-mail: vaka@fotonik.dtu.dk).

B. Guo, C. Jackson, R. Nejabati and D. Simeonidou are with the High Performance Networks Group, University of Bristol, Bristol, BS8 1TH, United Kingdom.

A. M. Fagertun, S. Ruepp, L. Dittmann and M. Berger are with the Networks Technology and Service Platforms Group, Department of Photonics Engineering, Technical University of Denmark, Ørsteds Plads, B343, Lyngby, 2800, Denmark.

configures each node to drop one burst of data. Furthermore, we show that the scheme is both wavelength division multiplexing (WDM) and space division multiplexing (SDM) compatible by using 15 WDM channels with 10 Gbit/s slotted OOK data and 7-core fiber for transmission. We successfully switch TDM slots at all wavelengths using the same control signal.

The remainder of this paper is organized as follows: in Section II we give a brief overview of the concept behind the optical subwavelength switching considered in this work i.e. optical TDM switching and present an overview of the synchronization requirements. Moreover, we review similar work and compare and discuss the various approaches to synchronization. In Section III, we present the proposed synchronization algorithm and elaborate on its implementation and validation. In Section IV, we present the results of the experimental demonstration for the data plane operation when synchronization is provided by using the implemented algorithm. At last, in Section V we conclude the paper by summarizing the proposed synchronization algorithm and the results for its validation and data plane operation.

II. Optical Subwavelength Switching and Synchronization

The main motivation behind optical subwavelength switching is the possibility to utilize the available bandwidth efficiently by sharing it in the time domain among several connections. Often full wavelength channels are underutilized if they are serving bursty traffic which requires far less capacity than provided. Optical circuit switching (OCS) technologies provide connectivity with coarse wavelength granularity and slow switching speed thus leading to low bandwidth utilization. On the other side, optical subwavelength switching technologies envision the use of fast optical switches that allow for enhanced traffic adaptation and hence aim to deliver high bandwidth utilization. For this reason, several different technologies have been proposed over the years such as OPS, OBS, optical time slot switching etc. A comparison between these technologies can be easily provided focusing on several main characteristics i.e. resource reservation (circuit vs. packet based, static vs. dynamic allocation), medium access (slotted vs. unslotted), burst size (fixed vs. variable), approach to contentions (contention-less vs. contentions exist and are resolved by the use of optical buffering, optical-electrical conversion followed by electrical buffering, deflection routing etc.), control information distribution (in-band/out-of-band headers vs. time as control information), synchronization mechanism (synchronizers vs. global synchronization) etc. Based on these features, the concept of optical TDM switching is described and contrasted against other optical subwavelength technologies.

A. Optical TDM switching

Different optical buffering technologies and hybrid approaches in which data is buffered in the electrical domain and retransmitted have been proposed. However, the need for extra components, ultimately leading to higher component cost and additional control overhead has shown to be detrimental for the lack of deployment. In order to provide support for optical subwavelength switching while at the same time avoiding the need for optical buffering in the network, we have previously proposed the concept of optical TDM switching [2,3]. Comparing optical TDM switching against OPS and OBS, the main difference lays in the nature of the connections. Unlike the connectionless data exchange of OPS and OBS which in turn requires optical buffering within the network to resolve contentions, optical TDM switching envisions circuit-oriented data transmission where connections are established apriori and resources are allocated in advance. Hence, contentions exist only during the allocation process where they can be resolved by buffering in the electrical domain at the end sides and do not exist after resources are allocated. This fundamental difference with other optical subwavelength technologies offers an indispensable benefit as it avoids the need for optical buffering.

Additionally, time driven switching is envisioned, meaning that the switches in the network will be reconfigured based on two things: the current time slot at each node and the control information for that time slot, which could be delivered by an SDN controller. Thus, by removing the need to transmit control information using packet headers, better bandwidth utilization is enabled and the need to have bitwise synchronization and header processing at each node is eliminated. In this way, the control is completely decoupled from the data plane, the network is fully transparent to the traffic that is passing through it and data is processed only at the end sides. Moreover, the propagation delay is the only delay experienced once the connection is established as no intermediate processing is done at any of the nodes.

Time sharing of a single wavelength is done by having slots arranged in a periodic frame structure. In order to provide trade-off between rather inflexible static connectivity and control-wise computationally demanding completely dynamic connection establishment, a mixed approach is envisioned, where an initial schedule of slot assignments exists, but can be modified on demand. By repeating the schedule in a periodic fashion, SDN control is required to initially configure the network and to convey information of any upcoming changes. Furthermore, we envision delivering the SDN control information on a (few) frame rather than slot basis, such that by the next update from the controller, sufficient control information is delivered to the switch for the upcoming slots.

In summary, the proposed optical subwavelength switching allows for circuit-based connectivity with apriori resource reservation and contention-less communication without the need of buffering within the network. The medium access is slotted and all bursts have the same size, although the common burst size can be dynamically changed. A global synchronization method is deployed to allow for synchronization among nodes and the switches in the network are controlled based on the time slot status and the control information distributed by an SDN controller.

B. Synchronization requirements

All optical subwavelength switching technologies such as OPS, OBS or optical circuit-based time slot switching that use *NxN* optical fast switches require synchronization

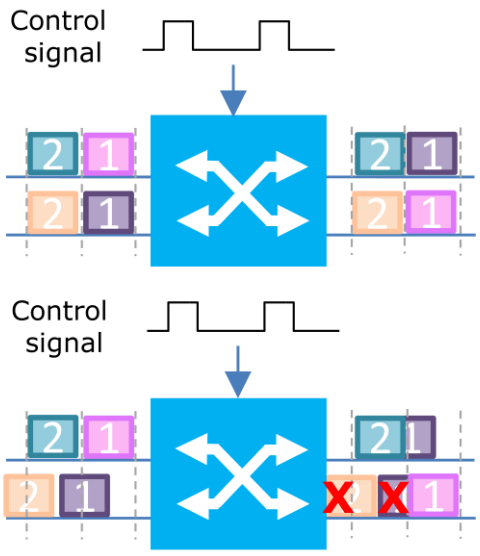

Fig. 1. Behavior of a 2x2 optical switch when signals at both inputs arrive aligned (top) and misaligned (bottom)

among signals arriving at different inputs. As these switches have switching time of few nanoseconds and switch in discrete time intervals, data arriving on all inputs of the switch has to be aligned, so that simultaneous switching on all inputs can occur properly. Figure 1 illustrates the behavior of a single 2x2 optical switch under two possible scenarios: data arriving on the different inputs aligned with each other and with the control signal (top); and data arriving aligned to the control signal on one input only (bottom). In order to avoid an improper behavior, it is necessary to deploy a control mechanism that will allow for synchronization among the nodes in the network.

There are different ways to provide synchronization in a network composed of fast optical switches. One way is to use synchronizers for each input that can dynamically compensate for delay offsets [5]. A common synchronizer scheme can be dynamically reconfigured and consists of 2x2 fast optical switches and fiber delay lines (FDLs). The length of the FDLs between the switches follows an exponential sequence; that is, the first FDL equals ½ slot duration, the second FDL equals ¼ slot duration, etc.

Another approach is to use global synchronization and distribute a control signal often denoted as clock or trigger along the network in order to provide each node with a time reference. Depending whether the clock/trigger distribution accounts for the propagation delay among the nodes or not, two different methods can be identified. If the clock is distributed along the same fiber infrastructure used for data transmission then the clock received at each node will be delayed exactly by the propagation delay between the node that distributes the clock and the receiving node. Another way is to distribute the same clock to all nodes by for example using local network for clock distribution [6] or the Global Positioning System (GPS) [7] allowing for the same time reference to exist in each node, however not accounting for the link propagation delay among nodes in the network.

The two global synchronization methods have significant

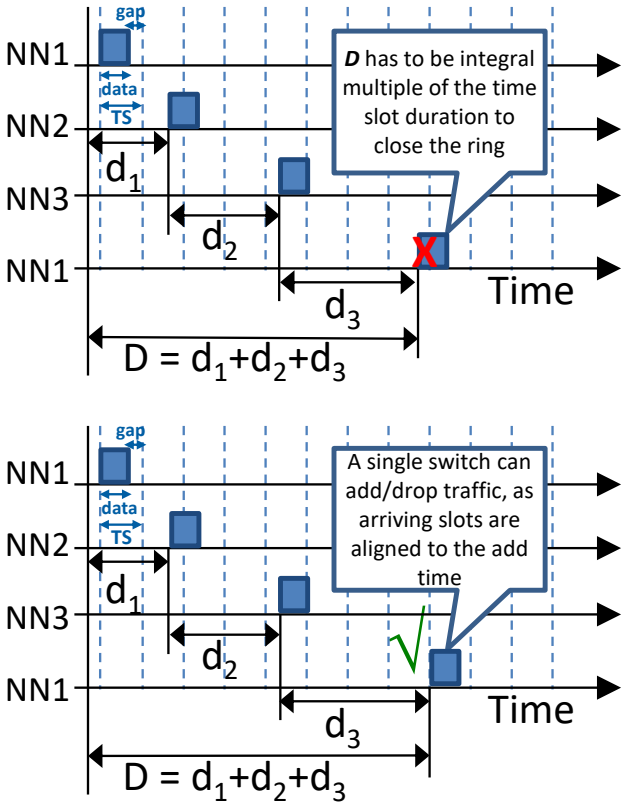

Fig. 2. A ring network with length that is not (top) and that is (bottom) an integer multiple of the time slot duration

impact on network operation and control. Using a method that does not account for the propagation delay between nodes makes synchronization among the nodes more challenging. This comes from the constraint that each link in the network would have to provide delay that is an integer multiple of the chosen time slot duration. Although, pre-engineered fiber links could be deployed, placing limitations on the fiber length or alternatively on the time slot duration is not practical as it makes installation and maintenance more complex. Furthermore, even if we assume that links are pre-engineered, in order to use centralized software defined control plane for configuring the switches it is still necessary to take into account the propagation delay on each link. For example, if a connection has to be established between two nodes, the SDN controller would have to know exactly when the sent burst of data will arrive in each of the intermediate nodes, in order to be able to generate the corresponding control signals for configuring the switches. From this perspective, using a synchronization scheme that accounts for the propagation delay among nodes greatly simplifies the network control.

Additional considerations on synchronization can also be made depending on the network topology. Synchronization approaches that do not account for the propagation delay are rather independent from the network topology, however the more complex the topology is, the more constraints would exist for the propagation delays on each link. Approaches that do consider the propagation delay are largely dependent on the network topology, as this also

defines the clock distribution network. Thus, providing synchronization in a star or tree network is relatively easy, however other topologies may face additional limitations. For example, in a ring network composed of nodes that deploy *NxN* switches and allow for simultaneous add and drop of traffic, the total propagation delay along the ring has to be an integer multiple of the time slot duration in order to have proper switching in all nodes.

The problem of synchronization in a ring network is illustrated in Fig. 2 assuming three network nodes connected in a ring. The links between the nodes, denoted as d_1, d_2 and d_3 are of arbitrary length. A time slot (TS) is composed of a portion allocated to the transmission of the data burst and a gap used to account for the switch reconfiguration time and additional inaccuracies as it will be discussed below. Each node deploys an *NxN* switch to both add and drop bursts, thus the add and drop time at each node have to coincide. For simplicity, if NN1 wants to add a burst and receive the same burst after propagating along the ring, unless the arriving slot is aligned to the add time of the switch, improper switching will be performed. Hence, the propagation delay along the ring has to be an integer multiple of the chosen slot duration in order for bursts to be switched correctly.

C. Overview of Previous Work on Synchronization

Although a lot of work has been done in the field of optical subwavelength technologies and various aspects of performance evaluations, the problem of synchronization has been mainly neglected. Most works on OPS and OBS assume the use of synchronizers at each input of all the fast optical switches deployed [5,8]. However, this approach may significantly affect the signal integrity. The cascaded switching in the synchronizers results in high insertion loss and the limited suppression ratio of the switches can impair the performance due to crosstalk.

Other examples exploit global synchronization with pre-engineered fiber link lengths and use either GPS [7] or a master clock reference [6,9] that is distributed in the network, without accounting for the propagation delay among nodes. Even though theoretically this enables accurate operation, in practice it introduces severe limitations on the network flexibility, planning and maintenance as fiber lengths must be accurately controlled.

In [10], an algorithm is presented for synchronization of time slots in multi-ring networks by separating the add and drop time at each node. This approach works well if tunable wavelength converters combined with an arrayed waveguide grating (AWG) are used for add/drop, however it does not support a single *NxN* fast switch performing simultaneous add/drop of bursts.

III. Proposed Synchronization Algorithm

The main network scenario for optical subwavelength switching in this work is intra data center network, based on the architecture proposed in [2,3] where the optical TDM switches are connected in a ring. The focus is therefore on the synchronization requirements for such a network. However, although the algorithm described in this section has been proposed for optical TDM switching, it can also be used to provide synchronization for any type of OPS or OBS with slotted medium access i.e. where transmissions are allowed only at the beginning of time slotted intervals, regardless of the switch control mechanism, resource reservation and the contention resolution approach taken.

A. Algorithm Implementation

The detailed algorithm behavior is illustrated in Fig. 3 and an overview of the implemented synchronization plane is given in Fig. 4. The algorithm is implemented on a field programmable gate array (FPGA) board. There is one node in the ring denoted as master node that is responsible for running the algorithm and providing synchronization. The master node sends out a signal, *sync_out* that is used to measure the propagation delay along the ring. Considering that DCNs are a closed and well controlled environment and that the propagation delay measurement is done at the physical layer, the ring length can be measured reliably and accurately.

After *sync_out* is sent out, a counter is started to mark the time elapsed. When the *sync_out* signal is received after propagation along the ring, the counter is stopped and the current status of the counter is used to estimate the propagation delay, *D*. The accuracy with which the propagation delay is measured depends on the clock frequency with which the counting is done. For a clock with frequency *f*, the period $T=1/f$ determines the measurement precision. In this specific implementation, a 200 MHz clock is used for counting, meaning that the accuracy with which the propagation delay is measured is 5 ns. In order to account for this accuracy it is necessary to accommodate this time in the switching gap. As the duration of the switching gap is mainly defined by the physical effect used for switching as well as the driving electronics used, it is desirable that any additional time overhead due to synchronization accuracy is ideally smaller or at least on the same order. In our case, the switching time of the fast optical switches is ~10 ns, thus we consider any additional gap contribution acceptable if it is on the same time scale. By using a higher frequency clock for performing the processing on the FPGA, the accuracy of the measurement can easily be narrowed down to nanosecond range, thus further improving the precision. However, although accuracy below 1 ns can offer some other advantages as it will be discussed below, there is no significant benefit for the switching gap, as it would still be dominated by the rise/fall time of the optical switch.

The main principle behind the algorithm is that once the propagation delay *D* is known, one should find slot solutions with duration *TS* such that *D* is an integer multiple of *TS*. Thus, after estimating *D*, the acceptable time slot durations are calculated as

$$\begin{aligned} &for\ (TS_{min} < TS_i < TS_{max}) \\ &\quad if\ (mod(D, TS_i) == 0) \\ &\quad\quad Save\ TS_i; \end{aligned} \tag{1}$$

where TS_{min} and TS_{max} are predefined boundaries on the range of acceptable slot sizes which can be given as input parameters to the algorithm and the saved TS_i are the

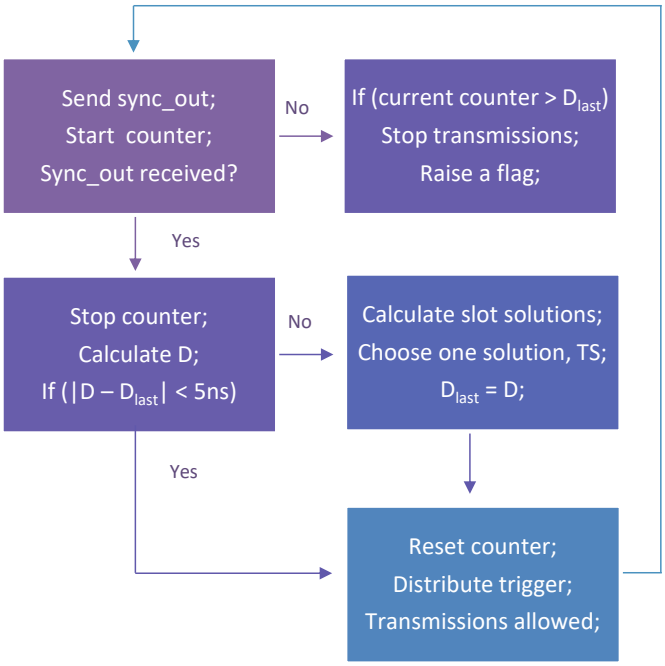

Fig. 3. Diagram of the proposed synchronization algorithm

possible solutions for which the initial condition is satisfied. It is worth mentioning that depending on the desired limits for the time slot size, it may be sometimes difficult to find a broad range of solutions or to find solutions at all. However, this can easily be avoided by using any of the solutions of the neighboring propagation delays under the cost of increasing the switching gap for the accuracy of the measurement. In addition, one may consider doing this also in cases when there are available solutions just to increase the flexibility of the system i.e. allow using a better suited slot size. From this perspective, having a better accuracy of the propagation delay measurement is a nice feature that enables improved flexibility and a wider solution space, while still providing only a relatively small contribution to the switching gap.

Once the acceptable solutions are found, one can be chosen to be used in the data plane. In this implementation, a slot size is randomly selected from the solution space and used for all transmissions within the data plane. The gap is fixed and set to 25 ns, although any ratio of useful data/gap can be implemented. For example, one may want to increase the gap size if solutions of neighboring propagation delays are used. Furthermore, the found slot solutions could be disclosed to an SDN controller allowing for the decision on the slot size to be done by an authority that has broader view on the network with respect to existing connections or traffic demands. The SDN controller would then convey the information of the chosen slot size to the master node responsible for synchronization, and thus to each TDM switch in the ring. This can provide the benefit of deciding on a slot size that will allow for improved performance in terms of lower blocking probability, lower latency etc. Moreover, knowing the possible slot solutions, the SDN controller can decide upon a dynamic slot size change and replace the selected solution with another one from the

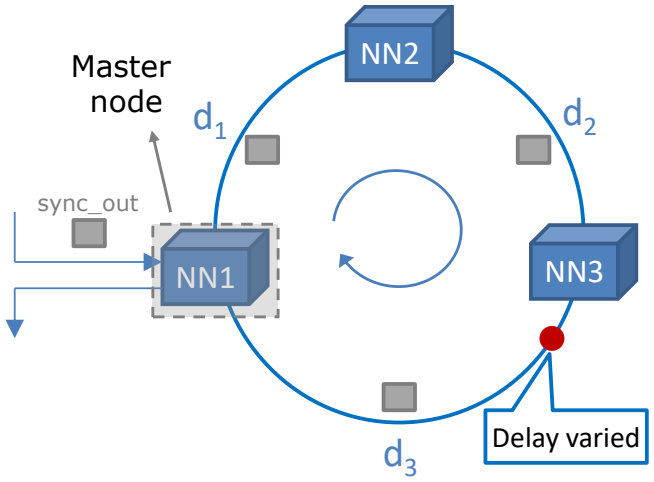

Fig. 4. Overview of the implemented synchronization plane

acceptable slot sizes that in the specific network conditions would lead to improved performance. Hence, the algorithm not only provides support for synchronization but it can also facilitate network dynamicity and flexibility.

After a slot size has been chosen and the last measured propagation delay has been saved by assigning it to the variable D_{last}, we reset the counter and allow that data plane operation can start. In order to synchronize all nodes in the ring, the master node is responsible for distributing a clock/trigger for synchronization. There are different ways to implement this. One is to distribute a trigger that would be used to trigger the counting process at each node based on a local clock. Another way is to distribute a clock with periodicity as the chosen slot size, allowing that each node can implement a counter based on the received clock. In both cases, each node would start counting with delay corresponding to the propagation delay. Thus, the counter status can basically be used as an identifier for the different slots. This allows for unique addressing of the slots, meaning that at one point in the network there will be only one slot 1 and a TDM connection can be uniquely addressed by the allocated slot and the nodes through which it is passing. Hence, in order to configure the network for this connection to be established, the SDN controller should configure all the TDM switches on the chosen path in the required configuration for slot 1. In this way, the resource allocation process is greatly simplified and no care has to be taken for the length of the links. Assuming a frame with duration equal to the propagation delay D, which depending on the chosen slot size will consist of a different number of slots, each node should keep entries for the slot configuration for at least one frame.

For this implementation the *sync_out* signal is sent out whenever it is received, allowing for a continuous measurement of the propagation delay and thus enabling automatic resynchronization if the propagation delay along the ring changes. In addition, if the measured propagation delay is within 5 ns from the last measured propagation delay, no action is taken, since this can be accommodated in the switching gap. If the measured propagation delay is offset by 5 ns or more from D_{last}, the slot solutions are recalculated and the whole procedure is repeated.

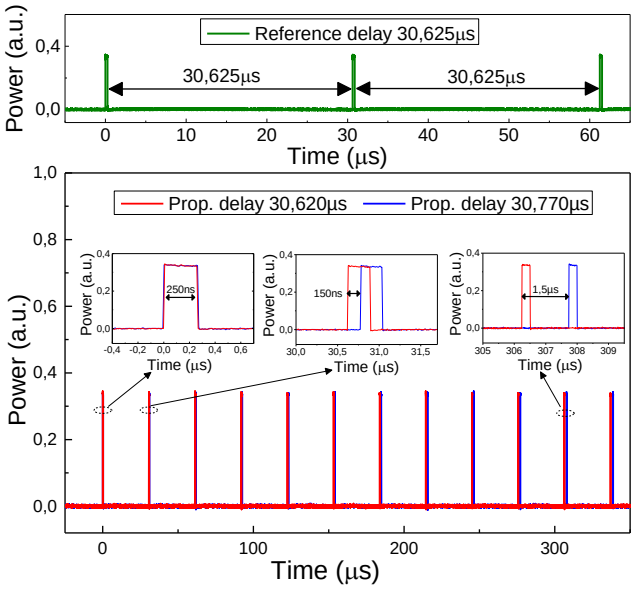

Fig. 5. Time domain traces of the received *sync_out* signal after propagation

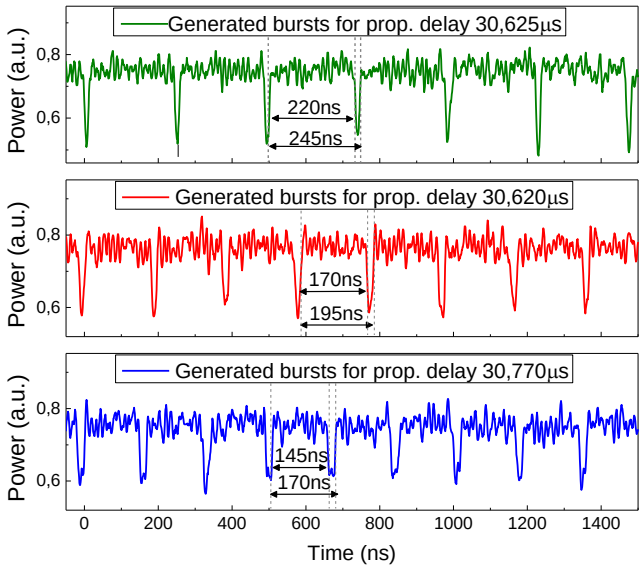

Fig. 6. Time domain traces of the generated bursts for the different propagation delays

If we consider the speed of resynchronization it can be seen that in the current implementation where the *sync_out* signal is sent out whenever it is received, it would take less than two roundtrips in the worst case to detect changes. However, this can easily be improved by increasing the frequency at which this signal is sent and make that independent of the reception after propagation. In the worst case, for a period that is negligible with respect to the propagation delay, the detection time can be reduced to one roundtrip. However, as the main application of this technology is in DCNs, where the ring length is relatively short i.e. in the order of tens of kilometers, a reaction time of at most one roundtrip is believed to be rather acceptable.

The main reason for variation of the propagation delay is the temperature dependence of the refractive index of optical fiber. For a standard optical fiber this delay variation is on the order of 40 ps/°C/km [11,12]. Considering that DCNs are well-controlled environment [13], extreme temperature variations are not expected. Hence, even if we assume 10 °C fluctuation and 10 km ring propagation, this would result with only 4 ns variation which can easily be accommodated in the switching gap.

Furthermore, the algorithm can also be used for joint synchronization of many wavelengths carrying slotted data. For wavelength span of 30 nm, typical fiber dispersion of 20 ps/nm/km and fiber length of 10 km, the delay variation is 6 ns, which can also be easily accounted for. This indicates that common synchronization is feasible for WDM systems and accurate switching can be performed simultaneously on all wavelengths using the same control signal for switching. Thus no dispersion compensation is needed for proper switching. Additionally, besides being WDM compatible, the synchronization method is also SDM compatible as it can provide joint synchronization for slotted data carried in all MCF cores.

Finally, the algorithm can also be used to facilitate failure detection in the network. Namely, if the *sync_out* signal is not received after propagation and the counter status has exceeded the value for the last measured propagation delay, then it is clear that something has changed in the network. At this point, transmissions can be stopped so that improper switching is avoided. Alternatively, the absence of the *sync_out* signal can be used to raise a flag, indicating possible link/node failure to the SDN controller, thereby assisting protection and restoration schemes.

B. Algorithm Validation

In order to verify that the implemented algorithm operates as desired, a simple validation test is performed using a ring composed of three nodes as shown in Fig. 4. NN1 is the master node and runs the algorithm. To interconnect the nodes we use a single 2-km long 7-core multicore fiber. A set of two cores is used on each link; one core for transmitting the *sync_out* signal and another for transmitting the data and trigger. The generated *sync_out* signal from the FPGA is used to modulate a continuous wave (CW) laser at 1549.72 nm. The propagation delay along the ring is varied between NN3 and NN1 to verify that the algorithm can provide automatic synchronization for three different ring lengths. The time domain traces of the received signal after propagation for the three cases are shown in Fig. 5. It can be seen that the periodicity of the received signal corresponds to the reference measured propagation delay (top) and increasing/decreasing the delay results with modified period (bottom).

After measuring the propagation delay, the selection of the correct time slot duration is verified. A transmitter sending a burst of data in each slot is assumed at NN1 and the time domain traces of the bursts right after generation are recorded. Figure 6 illustrates the time domain traces of the generated bursts for the three different propagation delays measured. It can be seen that for propagation delays of 30.625 µs and 30.770 µs, the algorithm has chosen a slot size for which the propagation delay is exactly an integer multiple of the time slot size i.e. 125 and 181 time slots, respectively. For the propagation delay of 30.620 µs the algorithm has selected a slot size such that the ratio

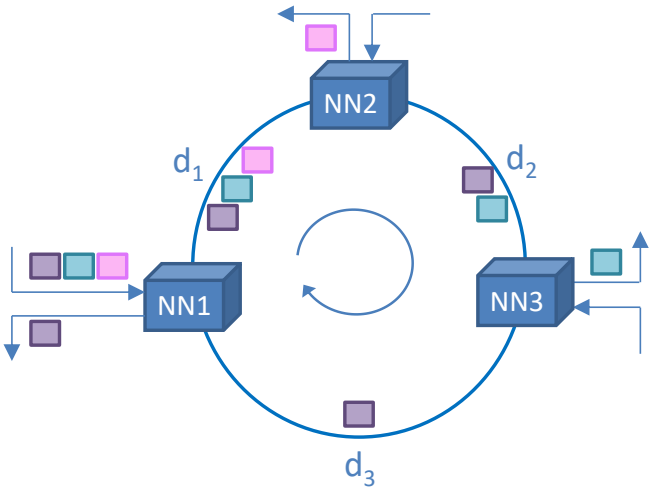

Fig. 7. Overview of the implemented data plane connections

propagation delay/slot size is 157.026. An integer number of slots i.e. 157 with duration 195 ns would be a correct solution for a propagation delay of 30.615 µs. As mentioned in the previous subsection, when the algorithm cannot find a solution for a specific propagation delay within the acceptable slot limits, then a slot size is chosen from the solutions of the neighboring propagation lengths measured. Since the propagation delay is measured with 5 ns accuracy and slots are accepted in the range from 100 ns to 1 µs, the chosen solution indeed conforms to the expected behavior (30620 ns has no factors in the range of 100-1000 ns).

IV. Data Plane Operation

After verifying that the algorithm is correctly deciding on a slot size, the performance in the data plane is assessed. The initial assumption is that the algorithm will always run after network initialization and when the first instance of measuring the propagation delay and choosing a slot size is over, the data plane operation can begin.

A. *Experimental demonstration*

For the data plane demonstration the same three nodes interconnected in a ring are used. As the correctness of the slot size decided by the algorithm has already been verified, the data plane performance is investigated only for the reference propagation delay of 30.625 µs and the chosen slot size of 245 ns. For connecting the nodes, three vacant cores are used from the 2-km long 7-core multicore fiber that is also used for synchronization. It is important to note that the propagation delay is continuously measured during the data plane operation in the three other cores in which the *sync_out* signal is sent. The three cores allocated for the data plane are used to both send data and distribute a trigger for synchronization.

Figure 7 gives an overview of the envisioned data plane connectivity. The detailed experimental setup is shown in Fig. 8. Initially, a single wavelength channel, 1549.32 nm is used to carry the slotted data. The *sync_out* signal and the trigger use the 1549.72 nm and 1550.12 nm wavelength channels, respectively. The trigger is generated at NN1 after the slot size has been decided and sent along the ring

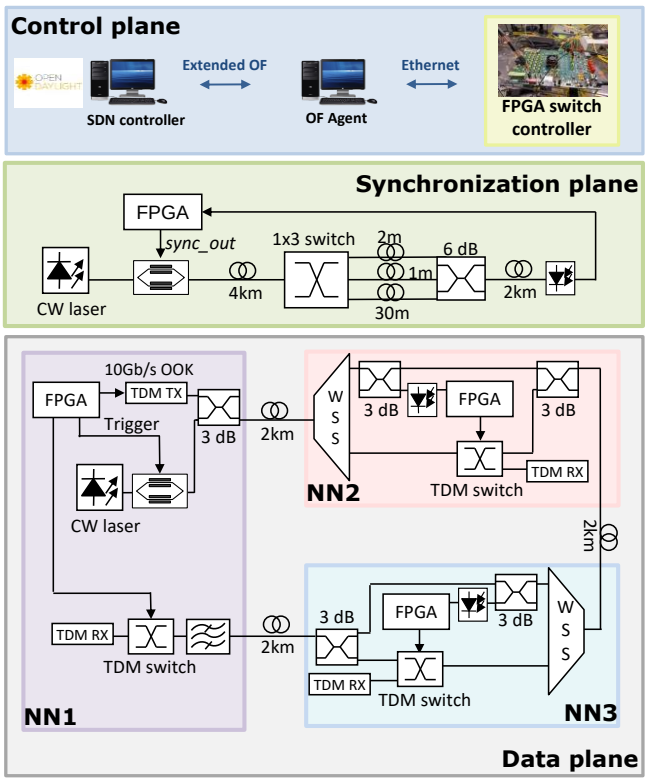

Fig. 8. Experimental setup

to provide synchronization to the other two nodes. At NN2 and NN3, the data and trigger are separated using a wavelength selective switch (WSS) and the trigger is passed to the FPGA after detection. Once transmissions are allowed, a continuous stream of 10 Gbit/s OOK modulated data bursts are generated and added to the ring at NN1.

The successful switching operation is assessed by dropping one burst at each node as shown in Fig. 7. We use three 1x2 $LiNbO_3$ switches [14] as TDM switches at each node. The logic for the configuration of the switches in each slot is provided by an extended Open Daylight SDN controller. The controller allows that a specific slot connection is established by setting the values for the input and output port and the time slot value. This information is sent to the SDN agent that in turn sends an Ethernet frame with predefined field arrangement to the FPGA. The FPGA is able to translate the commands into switch configurations, thus filling out the switching table for the different slots in each node. Both the synchronization algorithm and the control for the TDM switches are implemented using the same FPGA that communicates with the SDN agent. Based on the recovered trigger at NN2 and NN3, the slot information is retrieved and the control signals for the switches can be generated precisely. At NN1, the signal used to control the transmissions is applied to the switch for dropping the last burst in order to demonstrate that the ring is perfectly synchronized, i.e. the add and drop time coincide. Ideally, if *NxN* switches are deployed, then both the add and drop operations can be simultaneously performed at each node in the network.

In order to demonstrate the scalability of the scheme, we

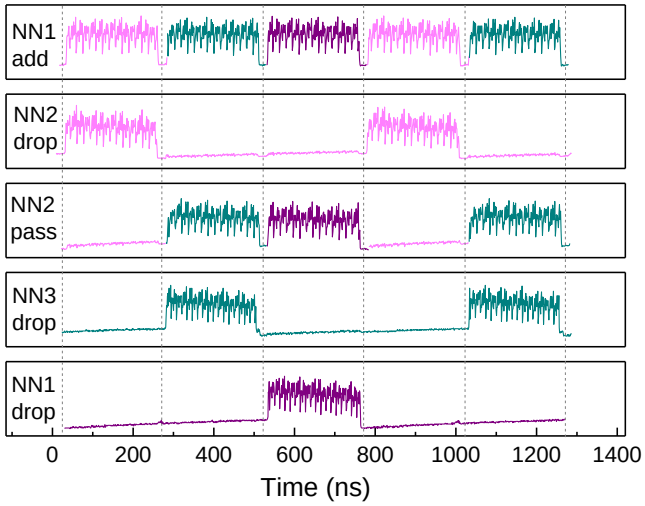

Fig. 9. Time domain traces of the generated and switched bursts at the different nodes

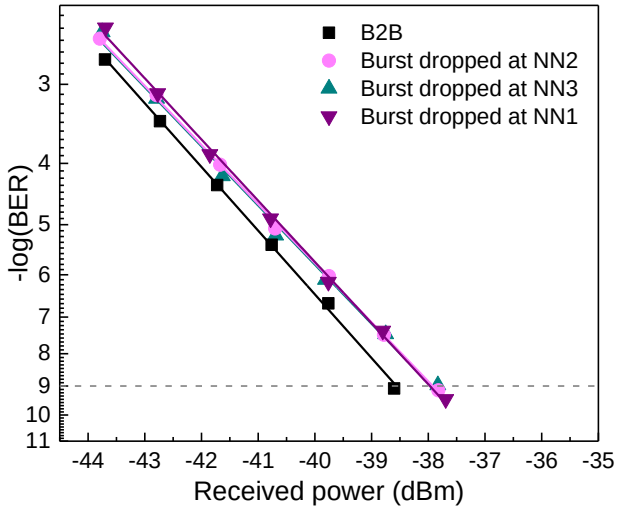

Fig. 10. BER performance of the bursts dropped at each node

increase the number of channels carrying slotted data to 15 WDM channels. The bursts are generated in each slot at NN1 and carry 10 Gbit/s OOK modulated data. The used slot size is 245 ns and the measured propagation delay is the reference propagation delay i.e. 30.625 µs. We use 15x50-GHz spaced channels within the range of 1547.72 nm and 1553.33 nm, inclusive. The trigger is distributed at 1553.73 nm and the *sync_out* signal at 1547.32 nm. Dispersion is compensated with dispersion compensating fiber (DCF) only at the receiver, thus no dispersion compensation is done for the purpose of switching.

B. *Results and Discussion*

The time domain traces of the bursts before and after switching are shown in Fig. 9. It can be seen that at NN1 bursts are generated in each slot with the correct slot duration. At NN2, one burst is dropped and the two remaining bursts are bypassed. At NN3, one burst is dropped and one burst is bypassed. Finally, at NN1 the default configuration is bypass, so the last remaining burst is dropped only if the switch is properly configured for that exact time slot. The time domain traces illustrate that with the used synchronization algorithm proper switching can be achieved at each node.

The BER results of the bursts dropped at each node are shown in Fig. 10 as function of the received power. It can be seen that each burst dropped at different nodes experiences only 1-dB power penalty and different connections in the ring have the same performance. This confirms not only the feasibility of the algorithm, but also the main motivation for this work, namely the ability to provide support for optical subwavelength switching in the data plane.

To confirm the WDM compatibility the focus is set on NN2 where one burst is dropped from each wavelength using the same control signal for the switch. The received spectra of each wavelength channel before the TDM switch and the measured receiver sensitivities (BER=10^{-9}) of the dropped bursts are shown in Fig. 11. All channels have similar performance (within 1.45-dB margin), thus verifying that the same control signal can be used to switch bursts on

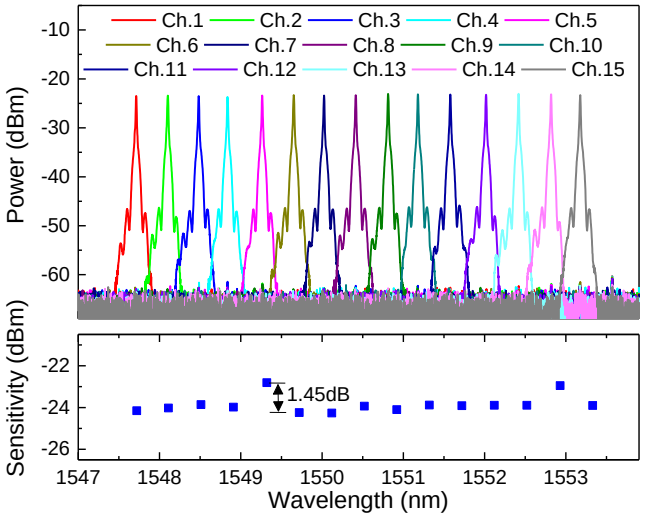

Fig. 11. Spectra and receiver sensitivity of all channels on a single burst dropped at NN2

different wavelengths without observing synchronization errors. Considering that no dispersion compensation technique is deployed prior to switching, the allocated switching gap is more than enough to compensate for any offset among the bursts carried on different wavelengths. This confirms that the proposed synchronization algorithm is both SDM and WDM compatible and scaling in space and wavelength is feasible with common synchronization for all bursts on all wavelength channels in all cores.

V. Conclusion

We discuss the main characteristics of different optical subwavelength technologies and present the concept of optical TDM switching. Based on the detailed synchronization requirements reviewed, we propose an algorithm that enables automatic synchronization of optical TDM switching nodes in a ring with random propagation length. Besides providing support for synchronization, the algorithm facilitates additional functionalities such as

flexibility by allowing SDN-controlled decision to be made regarding connection granularity as well as assisting protection schemes. We experimentally validate the algorithm behavior and achieve successful synchronization for three different ring lengths. Moreover, using the proposed algorithm to provide synchronization, we experimentally demonstrate data plane operation and successful optical TDM switching at three nodes connected in a ring network. At last, the SDM/WDM compatibility of the algorithm is successfully verified by switching data bursts carried on different wavelength channels using common synchronization and control signals for switching. Based on the proposed synchronization algorithm and its performance, we are confident that synchronization in a ring network can now easily be achieved in practice. We believe that this result can help pave the way towards commercially deployable optical subwavelength technologies.

ACKNOWLEDGMENT

This work was supported by ECFP7 grant no. 619572, COSIGN. We would like to thank OFS for providing the MCF with all-fiber fan-in/fan-out devices.

REFERENCES

[1] A. Singh *et al.*, "Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network," *Proc. SIGCOMM*, pp. 183-197, London, United Kingdom, 2015.

[2] V. Kamchevska *et al.*, "Experimental Demonstration of Multidimensional Switching Nodes for All-Optical Data Centre Networks," *Proc. ECOC*, Tu.1.2.2, Valencia, Spain, 2015.

[3] V. Kamchevska *et al.*, "Experimental Demonstration of Multidimensional Switching Nodes for All-Optical Data Center Networks," *J. Lightwave Technol.*, vol. 34, no. 8, pp. 1837-1843, 2016.

[4] V. Kamchevska *et al.*, "Synchronization Algorithm for SDN-controlled All-optical TDM switching in a Random Length Ring Network," *Proc. OFC*, Th3I.2, Anaheim, USA, 2016.

[5] A. Stavdas, A. Salis, A. Dupas, and D. Chiaroni "All-optical packet synchronizer for slotted core/metropolitan networks," *J. Opt. Commun. Netw*. vol. 7, no. 1, pp. 88-93, 2008.

[6] B. R. Rofoee *et al.*, "Demonstration of low latency Intra/Inter Data-Centre heterogeneous optical Sub-wavelength network using extended GMPLS-PCE control plane," *Opt. Express*, vol. 21, no. 5, pp. 5463-5474, 2013.

[7] M. Baldi *et al.*, "Scalable Fractional Lambda Switching: A Testbed," *J. Opt. Commun. Netw*. vol. 3, no. 5, pp. 447-457, 2011.

[8] N. Le Sauze, D. Chiaroni, O. Rofidal, A. Dupas, "New optical packet synchronizer for optical packet routers," *Proc. PS*, pp. 57-59, Monterey, USA, 2001.

[9] K. Hattori *et al.*, "Optical Layer-2 switch network based on WDM/TDM nano-sec wavelength switching," *Proc. ECOC*, We.3.D.5, Amsterdam, Netherlands, 2012.

[10] K. Hattori et al., "Method for synchronizing timeslot of WDM/TDM multi-ring network independent of fiber delay," *Proc. OECC/ACOFT*, pp. 227-229, Melbourne, Australia, 2014.

[11] A. Hartog, A. Conduit, and D. Payne, "Variation of pulse delay with stress and temperature in jacketed and unjacketed optical fibres," *Optical and Quantum Electronics*, vol. 11, no. 3, pp. 265–273, 1979.

[12] S. Yao, B. Mukherjee, S. Dixit, "Advances in Photonic Packet Switching: An Overview," *IEEE Commun. Mag.*, vol. 38, no. 2, pp. 84-94, 2000.

[13] *ASHRAE Thermal Guidelines for Data Processing Environments – Expanded Data Center Classes and Usage Guidance*, ASHRAE, 2011.

[14] EOSPACE, http://eospace.com/.