**Griffin: Programmable Optical DataCenter with SDN enabled Function Planning and Virtualisation**

Bijan Rahimzadeh Rofoee, George Zervas, Yan Yan, and Dimitra Simeonidou

Abstract—Optical networking technologies are very attractive for intra-DataCenter (DC) interconnectivity due to their intrinsic high bandwidth and low latency characteristics. As the bandwidth demand grows, quantity and size of IT and network infrastructure grows which puts pressure on power, cost, and space consumption. Therefore programmable optical networking solutions, which are highly controllable and provide granularity and scalability in services, can greatly enhance operations and resource efficiency for DC interconnects. This paper proposes and reports on a novel function-programmable and application-aware intra-DC technology. The hardware platform uses FPGA implementation of Top of Rack (ToR) switch which supports two main software-defined network functions of a) Ethernet over WDM and b) Ethernet over TDM sub-WDM. Each switch can be programmed to virtually operate as an optical Ethernet or sub-wavelength transport system. The TDM function further supports a range of additional programmable features in Time-Slice and frames. The system can be controlled to switch between the functions on-demand and on a hitless manner. This platform with unique flexibility in functions and features, is configured to operate with various processing times, throughput, and delays, enables on-demand Quality of Service (QoS) definition, to match diverse cloud application requirements. Optical fast switches enable all-optical transport between the ToRs, further supporting low-latency and resource efficiency. A multi-stage software-defined controller is built which at the first stage uses application requirements to virtually create function programmable sub-networks by repurposing the data plane functions as well as control plane (algorithms) features. In the second stage it allows dynamic allocation of resources of the sub-network for the duration of the application. This platform is characterised to operate with different QoS metrics and evaluated for DC use cases.

Index Terms—Network function programmability and virtualization, Software defined optical networks, sub-wavelength switching

1. INTRODUCTION

Datacenter traffic analysts predict the number of servers in Datacenter (DC) will increase tenfold by 2020 [1] while almost 76 percent of all the traffic of DC Networks (DCN) will be within DCs which is labeled as east-west traffic [2] [3]. Traditional DCN with campus-network model of multi-tier electrical switching in tree architectures introduces major challenges to DCN efficiency as well as scalability. They suffer from a) high inefficiency in Electrical-Optical-Electrical signal conversion b) Multiple electrical switching topologies are unsuitable in directing inter-cluster (set of racks) traffic in east-west direction, c) commodity and non-optimized technology deployment d) lack of application awareness with various QoS requirements and workloads (storage and VM management, interactive applications) etc. Such inefficiencies result in major difficulties for conventional DCNs to scale-up easily and match with the rapid pace of traffic growth [4]. This has led research and industry to consider optical networking technologies inside DCs to take advantage of their abundant capacity, small footprint and low power-consumption. Many of the technologies and architectures have been proposed to complement the switching/routing architecture of the traditional DCs, whilst some other works suggest all-optical networking as a substitute for the existing DCN. In this trend the proposed optical DCN solutions are generally categorized into hybrid (electrical and optical) and all-optical approaches [5]. The hybrid approaches aim at supplementing the current architectures with affordable, off-the-shelf optical components to introduce high bandwidth links next to the existing packet networks whilst keeping the solutions economically sound. All-optical DCN architectures on the other hand deploy variations of optical circuit and packet switching technologies, and use components and subsystems such as Micro Electro Mechanical Switches (MEMS), Semiconductor Optical Amplifiers (SOA), Wavelength Converters (WC), etc [6] to build highly transparent architectures within DCs.

In the meantime and thanks to powerful software platforms, Software Defined Networking (SDN) architectures have emerged promising of great advantages for DCN [7] [8]. The benefits of adopting SDN controllers (with OpenFlow being the flagship protocol [9] [10]) stem from separation of control and data-planes and bringing the control functions under a central and global framework. SDN facilitates intelligent network control by exposing open API for network programmability. It greatly simplifies IT and network integration and virtualization, and facilitates traffic engineering, load balancing, and applying network policies and configuration [11]. Apart from network logics and control, more recently paradigms such as Network Function Virtualisation (NFV) [12] have emerged to extract and virtually deploy hardware functions in software environment. SDN and NFV are greatly complementary in functions and operations.

In this paper, we demonstrate the implementation and
application of a novel highly-programmable intra-DCN technology and architecture, enhanced by a multi-stage SDN controller, which by exploiting re-programmability of wide range of protocol functions allows virtually creating technology-specific sub-networks for various requests. The data-plane uses Field Programmable Gateway Array (FPGA) based Ethernet to optical sub-wavelength transport platform operating as Top-of-Rack (TOR) module, which is interconnected to other TORs via fast optical switches. This platform is capable of switching between two protocols of Ethernet over WDM (EoWDM) and Ethernet over sub-WDM (EoSWDM) in a hitless (no loss) manner to gain different bandwidth granularities and to change between many-to-many aggregation mode and high throughput point-to-point with no disruption to running applications. In a novel approach, the TDM sub-wavelength features are made flexible and programmable which allows repurposing the TDM function to gain different levels of QoS. These features are: a) variable number of time-slice in each frame, and b) time-slices with variable payload, overhead, which are all implemented as programmable functions rather than fixed and hardwired components. This enables the FPGA TOR to virtually act as a EoWDM system, or EoSWDM on-demand, depending on the required mode of operation.

The TDM sub-wavelength switching enhances resource efficiency by aggregation of multiple flows into few wavelengths. Another advantage of adopting time-sliced sub-wavelength switching technologies for DC are the capability of these technologies in handling bursty traffic patterns [13] which are well dominant in the edge and aggregation tiers of DCNs [14]. Moreover optical fast switching of the bursts and/or circuits between the FGPA limits the electronic processing to the TOR points in the architecture. We call this highly programmable platform with hybrid functions Griffin for ease of referencing.

Latency and bandwidth requirements for DC applications generally are not a set and fixed value, however HPC applications are generally considered among those with tightest throughput and latency requirements (micro seconds to milliseconds) [15]. Capital markets and financial services are also among those which require very good response times (few milliseconds), as the faster response translates to higher revenues. EoWDM is a better fit to meet these requirements. Moving to application such as web based distributed computing the latency requirements drop to 100s of microseconds to seconds. For these applications EoSWDM can be a more suitable option as it meets the latency requirements of such application whilst introducing the advantages of optical transport within the DC. It should be considered however the networking latency usually referred to is the application-to-application figure which is inclusive of software and hardware components in different layers of the network. This however is out of the context of this work and we are using the rough numbers to categorise Griffin technologies for meeting various classes of traffic. Big data processing applications such as Hadoop which enable distributed processing of large sets of data across thousands of machines have strict networking requirements. These applications generate various flow types, mainly known as elephant and mice flows during their process, with the former being long lived and latency tolerant whilst the latter are short lived and latency sensitive [16]. For such applications Griffin multi functionality can be highly advantages where EoSWDM mode can serve elephant flows, whilst EoWDM can be used for mice flows.

It should be noted that, the extended programmability of Griffin in switching between EoWDM and EoSWDM, with finely tuneable EoSWDM features, enables this platform to be customised for both latency and bandwidth to meet wide ranges of application requirements in compute, storage, database, and so on.

All the programmable features in the hardware are exposed to a multi-stage software defined networking controller (MS-SDN). MS-SDN controller follows the centralised SDN architecture with multi-layered components, controller, and applications, and extends it to support software definition of hardware features as well. The MS-SDN is built to carry out planning and operation in two cyclic and sequential stages (Fig. 1): in function planning and programming a sub-network of Griffin nodes are created to provide inter-rack communications following the job/application requirements. This stage also includes instantiation of the relevant algorithms for dynamic provisioning on the sub-network over its course of operation. 2: The Operation phase which routing and resource allocation algorithm is used to dynamically add or drop connection flows over the established sub-network. The proposed solution operates with the enhanced MS-SDN controller, and supports Network Function Virtualisation concept and encompasses its objectives of dynamically recreating and adopting network services, as it allows virtually creation of sub-networks running different protocols for different transport requests.

In the following we are providing an extensive review of the various architecture and technology trends for optical DCNs. In section III the proposed network with programmable functions is introduced and demonstrated. In section IV the FPGA design for programmable data plane is provided. Its function blocks and the implemented mechanism to realise a programmable platform are explained. In section V the MS-SDN controller frame work is introduced, its structure and modules are explained, and the workflow in different stages of operations is described. Section VI provides the characterisation process of Griffin in which the impact of various TDM features programmed by the controller is quantified on real-time traffic. Section VII introduces the algorithmic applications developed which use the characterisation results and provide the TDM features needed to realise the system with expected latency and throughput.
Section VIII provides the testbed setup and the evaluations, and at last section IX concludes the paper.

II. LITERATURE REVIEW:

Optical DCN technologies: Programmable switching technologies have been introduced in form of soft switches such as Open vSwitch [17] and indigo IVSwitch [18] to provide open source implementation of L2 switching for virtualized and actual deployments. Also hardware switching platforms called white boxes have been built exploiting the OVS or Indigo references in products such as Pica8 [19] products to convert the software based programmable switches into high performance hardware implementations. On the other hand and the giants of telecom have also introduced their programmable switching technologies such as HP, Brocade, NEC, and so which feature high port density and low latency transport utilizing specialized hardware, with SDN capability designed for DC applications.

Optical networking solutions for intra-DataCenter (DC) interconnectivity have gained considerable attention due to their intrinsic high bandwidth and low latency characteristics. Numerous optical DC interconnects have been proposed recently and can generally be categorized in to all-optical or hybrid solutions [5]. Most notable hybrid DCNs have been introduced in c-through [20] and Helios [21]. These solutions aim at improving throughput and cost effectiveness for cloud computing by combining electrical packet with optical circuit switching and applying them in DCN architecture. They propose use of optical switching of fiber, wavelengths, where they are used for delay sensitive/bandwidth-hungry applications. These proposals have been adopted by many as a reference for hybrid DCN design. However their scalability have been questioned, considering coarse optical channels (c-through mostly with fibre channels), static architectures and also issues such as slow provisioning times of traditional telecom technologies for highly dynamic datacom application.

The work in [22] explains possible difficulties of these technologies in serving multi-tenant and heterogeneous modern DC environments.

Apart from hybrid opto-electronic DCN solutions, all-optical DC networks have also been addressed and studied. Proteus [23] is an all-optical switching solution for interconnecting electronic TOR switches within DC which proposes use of MEMS optical switches along with Wavelength Selective Switch (WSS) filters to direct the traffic between different racks. This solution provides the flexibility of using different number of wavelengths for communications through reconfiguration of a central MEMS switch. However it still can impose scalability issues due to limited wavelength channels. Besides network reconfiguration latency of MEMS can be a limiting factor in responding to dynamic traffic patterns. From architecture point of view the connectivity between racks depends on whether the racks are directly connected through the MEMS or need to traverse through other hops to communicate which makes the in direct paths non-deterministic. This is caused by limited direct connectivity each TOR has to other hops through the MEMS switch. In a previous work [24] we have introduced an intra-DCN architecture which uses Network Interface Cards with programmability features on the FPGA chip and communicate over programmable optical switches so to dynamically create wavelength and sub-wavelength supporting networks between servers depending on the networking requirements.

A number of all-optical DCN solutions are based on packet circuit switching technologies. DOS architecture in [25] proposes all-optical DCN using Arrayed Waveguide Gratings (AWG) for interconnecting TORs. DOS is based on extracting labels from packets, which are processed by a controller before the AWG router node. The controller using the label decides which wavelength the packet should be transmitted on and uses tunable wavelength converters (TWC) for this purpose. Contention avoidance is also considered, by adopting an electrical buffer connected to the AWGR, which the contending traffic is looped back on to it. The solution provides flexible bandwidth however the higher costs of TWCs besides the proposed electronic buffering within the optical architecture can become a scalability bottleneck.

Authors of [27] propose a central all-optical peta-switch architecture. Its structure is similar to DOS in that it uses TWC and AWG components for traffic routing. AWGs are connected in three-stage close non-blocking architecture. Each TOR uses tunable lasers with electrical buffering of the packets prior to transmission to avoid contention. This solution promises greater flexibility compared to DOS architecture mainly due to removal of electronic buffer from optical switch architecture. However the large number of TWCs and also a single large scale switch as proposed can be a point of failure and also a scalability issue. Osmosis [29] is another all-optical solution, which uses couplers and splitters to allow the WDM transmitter from each TOR to send and receive traffic following the scheduled transmission turns. It uses fast SOAs to route the traffic between the two ends. The main disadvantage of this solution is power inefficiency by excessive use of SOAs in the architecture to make switching matrices. E-rapid proposed in [30] can be used as an intra DCN solution [5], which uses short range VCSEL transmitters on different wavelength per each node (TOR) to achieve optical parallelism through spatial division multiplexing of traffic flows. Each node then sends the data over a super high-way optical path which consists of optical rings with add/drops per hop. The proposed solution is energy efficient as well as providing fast end-to-end communications. This solution on the downside can be limited on traffic engineering and load balancing due to ring connectivity. Another all optical solution for intra DCN is IRIS [31]. It uses three stage non-blocking AWGR switches, with TOR nodes transmitting over a number of WDM transceivers. First stage is wavelength switching using TWC, second stage uses time switches enhanced with wavelength converters to form a big optical buffer, and the third stage uses a space switch to direct the traffic to various destinations. This solution is non-blocking using scheduling algorithms however it extensively deploys WCs. Data vortex proposed in [32] also uses SOA based interconnectivity, in which nodes are connected in vortex of ring light paths with some cross ring links. The use of SOAs
provides fast reconfiguration; however the topology can causes traffic management complexities as scales when traffic needs to traverse over increasing number of hops [5]. Another architecture using SOAs, proposes bidirectional switching matrices by cascading 2x2 fast switches in [33]. These switches are connected in a tree topology however scalability is restrained by operationally expensive SOAs.

Optical Orthogonal Frequency Division Multiplexing (OOFDM) technologies have also been introduced into DCNs [34]. The work in [35] performs feasibility studies of deploying such technologies in DCN. OFDM based architectures offer high spectral efficiency and fine-grained bandwidth allocation, with the main drawback of scalability and high cost due to complex DSP and ADC/DAC systems.

SDN solutions for Optical networks: Optical Transport Working Group (OTWG) has been initiated within the Open Network Foundation (ONF) with the focus on SDN technologies for optical networks [36]. This activity is resulted from the benefits of adopting SDN technologies in several attempts for optical transport-network control and management as well as SDN based virtualization services [37]. OpenFlow based SDN approaches have a prominent position in this effort. Authors in [9] propose OpenFlow based architecture in optical networks and discuss required extensions to support optical transport along with packet networking. The work in [38] demonstrates adopting OpenFlow SDN to control a programmable optical network platform with variable bandwidth over multi-core links. The controller is equipped with network service applications purposely built to manage network resources for infrastructure slicing and isolation and dynamic bandwidth assignment. A proof-of-concept demonstration of OpenFlow based wavelength path control in transparent optical networks is presented in [39]. In this work, virtual Ethernet interfaces are mapped to physical interfaces of an optical node (e.g. photonic cross-connect - PXC), and enable an SDN controller (NOX [40]) to operate the optical light-paths via the OpenFlow protocol. In a previous work of ours, we have reported on a novel intra and inter DC experimental testbed which uses two heterogeneous sub-wavelength technologies of OPST by Intune networks [41] and TSON [42] enhanced with extended Generalized Multi-Protocol Label-Switched (GMPLS) technologies for sub-wavelength switching. Authors of [43] introduce Software Defined Optical Network (SDON) architecture with QoS-awareness and with control protocol for optical burst switching (OBS). The performance of the proposed protocol was evaluated with the conventional (GMPLS) distributed control and the results show SDON improves resources efficiency and capacity utilization compared with distributed GMPLS. OpenFlow controller has also been extended to support multi-layer networking from L3 and L2 down to the optical layer. Authors of [44] use OpenFlow for unified multi-layer control of IP/Ethernet and TDM switched packet and circuit switching networks. Work in [45] is another multilayer control demonstration across multiple layers which uses OpenFlow controller to establish network paths with different granularities and captured latencies for end-to-end path creation and restoration. A multi-domain, multi-technology of packet, fixed & flexible DWDM Grid Technologies over an OpenFlow SDN based unified control plane has also been reported in [46]. Central network management using web-services based provisioning and control with the benefit of improved efficiency in managing network resources and configuring devices has been studied in a number works as well [47] [48]. In a recent work [49] we introduced a multi-rate system which combines super-channels and sub-lambda channels and exploits wide range of allocation algorithms through web-service APIs.

III. SYSTEM AND NETWORK OVERVIEW

The proposed platform implements programmable function provisioning on the chip to adopt EoWDM and EoSWDM modes of operation to offer flexible and application-aware services. EoWDM carries traffic over DWDM wavelengths, it however can be too coarse allocation of wavelength data rate for fine-granular (less than Wavelength) Ethernet flows. EoSWDM on the other hand provides resource efficiency by aggregating (TDM) granular traffics in bursts of data and switch them all optically. This extra flexibility in TDM function enables the user to tune the system to compose systems with different QoS capabilities by setting frame and TS values. However to understand how the micro changes on TDM features of EoSWDM, its structure better be explained.

![Diagram](image)

Fig. 2. (a) A connection, frame and time-slices in hierarchical order (b) the effect of programmable features
recovery (CDR) at the receiver and also accommodates a switching time when the burst is routed over the network.

In Fig. 2.b, a Venn-diagram out of Griffin features is provided to illustrate Griffin programmable functions and their impact on EoSWDM operations in terms of latency and throughput. TS payload size is the number of data bits in each TS where Ethernet traffic is encapsulated. The overhead bits on the other hand provide safety gap for error free switching of the TS. Evidently a burst is made of payload and overhead bits, and the relative sizes of payload and overhead determines the TS throughput, which is programmable and can be adjusted. The frame size is basically the number of separate traffic flows in TSs to be aggregated in a frame over a Wavelength which defines the capability of the system in aggregating multiple flows.

The intersecting sections between the sets show how different programmable features can affect the output performance. Combination of frame-size and TS payload provides total payload bits carried in frame iteration. The common area between Frame-size and TS indicate the frame level overhead, and the zone between TS payload-size and the TS payload-overhead define the TS throughput. In the central intersection the available bandwidth and latency per TS can be derived exploiting the TS and frame level attributes. We have used these co-relations between the programmable features to characterise the FPGA behaviour and provision TDM systems for different requirements (section VII). The functional view of the Griffin FPGA node is shown in Fig. 3.a. The Griffin node receives Ethernet traffic on the access (Rack) side, and based on the planning information decides whether to operate as EoSWDM ToR or EoWDM ToR. The EoSWDM passes through an aggregation process where data bursts are multiplexed whilst EoWDM bypasses this process. Traffic is then sent for optical transport via transceivers.

The network architecture is displayed in Fig. 3.b in which DCN has three tiers of racks, aggregation and core. Griffin nodes are placed as TOR switches on different clusters with fast switches (green hexagons) as optical interconnects in a folded-clos aggregation layer. MS-SDN on the other hand has the global vision of the network elements and programs and operates the nodes. The suggested topology facilitates scalability as well as multipath communication between the clusters in east-west direction [50]. In addition and exploiting the multi-protocol features of Griffin, ToR-to-ToR direct communication between the Griffin is considered as well (case 4). This enables high-throughput and low-latency intra-cluster communications and skips switching to get to the next rack. There are sample traffic flows drawn between the racks, which aim to illustrate how the programmable feature of Griffin are used in this architecture. Fig. 3.c also visualises the frame and TS structure in each flow using the associated label with each of them. Frame labelled (1) shows intra-cluster EoSWDM via the optical switch where 4 different traffic flows (F1, F2, etc) are using the same Wavelength. Frame (2) shows the same length frame as in (1) with same number of flows aggregated however with different proportions of payload and overhead. The objective in (2) is to keep the delay (defined by frame length) as it was in (1) whilst accommodating extra overhead for longer switched paths as more switches require more overhead. It is evident the throughput is compromised in exchange to meet the latency requirements. Frame (3) is used for data transfer among clusters but going through the same number of switches as in (2), and shows how the frame is being modified to accommodate only 2 flows to keep the frame throughput the same as it is in (1). This means the level of aggregation (number of time-slices) is compromised. From

![Diagram](image.jpg)

Fig. 3 (a) The high level view of different data paths inside the Griffin nodes. (b) Topological view of Griffin network. Inter cluster interconnectivity using clos and fold, and mesh intra cluster communication (c) Visualization of flexible time slice and functions using the EoSWDM mode.
another perspective if case (2) had only two flows of F1 and F2, cases (3) and (2) would have provided the same level of aggregation but (2) would have experienced half of the latency in aggregation. The EoWDM is also displayed using identifier (4) where it can establish ToR-to-ToR connection. These cases demonstrate how the programmability in Griffin functions makes it highly adaptive to the application requirements and also for different networking conditions (intra/inter-cluster paths). EoWDM high bandwidth and low latency transport vs more resource efficient EoSWDM offer a coarse selection of two service types. This means there can be traffic types, which in terms of requirements sit somewhere in between of the two services. Griffin tries to address this issue by enabling programmable EoSWDM transport so the latency and bandwidth can be adjusted to achieve various latency and bandwidth figures.

On the controller side and as shown in Fig. 3.b, a multi-stage SDN controller is shown (identified by MS-SDN), which is communicating with network-service consumers (northbound REST API) in Function Planner stage for creating sub networks, and in operation stage for running the network. Function Planner uses algorithms to calculate the TDM features using latency and bandwidth input information in addition to characterisation data. During the operation stage, the controller uses routing and allocation algorithms so the participating nodes can dynamically add and remove connections. The outcomes of either stage are processed and passed to the FGPAs via software agents.

IV. Griffin FPGA Architecture

Fig 4, displays the FPGA (HTG-V6HXT-100G) function blocks used for Griffin control, ingress and egress paths [51]. The data-flow follows the direction of the arrows. Fig 4.a shows the control Look-Up-Table LUT and its connection to the node agent on the north side. The agent feeds the control information to the FPGA via a 10GE link. The LUT holds the information for network functions and their operation which are used by data path blocks (EoSWDM/EoWDM) as shown on the figure. The dotted lines from LUTs go to the blocks which are in control of the planning stage (coloured orange): EoSWDM/EoWDM switchover to change the mode of the switch. On the other hand there are straight lines going to the blocks for node operation (Green): Traffic buffer and isolation, fast switch control. The red box uses the information for TDM features as well as burst aggregation and transmission, which is controlled in both stages. Fig 4.b displays the ingress traffic flowing from rack into the FPGA. Ethernet frames go through the transceiver as well as the header parsing process to be identified and classified. Afterwards they are routed onto either EoSWDM or EoWDM paths based on control information. In EoWDM path Ethernet frames are recreated and sent out on the right output interface. On the EoSWDM path the frames are queued and aggregated, and based on the information on the scheduler they are emitted on their allocated TS onto the right output. Fig 4.c shows the egress path for Ethernet frames received at a Griffin end point. EoSWDM and EoWDM signals are separated after reception, and EoSWDM bursts are sent to the segmentation block so to extract the Ethernet frames and later send them out. For EoWDM the Ethernet frames are received, parsed and sent out without extra process. EoWDM and EoSWDM switchover has been discussed in a prior work [24] and also in [51] where more detailed explanation of EoSWDM programmable sub-function implementations and results have been reported. In summary the mechanism takes place as in following:

Switching from EoSWDM to EoWDM, upon being triggered by the controller, the FPGA would still carry on processing the group of Ethernet frames according to the allocation info, which essentially means delaying the EoWDM for up to 8x1500 Byte (~50 µs) frames stored in the aggregation buffer so to avoid losing any data packets. On the other hand and the change from EoWDM to EoSWDM modes, the FPGA platform of Griffin can switchover after one Ethernet frame which is 1500B before moving to EoSWDM datapath, where the subsequent Ethernet frames are buffered and aggregated. The FPGA operating as EoWDM achieves 1.7μs back to back latency with 9.18 Gb/s per single wavelength channel. The granularity of the traffic bandwidth per time slice which can be configured on Griffin is 6.8Mb/s up to 8.8 Gb/s (considering no switching overhead but inter burst K characters) working with the time slice duration, and the minimum latency can be achieved having single time slice in the EoSWDM mode is 13.1µs up to 118.2µs.
V. MULTI-STAGE SDN CONTROLLER

The multi-stage software defined controller (MS-SDN) is built for this programmable system to consolidate planning and operation in single framework and allow applications define function for technology-virtualised sub-networks as well as performing allocation and provisioning. The MS-SDN functions in a proactive way[36], which is similar to many other optical based SDN controllers, where applications/users tune the system parameters ahead of the traffic. So every traffic stream or flow needs to be identified and then the Griffin network is scheduled to host the traffic, in hardware and software. It uses a central backplane, which communicates with all other modules and coordinates them in different stages (Function Planning and Operation) Fig. 5. Each of the modules are developed and placed in a different VM and communicate through REST API. (The VMs are marked on the figure) Each module including applications and algorithms expose their functions as resource Unique-Resource-Identifier (URI) and are registered at the backplane VM. The modularity of the controller facilitates extensibility and scalability. MS-SDN components are explained in the following.

Controller-Backplane: MS-SDN uses a software backplane (VM 101) which hosts a set of essential function blocks for the network control and consists of Network Topology Manager, Function-Planner logics, Network-Operations logics, and Dataplane-Technology-Mapping. Network Function Planner and Network Operations logics are used to communicate with the corresponding application, depending on the request type. Network topology is built using the Griffin nodes information and adjacencies at an initial phase when a set of Griffin nodes are connected to the controller. It holds a list of the nodes, their status, and the sub-networks created. Dataplane-Technology-Mapping is used to map the abstract allocation information to the available resources in the testbed.

Network-Services: Network-Service hosts applications and algorithms and use the northbound REST-API to communicate with the controller. Network Request is users interface requesting for a sub-network with specific bandwidth and latency requirements in the first stage (Planning), and also to dynamically establish and control light paths between the nodes over the sub-network after that (Operation). Function Planner is the algorithm built for planning the EoSWDM and EoWDM functions on the selected Griffin nodes that best meets the aggregation, delay and bandwidth requirements. Routing and Resource allocation is an algorithm repository of shortest path routing, Routing and Wavelength Assignment (RWA), Routing wavelength and Time Assignment (RWTA) which are instantiated in planning and invoked in operation by the central controller.

Node-Agent: Each Node-Agent is implemented as a separate VM and is associated with each Griffin node; it holds node information and parses the messages between the FPGA and the controller. A REST-API as southbound protocol is used to communicate with the agent from the controller. The agent speaks to the FPGA via a 10GE SFP+ interface and passes information for node function repurposing and also for nodes operation including matching, transmission, and switching.

WorkFlow: The complete control workflow is shown in Fig. 4. b.: (1) A request arrives for a sub-network with specific requirements for a specified duration. (2) Set of shortest paths interconnecting the selected nodes are computed to create the sub-network of the selected nodes, (3) the application

Fig. 5. (a) Modular view of the SDN control and infrastructure management (b) Communication workflow in the controller
requirements of number of aggregation flows, bandwidth and latency are fed to the Function Planner algorithm to compute the functions specs for the selected nodes. After planning, the features of the selected nodes are passed to the central controller and then to the node, which the controller then (4) instantiates an appropriate resource allocation algorithm (RWA, for EoW, RWTA for EoS) considering the network specs at the first stage, and then (5) uses it for serving provisioning path requests during the operation phase. Fig. 4. c and d on the other hand show the breakdown of the function blocks which are invoked in steps and serving for different stages.

VI. GRIFFIN CHARACTERIZATION:

Griffin Function Planner (VM103 on Fig. 4.a) is a decision making entity, and aims at tuning the nodes behaviour, so different applications communicate using virtually EoWDM or EoSWDM networking elements with customised TDM parameters. To enable this we need to understand the Griffin node behaviour upon making changes to its function programmable features from the application perspective (end to end delay, bandwidth, etc). In the EoWDM mode, the data path has a fixed number of buffering points, and therefore the latency and bandwidth would not vary considerably if the system is not overloaded. The back to back experimental measurement of EoWDM over 1 meter of fibre are 9.18 Gb/s of throughput with 1.75μs port to port matching and forwarding latency [51]. In the EoSWDM however due to the variable sized aggregation mechanism caused by programmable TDM, the relation between the payload size and delay is not straight forward to measure. As the aggregation becomes larger (more time slices, larger frames) the latency incurred in aggregation becomes the dominant latency cause. Therefore considering the co-relation between different TDM features (Fig. 2), we characterised Griffin by programming for different setups and observing the impact on the real-time application, so we can derive models which help us to estimate on TDM parameters using application latency and bandwidth requirements.

Methodology: The bandwidth provided by each TS can be calculated from the payload duration of each TS over the course of a frame excluding the overheads. To calculate the delays experienced by each TS we captured the difference in delay between two consecutive allocation patterns in a range of network loads so to get the net delay for a single TS (Ethernet frames in the TS) staying in the aggregation buffer. Measurements for a set of allocation patterns have been made as shown in Fig. 6a. It shows the end-to-end latency for traffic rates of 8.7, 4.3, and 2.9 Gb/s with each TS carrying 128Kb data and 8704 overhead bits. The allocation patterns show frames made of TSs which are represented by either "1" or "0" (where "1" represents allocated TS whilst "0" shows not allocated TS. The total number of "1" and "0" is the frame length). For example, we measure the latency of the traffic in Alloc2 with half of the available bandwidth which is roughly 4.3Gb/s. So by keeping the same rate, and then changing the allocations to Alloc3, the difference between the two allocation delays gives us the latency at 4.3 Gb/s for the individual TS. From Fig. 6a and the latency bars for each rate it is evident the difference between neighbouring allocations increases linearly. When subtracting the allocations to get the increase in latency by each TS, we are excluding the delays imposed on traffic on other stages in the datapath, as apart from aggregation, other stages are constant in size. These delays however can be considered negligible when the frame lengths are large enough so they produce the predominant delays (e.g. a frame with 1ms length, the frames who experienced few micro seconds in EoWDM, would experience 100s of micro seconds in EoSWDM). Besides the fibre

---

![128Kb TS latency-rate benchmarked data](image1)

![128Kb TS latency calculated vs experimented](image2)

![TS latency-size relation chart](image3)

---

Fig. 6(a) Data collected for shifting allocation pattern of 128kb of time-slice size. (b) The charted and extended lines for time-slice duration. (c) The TS-size-duration charts for different rates, and the estimated
latency is also not considered which roughly takes about 5 μs per kilometre which is quite a long distance for intra-DC environment. The same process and results are collected for 192 and 256 Kb TS payload sizes with the same amount of overhead.

Demonstrated in Fig. 6.b the lines of experimentally measured delays (8.7, 4.3, and 2.9 Gb/s) against theoretically calculated (using previous process of extracting latency per each TS) values over different allocation patterns and different rates. To theoretically create the latency lines, the following formulation is used which is basically the frame delay calculated from single TS durations in each frame (Table 1).

\[
\text{Framedelay} = \text{Initial}_\text{value} + (\text{TS}_N - 2) \times \text{TS}_D
\]

The formulation is more accurate for frames longer than 2 TS, since with \(\text{TS}_N < 2\) the delay was not linear so the value was inserted as a constant identified and displayed as the \(\text{Initial}_\text{value}\). The rates of 1.9 and 6.5 Gb/s where extrapolated following the trends of the measured data in the cited rates. The overlaps of the theoretical lines with the experimental ones demonstrate the close estimate of the TS duration on the experimentally measured lines, which increases linearly. Finally, as displayed in Fig. 6.c, the graphs of the calculated TS durations versus the TS sizes of 128, 192, and 256 Kb are sketched. These charts (to be called “TS size-delay” chart) are used in the Function Planner when repurposing the Griffin nodes in response to application requirements. To simplify our charts for ease of calculations later on, we have used linear regression of the data, which has led to the equations shown in Fig. 6.c.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{TS}_R)</td>
<td>Time-Slice Rate (bit/s)</td>
</tr>
<tr>
<td>(\text{TS}_S)</td>
<td>Time-Slice Size (bit)</td>
</tr>
<tr>
<td>(\text{TS}_D)</td>
<td>Time-Slice Duration (sec)</td>
</tr>
<tr>
<td>(\text{TS}_{ps})</td>
<td>Time-Slice Payload size(bit)</td>
</tr>
<tr>
<td>(\text{TS}_{dp})</td>
<td>Time-Slice Payload duration</td>
</tr>
<tr>
<td>(\text{TS}_{HD})</td>
<td>Time-Slice Over-Head Duration (sec)</td>
</tr>
<tr>
<td>(\text{TS}_{OS})</td>
<td>Time-Slice Over-Head Size (bit)</td>
</tr>
<tr>
<td>(\text{BW}_{req})</td>
<td>Requested Bandwidth for TS (bit/s)</td>
</tr>
<tr>
<td>(\text{DEL}_{req})</td>
<td>Requested Delay for each TS (sec)</td>
</tr>
<tr>
<td>(\text{LR})</td>
<td>Line rate (bit/s)</td>
</tr>
<tr>
<td>(\text{TS}_N)</td>
<td>Number of Time-slices</td>
</tr>
</tbody>
</table>

As a hardware constraint for buffering bursts in the design a TS cannot be larger than 480Kb in size otherwise it would cause buffer overflow, which means the sum of the TS data and the overhead is limited to that. To switch TSs using the optical switching components in the architecture, each TS need to have a minimum amount of overhead to allow error-free switching, and also to enable correct reception at the destination. Table 2 shows the values measured by characterizing the effect of multiple stages of switching on the signal reception to achieve error-free communications. It can be seen FPGA-to-FPGA communications without switching modules requires 8709 bits of overhead solely for receiver synchronisation. By adding the switching modules, the required overhead increases dramatically since the signal goes on and off to alternate between the light paths and this greatly affects receiver CDR and requires longer time to recover the signal clock. The characterisation results are displayed in Table 2. It’s worth mentioning employing burst-mode receivers will enable faster resynchronization which leads to higher throughput however it is not the focus of this work.

<table>
<thead>
<tr>
<th></th>
<th>0 Hop</th>
<th>1 Hop</th>
<th>2 Hop</th>
<th>3 Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data bits</td>
<td>471296</td>
<td>320000</td>
<td>288000</td>
<td>256000</td>
</tr>
<tr>
<td>Overhead bits</td>
<td>8709</td>
<td>160000</td>
<td>192000</td>
<td>224000</td>
</tr>
</tbody>
</table>

VII. APPLICATIONS: GRIFFIN FUNCTION PLANNER AND RESOURCE ALLOCATION

By now, we have built the bridge between the technology and application requirements by associating the impact of the programmable features on the network performance. A number of algorithms are designed for the Function Planner to respond to four types of planning requests. Table 1 provides the acronym descriptions.

Algorithm I. Input: \(\text{TS}_N\) and bandwidth/output; \(\text{TS}_{ps}\) and \(\text{TS}_{OS}\): Aggregation of n traffic flows, where bandwidth is the main concern so the system needs to be tuned for the requested bandwidth and the best achieved latency possible. This process and function set serves applications where the throughput between a set of nodes is of major importance. Knowing the TS overhead already from the maximum number of hops, the algorithm can only shorten the TS payload to a level that provides the bandwidth requests for n flows. This leads to shorter TS and therefore to a shorter frame which means the best possible latency. The following equations define the TS payload size, which is derived from the aggregated bandwidth requested out of the available single channel rate, knowing the overhead. Table 1 helps with the acronyms

\[
\frac{\text{TS}_{ps}}{\text{TS}_{ps} + \text{TS}_{OS}} = \frac{\text{TS}_N \times \text{BW}_{req}}{\text{LR}}
\]
Theoretical bandwidth. This is the case where not much

\[ L = 1.00E+09 \]

output nodes “s” and

\[ TS_n = 6 \]

\[ TS = 25704 \]

Aggregation of unknown number of flows with the bandwidth and delay figures determined. So the requirement is how many flows can be aggregated and transported optically with certain bandwidth and delay requirements. For this purpose, TS payload and TS duration from equations 2 and 3 are replaced in the TS size-latency linear formulas (using the relevant rate and instead of “x” and “y”), which create quadratic equations (eq.6) with "a" (eq.7), "b" (eq.8), "c" (eq.9) and leaving TS_n being the unknown variable. Using the TS_n value TS_{PS} and TS_{OS} are obtained.

\[ a n^2 + bn + c = 0 \]

\[ a = E(-10) \cdot BW_{req} \cdot TS_{OS} + BW_{req} \cdot \]  

\[ OH_D - 5E(-10) \cdot BW_{req} \]

\[ b = 5E(-10) \cdot L_R + BW_{req} \cdot Del_{req} - TS_{OS} \cdot L_R \]

\[ c = -Del_{req} \cdot L_R \]

Algorithm IV: Point-to-point/Highest throughput:
In this case the requested connectivity is for highest throughput (greater than the maximum achieved in EoSWDM > 9.5Gb/s) or the one with the lowest latency (<2µs) for a single connection, and EoWDWM will be used.

The sub-networks are composed of function programmed nodes, which are interconnected using shortest path calculations in the topology. Depending on the mode of operation (EoWDWM and EoSWDM with TDM parameters) routing and resource allocation algorithms of RWA and RWTA are instantiated: Given the network graph of \( G = (V; E) \) with \( V \) indicating the vertices and \( E \) as the edges, Each “E” also holds a database of \( \omega_i \) wavelengths (\( \omega_i \in W \)) where each \( \omega \) is composed of frames and times slices.

Next and in the operation stage, The algorithms of RWT and RWTA are used to dynamically allocate and delete resources to provision light paths between the ToRs in the sub-network: a request of \( R = \{s,d,b,u\} \) dynamic provisioning over the sub-network asks to find a path between the source nodes “s” and destination node(s) of “d”, with a requested bandwidth of “b” for a duration of “u” over any available \( \omega_i \).

**Algorithm III. Input: bandwidth and latency/Output: TS_n, TS_{PS} and TS_{OS}:** Aggregation of unknown number of flows with the bandwidth and delay figures determined. So the requirement is how many flows can be aggregated and transported optically with certain bandwidth and delay requirements. For this purpose, TS payload and TS duration from equations 2 and 3 are replaced in the TS size-latency linear formulas (using the relevant rate and instead of “x” and “y”), which create quadratic equations (eq.6) with "a" (eq.7), "b" (eq.8), "c" (eq.9) and leaving TS_n being the unknown variable. Using the TS_n value TS_{PS} and TS_{OS} are obtained.

\[ a n^2 + bn + c = 0 \]

\[ a = E(-10) \cdot BW_{req} \cdot TS_{OS} + BW_{req} \cdot \]  

\[ OH_D - 5E(-10) \cdot BW_{req} \]

\[ b = 5E(-10) \cdot L_R + BW_{req} \cdot Del_{req} - TS_{OS} \cdot L_R \]

\[ c = -Del_{req} \cdot L_R \]

Algorithm IV: Point-to-point/Highest throughput:
In this case the requested connectivity is for highest throughput (greater than the maximum achieved in EoSWDM > 9.5Gb/s) or the one with the lowest latency (<2µs) for a single connection, and EoWDWM will be used.

The sub-networks are composed of function programmed nodes, which are interconnected using shortest path calculations in the topology. Depending on the mode of operation (EoWDWM and EoSWDM with TDM parameters) routing and resource allocation algorithms of RWA and RWTA are instantiated: Given the network graph of \( G = (V; E) \) with \( V \) indicating the vertices and \( E \) as the edges, Each “E” also holds a database of \( \omega_i \) wavelengths (\( \omega_i \in W \)) where each \( \omega \) is composed of frames and times slices.

Next and in the operation stage, The algorithms of RWT and RWTA are used to dynamically allocate and delete resources to provision light paths between the ToRs in the sub-network: a request of \( R = \{s,d,b,u\} \) dynamic provisioning over the sub-network asks to find a path between the source nodes “s” and destination node(s) of “d”, with a requested bandwidth of “b” for a duration of “u” over any available \( \omega_i \).
VIII. EXPERIMENTAL SETUP AND RESULTS

The testbed setup is shown in Fig. 7. It consists of the network hardware including FPGA and fast switches, and the MS-SDN controller software modules which are placed in server and VMs. The FPGA platform consists of FPGA HTG-V6HXI-100G and an extender card HTG-SFP-PLUS-MDL. It uses 1300nm SFP+ for 10GE on the aggregation side, and 1546nm DWDM channel with SFP+ for inter rack EoSWDM/EoWDM transport. Four 2x2 PLZT fast switches with switching gap time of 10ns were used in the test bed. The switches were cascaded to make for multi hop connectivity.

The controller, the FPGA control and debug, resource allocation tool and the node agents were put placed in virtual machines, and installed in a server with 2 intel i7 processors and 24 GB RAM.

Fig. 9a shows the IP packets between the VMs captured using Wireshark. Different VMs for different tasks are identified with different IP addresses (101: controller, 102: allocation application, 103: Griffin Function Planner, 104: Virtual Node Manager). A XML request is generated in the controller (Job Request), which is then parsed and passed to the Griffin Function Planner approach (Alg.II). The response of the Function Planner module with a HTTP POST message (Wireshark message No 9). The acknowledgement sent back to the controller confirms the reception of the command. The response of the POST communication is the Griffin Function Planner calculations including the sub-network to be used, $T_{S_S}$, $T_{P_S}$. After this stage a Routing and Resource Allocation algorithm is instantiated (No 20) and the sub-network information are then passed to it in Operation stage (No 37). This allocation module is then kept in a loop until the end of the network slice operations. The requests coming to the controller for allocation over this network are bypassed to the allocation engine. The outcomes of the allocation applications are then sent back to the controller, which then parses them and sends them to node agent (No 42).

Node agent then creates an Ethernet frame with the reconfiguration and control information in it and sends it to the FPGA (Fig. 5b). The host specs for agents were: VMs with dual core 2.4GHz with 4GB RAM, and the time it takes the RWTA to allocate time slices following application is about 35ms. The communication between the controller and the node agent is about 28ms, and the time it takes for the agent to parse and send the info to the FPGA is less than 100μs. So the time it takes the controller to add (and remove) a connection is about 53ms. On the other hand the function planner takes about 40ms to reply back to the controller requests for calculating the EoS attributes. It should be noted these figures are inclusive of the applications running time plus the time needed for communicating through the REST interfaces. Also it should be noted as the network scales, the timings can vary.

Following the EoSWDM usecases and their corresponding algorithms a number of sample requests have been applied to the controller to compare the theoretical and actual experimental output values. Latency and throughput for each case is calculated and displayed in Fig. 8, after calculating the TS information ($T_{S_2}, T_{P_2}$) which are displayed in the center of the figure. Blue bars represent the theoretical calculations based on the input values, and the red bars represent the actual experimental measured data. The first 3 testcases of 1-1,1-2, and 1-3 have bandwidth and number of TSs as inputs, and are addressed by the Alg. I. Based on the bandwidth requested per each TS, which can be seen on the blue bars in the bandwidth bars (800,500, and 400 Mbps for testcases 1-1, 1-2, and 1-3, with $T_{S_N} = 6,8,5$ accordingly), $T_{S_2}$ is calculated considering the network hops for each case. Each $T_{P_2}$ is found from the TS size-latency charts matched with the aggregate bandwidth calculated. Testcases of 2-1, 2-2, and 2-3 are requests for Griffin programming using the latency requirements for n number of connections and Alg. II corresponds with it. This approach aims to find the Tsd from the delay (0.5, 1, and 0.8 ms for testcases 2-1, 2-2, and 2-3, with $T_{S_N} = 4,7,4$), and then calculates the $T_{P_2}$ from the charts. Testcases 3-1 and 3-2 request for the maximum number of connections to be deployed by EoSWDM while meeting the delay and bandwidth requirements, and are applied to Alg.III. They both ask for 1 ms delay and 1Gb/s of bandwidth. The difference between the two cases is the number of hops (1 vs 2). One general observation from the experimental results is the expected opposite trends between data rate and latency: the higher the aggregate bandwidth is, the lower the latency is and vice versa. These effects are also followed in the blue bars as expected. Another observation is that the lower latency however does not mean smaller $T_{P_2}$ necessarily, for instance small $T_{P_2}$ with big $T_{S_N}$ means longer frames and longer latency as the result. Comparing the delays of two cases of 1-2 and 1-3, 1-2 has smaller slot latency compared to 1-3, but has longer end to end delay. Also the
effect of increased overhead bits on the latency can be seen when comparing 3-1 and 3-2 with 1 and 2 hops accordingly. The captured results on the red bars of bandwidth and delay show similar values, however slightly higher latency for 3-2 which has longer frames. The calculated numbers of TSs for these cases are fractional, however for operation they were rounded down so not to trespass the bandwidth and latency limits. The variation between the theoretical and experimental values (average of 3.5% for bandwidth, and 27% for latency) however is expected to some extent due to some of the following reasons:

1- Limited samples for creating the TS size-latency charts (Fig. 6) which could be captured and verified experimentally using the testbed. The rest of the data are synthetic.

2- The estimation error in TS size-latency charts: A linear model used for the size-latency charts to simplify the proof of concept however the charts are nonlinear specifically in cases such as smaller TSs.

3- Rounding error: when matching the request aggregate bandwidth with the available charts, the requested values are rounded to available samples. This diversion cause’s error which can be variable depending on the spacing’s between the rates and so as explained in previous points.

4- Bandwidth calculation is less prone to errors than the delay calculations. Bandwidth per slice is achieved by the time portion allocated to the slice in a frame excluding overheads. The delay however can get impacted randomly from multiple stages of buffering, patterns on burst aggregation and transmission, even issues such as hardware temperature conditions. Besides for shorter frames, latency experienced end-to-end is less dependent on frame size which decreases the accuracy of the model.

Fig. 10 and Fig. 11 are demonstrating dataplane results for the two applications of deploying the Griffin for creating intra and inter-cluster communications in DCN, and that how it can be flexibly modified to adapt to different application requirements to be mapped to sub-networks created within or between the rack clusters. In Fig. 10 it is shown how the TDM features in EoSWDM service are configured in a way that bandwidth can be preserved over paths with nil hops and three hops in expense of delay variations. This set up demonstrates a case in which the application is not networking sensitive, and we can build the cluster of servers inside DC where the bandwidth is guaranteed between the nodes of a distributed configuration as in HPC applications.

Fig. 11 on the other hand demonstrates, how EoSWDM TDM features can be tuned by compromising bandwidth but keeping the frame length constant between end points latency can stay deterministic. In this case the optical aggregation benefits of EoSWDM service have been tuned in a way the the time limits can be guarantied between the nodes in the DC latency sensitive applications such as interactive media.

IX. CONCLUSION

In this work we have introduced Griffin, a programmable intra-DC ToR technology and architecture, which by design and implementation demonstrates a highly programmable optical intra-DC switching platform of hardware and software components. The proposed platform addresses the aggregation layer in DCN where dominating east-west traffic between racks can be handled more resource efficiently and therefore enhancing scalability. Griffin nodes support Ethernet transport using sub-Wavelength (EoSWDM) and over Wavelength (EoWDM), which enables them to virtually providing different switching networks for different applications with all-optical switching between the ToRs. The EoWDM service provides fastest with highest throughput best suiting delay/bandwidth sensitive cloud services. However EoSWDM service enables optical aggregation of multiple flows in the expense of delay or bandwidth while increasing resource efficiency. Griffin with flexible TDM functions of the EoSWDM additionally offers greater application awareness by enabling adjustment of frame length, time slice duration, and the overheads. A unique, modular and multi-stage SDN controller is developed for this platform which programs the Griffin nodes to create sub-networks of EoWDM and EoSWDM with programmable TDM features to best meet the application requirements in the planning stage. It then uses different types of allocation mechanisms to operate the composed sub-networks in the operation stage. We also demonstrated how this platform was characterised in different modes of operation and how external algorithms were deployed to make full use of Griffin programmable features.

ACKNOWLEDGEMENT

This work is supported by LIGHTNESS and CONTENT projects funded by the European Commission, EPSRC funded
Hyperhighway, EPJ/L027070/1 SONATAS: Synthetic On-Chip and Off-Chip Optical Network System and EPJ/L020009/1 Towards Ultimate Convergence of All Networks (TOUCAN).

REFERENCES


Bijan Rahimzadeh Rofoee received BSc in electronics and control engineering from University of Tabriz in Iran, and MSc in Telecommunications from the University of Essex, ranked first and awarded with IEEE chapter prize, and PhD from University of Bristol. He’s research focuses on the use of dynamic optical networks, high speed optoelectronics, software defined networks, and network convergence in metro, DC architectures. He has been involved in development and integration activities in several UK and EU projects on advanced optical networks including Carrier Ethernet trial, MAINS, ADDONAS, CONTENT, and has contributed to top tier journal and conference publications in the fields of optical, wireless, and intelligent networking.

Georgios Zervas is currently a Lecturer in Optical and High Performance Networks and member of High Performance Networks group at University of Bristol. He received the MEng and PhD degrees and was a Research Fellow and Lecturer at University of Essex. Georgios is leading the research activities in optical transport and switching, as well as software/hardware programmable systems and networks for backbone and Data-Center/High Performance Computing interconnects. He has published over 150 international journal and conference papers including 9 post-deadline papers and has received best-paper awards. Georgios is TPC member of leading international conferences including OFC, ICC and contributor to standards. He has given numerous invited talks at international conferences, symposia, workshops and standardization fora.

Yan Yan received her B.Eng. degree in 2003 from Dalian University of Technology in China and her M.Sc. degree from the Royal Institute of Technology (KTH) in Sweden in 2007. She is currently a research associate in the High Performance Networks Group at the University of Bristol, UK. Her research interests are in high speed electronics design, programmable networking hardware, network function programmability. She has been involved in a number of UK and EU projects.

Dimitra Simeonidou is Professor of High Performance Networks group at University of Bristol. Dimitra is a leading academic in Optical Networks, Future Internet Research and Experimentation (FIRE), Grid and Cloud Computing, and a founder of Transport Software Defined Networking. She has chaired a number of international conferences and committees across these technical fields. She is the author and co-author of over 400 publications of which many have received best paper awards, 11 patents and standards.