NIST
National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

Links in the GMPLS Lightwave Agile Switching Simulator (GLASS)

Version: Draft 1.0

TABLE OF CONTENTS

1 INTRODUCTION

This document presents the structure of links in the GLASS simulator. The configuration of a link is very important in a network. The simulator provides the user with fully configurable links that contain a high variety of characteristics.
In the first part, the document shows the architecture of the links and in the second part, it explains the different ways to configure them, depending on the level of details wanted.

2 LINK MODELING

This section presents the architecture of the links used in SSFNet followed by the optical links and fibers that have been added by GLASS.

2.1 LINKS

A link in SSFNet models link-layer connectivity among a set of attached interfaces. A link with more than two attached interfaces implicitly performs level-2 routing of IP packets sent on any attached interface [1].
To get more information about the interfaces, refer to [2].
The characteristic of the link is the transmission delay. The user can also specify the CIDR and IP address for this link but SSFNet provides an auto-generation of IP addresses for the interfaces and links in a topology.
What SSFNet does not consider is the failure of a link during the simulation. The GLASS simulator extends this link and includes the possibility to fail a link and provides mechanisms for failure notifications. With this capability, it becomes easy to test the behavior of algorithms and protocols for restoration of a network.
The annex contains the DML format for the component explained in this section.

2.2 OPTICAL LINKS

The optical links in GLASS are based on the extended link, which is explained in the chapter above. That means the capabilities of the regular links are also available for the optical links.

The important difference between the link and the optical link is in the component itself. The link is composed of one cable between two interfaces but the optical link is composed of one or more fibers. The fibers themselves are composed of lambdas. By this difference, the configuration is not the same. In an optical link, it is possible to configure the distance, the fibers, the bit error rate (BER), the number of regenerators, the number of amplifiers (which may include some delays in the transmission), and attributes like the Shared Risk Link Group (SRLG) that plays an important role for the algorithms.

Another attribute that has been added in the optical link is the ID. SSFNet link does not have an ID but in GLASS using ID makes much easier the search and manipulation of the data. That is the reason why GLASS link also include an ID. This ID is mandatory for optical links but not for regular links to be compatible with SSFNet. The ID of the link must be unique in the topology.

The optical links cannot be connected to regular interfaces because of the different structure of this kind of link. They must be connected to an optical network interface card (ONIC). GLASS provides an implementation of these interfaces that can send data over optical links. It is not possible to connect an optical link to more than 2 interfaces like it is possible with a normal link.

The direction of an optical link is not necessary bi-directional like the regular link of SSFNet. The direction depends on the setup of the components of the optical. A link can contain a set of fibers that are all unidirectional and point in the same direction. Then the link is unidirectional. Does at least one of the fibers point in the opposite direction or at least one of the fibers is bidirectional the link is bidirectional too. This also affects the bandwidth of a link. Depending on the direction a link can have a different bandwidth.

Figure 1 shows the general structure of the optical links.

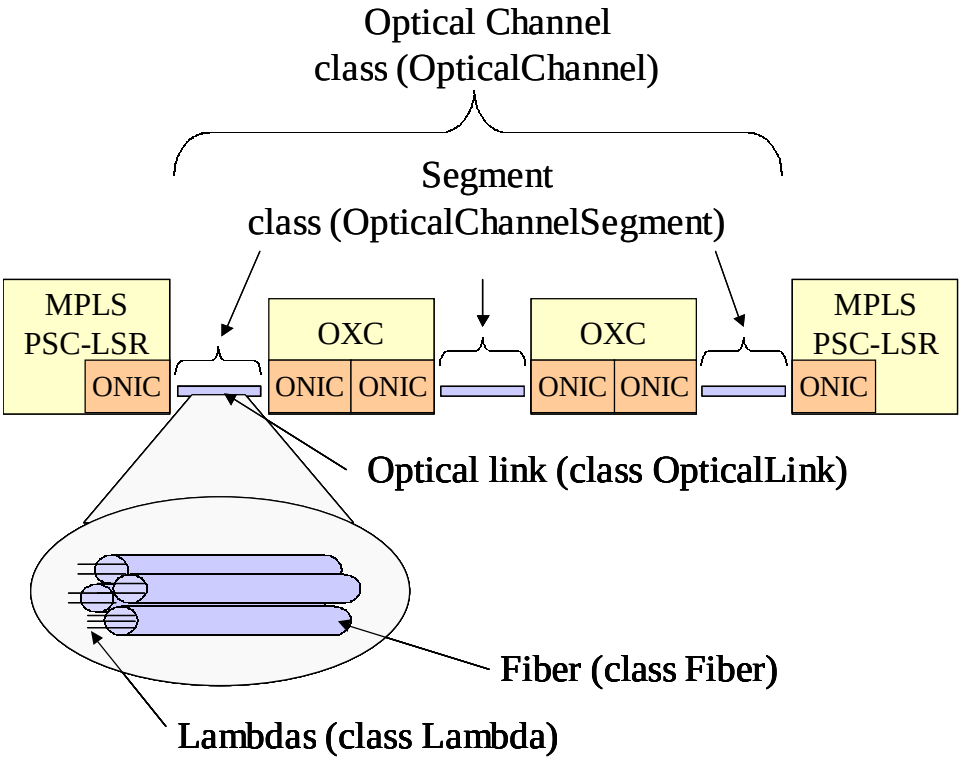

Figure 1: Optical Links

It is possible to connect multiple optical links between two nodes. In this case, the nodes must have multiple optical interfaces.

We also added some special attribute that allows the user to specify if an optical link is going to be used for protection/restoration process or for data (a combination is also possible).

2.3 FIBERS

The fibers describe the content of an optical link. In the design, an optical link can be considered as a bundle of fibers. The fibers in the same link don't necessary have the exact same characteristics.

GLASS supports unidirectional and bi-directional fibers. Currently the use of the bi-directional fibers is only for short distance but probably is the future, the use of these fibers will become more important. That's the reason why this characteristic has been included in the simulator.

As the configuration of the distance is done in the optical link, all the fibers in the assigned optical link have the same distance. It is also possible to specify the protection use of a fiber like the optical link. This is mainly for restoration algorithms. The user can also fail a fiber independently to the link and a notification mechanism is provided to inform the potential target about the link status.
Some physical characteristics are available in order to configure the fiber. For example, GLASS is checking the value of the wavelength inside itself. Some mechanisms are provided to avoid so called "non-sense values". It is not possible to have two lambdas with the same wavelength value inside a fiber. Lambdas must have a distance between two wavelengths. All this parameters are configurable by the user. Currently, the distance between two wavelengths is 0.8 nm and the central wavelength is 1550 nm. That means auto generated lambdas have the following wavelengths: 1550, 1550.8, 1549.2, 1551.6, 1548.4...

To identify a fiber, GLASS uses two mechanisms. On one hand, the fiber has an ID that is unique in an optical link. On the other hand, a fiber port ID has to be defined for each interface, the fiber is connected to.
The following figure shows a basic configuration of fiber in an optical link.

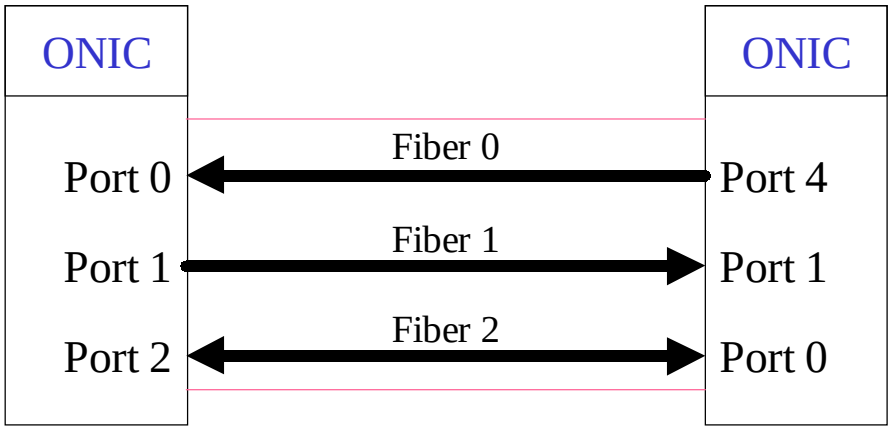

Figure 2: The Fiber

2.4 LAMBDAS

The lambdas are configured independently inside fibers.

An ID that is unique inside a fiber identifies each lambda. The value of the wavelength is unique in a fiber. It is also possible to fail a lambda and to specify if it must be used for protection.

As the user will notice, GLASS does not provide a bandwidth attribute in an optical link. In fact, each lambda has its own bandwidth and the bandwidth of a link will be calculated depending on the configuration. Inside a fiber, it is possible to have different lambdas that have different bandwidth. The bandwidth of a fiber is the aggregation of the lambda's bandwidths that compose it.

The attributes: signal to noise ratio (SNR), the pair and control are also configurable in a lambda. The pair attribute allows the user to include a relationship between two lambdas (for example an input and an output lambda). The control attribute is quite important because it defines whether or not the lambda is used to carry signaling messages. This attribute specifies the "control lambda". A control lambda is only used in the point-to-point connection. This means the information can only be carried to the next hop. The data lambda on the other side carries information on end-to-end connection.

3 MANUAL CONFIGURATION OF OPTICAL LINKS

As explained before, GLASS allows the user to configure optical links, fibers and lambdas, as he wants. This is to be able to simulate a lot of cases. But this can become quite complicated when the topology becomes bigger. That's why, there are many ways to have an easier, quicker configuration. Of course, this will produce some restriction. For example, the bandwidth of the lambdas may be all the same. The next sections will present how to configure manually or automatically an optical link using the DML. Refer to annex for a complete description of DML schemas for the components.

For a full control of the configuration of an optical link, the user has to specify all the characteristics of a link, fiber and lambda. No checking is done, so you can define a unidirectional link for data traffic as well as for control traffic. This is not done automatically. You can also have multiple lambdas in a fiber that have different bandwidth. Table 1 is an example of a fully configured optical link.

OpticalLink [
  id 1
  distance 10
  srlg 1,5
  attach 1(1) attach 2(0) delay 0.001
  Fiber [
  id 1
  receiverHostId 2
  senderPortId 2
  receiverPortId 1
  Lambda [
  Id 0
  wavelength 1549.2
  bandwidth 4
  control true
  ]
  Lambda [
  Id 1
  wavelength 1550.8
  bandwidth 2.5
  control false
  ]
  ]
]
	Optical link 1 between OXC1 and OXC2. 2 fibers with 2 lambdas.
	Distance = 10 km
	This link belongs to SRLG 1 and 5.
	Attach interface 1 of node 1 to interface 0 of node 2.
	Only one unidirectional fiber to send messages to node 2 (receiverHostId=2).
	Use port 2 (in node 1) to send through this fiber
	Receive messages on port 1 in node 2.
	Define a control lambda.
	
	Define a data lambda.

Table 1: Manual Configuration of an Optical Link 1

Figure 3 shows the graphical representation of this configuration.

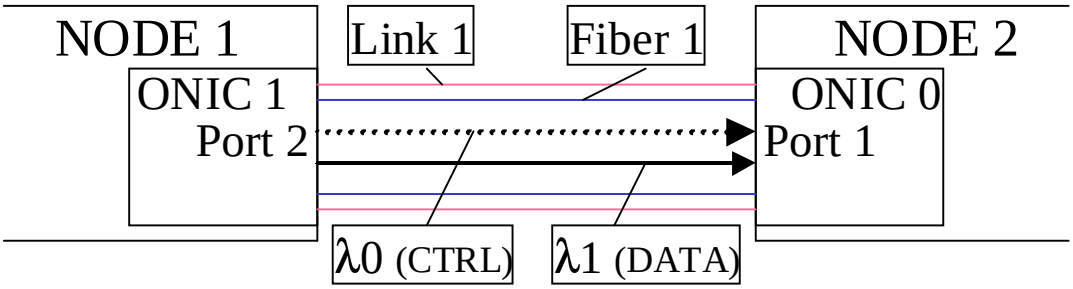

Figure 3: The Link of the Manual Configuration 1

OpticalLink [
  id 5
  distance 10
  srlg 2
  attach 4(1) attach 2(6) delay 0.001
  Fiber [
  id 1
  receiverHostId 2
  senderPortId 2
  receiverPortId 1
  Lambda [
  id 1
  wavelength 1550.8
  bandwidth 2.5
  control false
  ]
  ]
  Fiber [
  id 2
  bidirectional true
  receiverHostId 2
  senderPortId 6
  receiverPortId 2
  Lambda [
  id 1
  wavelength 1550.8
  bandwidth 2.5
  control true
  ]
  Lambda [
  id 3
  wavelength 1549.2
  bandwidth 5
  control true
  ]
  ]
]
	optical link 5 between OXC4 and OXC2.
	2 fibers with 2 fibers.
	Distance = 10 km
	This link belongs to SRLG 2.
	Attach interface 1 of node 4 to
	interface 6 of node 2.
	Unidirectional fiber to send messages
	to node 2 (receiverHostId=2).
	Use port 2 (in node 4) to send through
	this fiber
	Receive messages on port 1 in node 2.
	Define a data lambda.
	
	Bidirectional fiber to send messages
	in both directions. To know where the
	ports are, we still have to define a
	receiver.
	Use port 6 (in node 1) to send through
	this fiber
	Receive messages on port 2 in node 2.
	Define a control lambda with a
	bandwidth of 2.5Gbps.
	
	Define a second control lambda with a
	bandwidth of 5 Gbps.
	The direction of each lambda is
	defined by the setting of the
	switching table.

Table 2: Manual Configuration of an Optical Link 2

Figure 4 shows the configuration of this optical link.

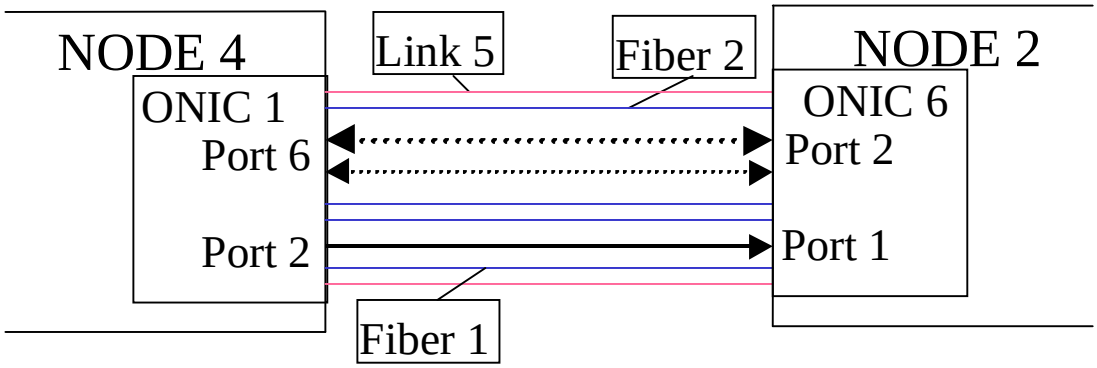

Figure 4: The Link of the Manual Configuration 2

4 AUTO CONFIGURATION OF OPTICAL LINKS

4.1 AUTO COMPLETION OF FIBERS IN OPTICAL LINK

To make the configuration of optical links easier, GLASS provides auto configuration of the fibers. When the user does not want to specify each lambda, he just has to specify the bandwidth of the whole fiber, the number of lambdas and control lambdas he wants to have in this fiber (one for unidirectional and two for bi-directional fiber).

Examples:

Fiber [
  Id 2
  receiverHostId 2
  senderPortId 2
  receiverPortId 1
  bandwidth 12.5
  noLambdas 5
]
	This DML configuration will create 5 lambdas in the fiber 2. Each lambdas will have a bandwidth of 2.5 Gbps.
	As the attribute controlLambda has not been defined, we will set one of the lambdas as a control lambda. It means, the user will have 10 Gbps for data.

Fiber [
  Id 1
  receiverHostId 5
  senderPortId 4
  receiverPortId 2
  bandwidth 10
  noLambdas 4
  controlLambda false
]
	This configuration will generate 4 lambdas in the fiber 1 without control lambda.
	Like in the previous example, the user has 10 Gbps for data.

Fiber [
  id 1
  bidirectional true
  host2ID 5
  senderPortId 4
  receiverPortId 2
  bandwidth 10
  noLambdas 4
  controlLambda true
]
	This configuration will generate 4 lambdas in the fiber 1 with control lambdas.
	The default number of control lambdas in a bi-directional fiber is 2.
	The user has 5 Gbps for data in 2 lambdas.

Fiber [
  id 5
  host2ID 1
  senderPortId 4
  receiverPortId 2
]
	This configuration will generate 5 lambdas in the fiber 5 with control lambda.
	The default number of control lambdas in a unidirectional fiber is 1.
	The user has 10 Gbps for data in 4 lambdas and 2.5Gbps in control lambda.

4.2 AUTO COMPLETION OF OPTICAL LINKS

This is the highest level of auto completion. When the user doesn't want to care about the fibers and the lambdas, he just has to specify the link Id, the attached interfaces (and if necessary the distance and delay) but not the fibers.

The result is the creation of 2 unidirectional fibers (one for each direction). Each fiber contains one control lambda and nine data lambdas. All the lambdas have a bandwidth of 2.5 Gbps.

The manual configuration is higher prior as the automatic configuration. This allows mixtures between auto and manual configures components. GLASS first processes manual configuration and then the auto configuration to avoid configuration conflicts of Ids.

5 ANNEX: DML SCHEMA FOR OPTICAL LINKS, FIBERS AND LAMBDAS

NOTE: the link description is a copy of the documentation provided by SSF in [1].

link [
  attach %S2
  delay %F
  alignment %S
  cidr %S
  ip %S
]
	Net.link specifies a level 2 network connecting a set of attached host and/or router network interfaces. A link should be understood as a "link layer". It must have two or more link.attach attributes naming the attached network interfaces. Does not mandate what level 2 (i.e., link layer) routing or broadcast capabilities are present, or what symmetries of interface properties must exist (e.g., matching bitrates) -- that's left to the interfaces involved. link.delay, with value in seconds, represents the contribution of the link itself to total transmission delay of an IP packet. It does not include interface-specific sources of delay such as NIC frame processing latencies or buffering (see Net.host.interface). Attributes Net.link.cidr, and Net.link.ip are explained in the section on CIDR IP address blocking and section on IP address assignment, respectively. It is recommended not to use them and let SSF.Net.Net automatically allocate IP addresses unless you know what you are doing.
	Attribute Net.link.alignment in SSFNet 1.1 causes different synchronization behavior for point-to-point links (class SSF.Net.ptpLinkLayer) and for LANs, i.e., links with three or more attached interfaces (class SSF.Net.lanLinkLayer). The former does not contribute to the multi-timeline synchronization, while the latter does (via the link delay attribute).

OpticalLink [
  _extend
  .schemas.Net.link
  id %I1!
  ber %I
  distance %I
  noAmplifiers %I
  noRegenerators %I
  protection %S
  protectionMode %S
  failure %S
  srlg %S
]
	OpticalLink
	An optical link extends the class link. This link must have an integer ID.
	Optional attributes :
	ber: bit error rate. Default value=0
	distance: length of the link in meters
	nbAmplifier: number of amplifiers on the optical link
	nbRegenerator: number of regenerators on the optical link
	protection: indicates if the link is used as a protection link. Default value : False
	protectionMode : defines the mode (NEVER, ONLY, SHARED) associated to the attribute protection
	ONLY: The algorithm is not able to change the protection attribute, the value is "true". The component is not allowed to be used in a non backup scenario.
	SHARED: The algorithm is able to change the protection attribute back and forth. The component can be used for everything.
	NEVER: The algorithm is not able to change the protection attribute, the attribute is "false" for ever, the component is not allowed to be used in a backup setup.
	The default values are "protection false" and "protectionMode SHARED".
	failure: indicates if the link is failed. Default value: False
	srlg: The Shared Risk Link Group. This attribute is an integer list that represents all SRLG this link share. The srlg attribute is a list of integer seperated by comma.
	The user does not have to include fibers in the configuration of an optical link. If so, we automatically create 1 unidirectional fiber with 10 lambdas (included 1 control lambda) in both direction.

fiber [#general attributes id %I1! failure %S protection %S protectionMode %S bandwidth %I noLambdas %I controlLambda %S startWavelength %I deltaWavelength %I] fiber [#attributes for a bidirectional fiber bidirectional true host2ID %I1! host1PortId %I1! host2PortId %I1!] fiber [#attributes for a unidirectional fiber bidirectional false receiverHostId %I1! senderPortId %I1! receiverPortId %I1!]	Fiber: defines the structure of a fiber. The ID of a fiber must be unique in a Host. Attribute receiverID represents the ID of the host that will receive the messages sent through this fiber. The host can be a router or an OXC. Optional attributes failure indicates if the fiber is failed (default value : false). The optional attribute protection shows if the fiber is used to protect another fiber. protectionMode : defines the mode (NEVER, ONLY, SHARED) associated to the attribute protection. See the information of these attributes in the link. If attribute bidirectional is false, use the attribute receiverHostId to specify the direction of the fiber. If it is true, the user has to specify the attribute host2ID to be able to assign the good port ID of each interface. The attribute host1PortId can be used in lieu of senderPortId and host2PortId in lieu of receiverPortId. Optional attributes bandwidth , noLambdas and controlLambda : with these three attributes, the user can ask to automatically generate the lambdas inside the fiber. If they are not used, the user has to define each lambda. Optional attributes bandwidth , noLambdas and controlLambda (default true) : with these three attributes, the user can ask to automatically generate the lambdas inside the fiber. If they are not used, the user has to define each lambda. If controlLambda is true, one control lambda is inlcuded for unidirectional fiber and 2 for bidirectional fiber (default value : true). Optional attibutes startWavelength and deltaWavelength are used to define the first wavelength to use and the space between 2 wavelengths when lambdas are automatically generated and to check the user-defined wavelength of the lambdas.
lambda [id %I1! wavelength %I1! bandwidth %I control %S1! snr %I failure %S protection %S protectionMode %S pair $S]	The ID of a lambda is unique inside a fiber. bandwidth: defines the bandwidth of a lambda (default value 2.5 Gbps). Control (optional) : indicates if this lambda is used for control messages. At least one lambda must be define in a fiber as a control lambda. The snr attribute is the Signal to Noise Ratio. Failure: indicates if the lambda is failed. If the attribute protection is true, it means that this lambda can be used for protection purposes. protectionMode : defines the mode (NEVER, ONLY, SHARED) associated to the attribute protection. See the information of these attributes in the link. pair: defines a relation ship to another lambda that must have the opposite direction and must be inside the same link. To define a pair, use the structure fiberID(LambdaPairID).

6 ACRONYMS

GLASS	GMPLS Lightwave Agile Switching Simulator
SSF	Scalable Simulation Framework
CIDR	Classless InterDomain Routing
IP	Internet Protocol
SRLG	Shared Risk Link Group
BER	Bit Error Rate
ONIC	Optical Network interface Card
SNR	Signal to Noise Ratio

7 REFERENCES

[1] Standard SSFNet DML attributes

By Renesys Corporation

URL: http://www.ssfnet.com/InternetDocs/ssfnetDMLReference.html

[2] Nodes in GMPLS Lightwave Agile Switching Simulator (GLASS)

By NIST/ANTD.