****

ATM Forum Document Number: 97-1089

****

Title: Modifications to Appendix B and Sections 3.1.7 and 3.2.7 of Performance Testing Baseline Text.

****

Abstract: Appendix B of the performance baseline text describes ways to generate scalable configurations such that a large switch with many ports can be fully loaded and tested with a minimum number generator(s). The text that was accepted in the September 1997 meeting of the ATM Forum requires using loopbacks that connect the output of a port to the input of the same port. Some switches do not allow this. Therefore, another more generalized method of preparing scalable configurations is presented in this contribution. This method allows the flexibility of connecting the output the same port or another port. We propose to replace the entire text of Appendix B with the main text of this contribution.

This contribution also includes changes that should be done in Sections 3.1.7 and 3.2.7 of the Performance Testing Baseline Text as consequences of acceptance of new Appendix B.

****

Source:
Arjan Durresi, Raj Jain, Gojko Babic, Justin Dolske.
The Ohio State University

Raj Jain is now at Washington University in Saint Louis, jain@cse.wustl.edu http://www.cse.wustl.edu/~jain/

The presentation of this contribution at the ATM Forum is sponsored by NASA Lewis Research Center.

****

Date: December 1997

****

Distribution: ATM Forum Technical Working Group Members (AF-TEST, AF-TM)

****

Notice:

****

Appendix B: Methodology for Implementing Scalable Test Configurations

B.1. Introduction

In Sections 3.1.5 and 3.2.6 of the baseline text, a number of connection configurations have been presented for throughput and latency measurements. In most of the cases, these configurations require one traffic generators and/or analyzers for each port. Thus, the number of generators and/or analyzers increases as the number of ports increases. Since this equipment is rather expensive, it is desirable to define scalable configurations that can be used with a limited number of generators. Sections 3.1.7 and 3.2.7 present several scalable test configurations. However, one problem with scalable configurations is that there are many ways to set up the connections and measurement results could vary with the setup.

In this appendix, a standard method for generating scalable configurations is defined. Since the methodology presented here applies to any number of traffic generators, it can be used for non-scalable (full-scale) test configurations as well. Performance testing requires two kinds of virtual channel connections (VCCs): foreground VCCs (traffic that is measured) and background VCCs (traffic that simply interferes with the foreground traffic). The methodology for generating configurations of both types of VCCs is covered in this appendix.

The VCCs are formed by setting up connections between ports of the switch. The connections are internal through the switch fabric and external through wires or fibers, depending on the port technology. An external connection between two switch ports is referred in this appendix as a **wire W**. The methodology presented here has two phases. During the first phase the switch ports are connected externally by numbered wires as given in the section B.2. The second phase consists of setting up PVCs, i.e. internal connections, between appropriate ports as explain in section B.3.

The sequence of concatenated connections (internal and external) is called a **VCC Chain**. For example, the VCC shown in Figure B.1 is formed by setting a VCC chain starting from P1 IN, passing through wires W1, W2, W3, which are internally connected, and ending at P1 OUT. P1 IN is connected to the generator and P1 OUT is connected to the analyzer. Each wire connects a pair formed by an output port and an input port, so W1 connects P2 OUT to P3 IN, W2 connects P4 OUT to P2 IN and W3 connects P3 OUT to P4 IN. This VCC chain is indicated by the notation P1-W1-W2-W3-P1. This notation implies a unique configuration of internal connections. In Figure B.1, external connections are shown by thick lines while the interval connections are shown by thin lines. This notation is followed throughout this appendix.

Another possible configuration for this "n-to-n single generator scalable configuration" would be P1-W2-W1-W3-P1. For an n-port switch, there is a maximum of (n-1)! possible configurations that can implement this configuration.

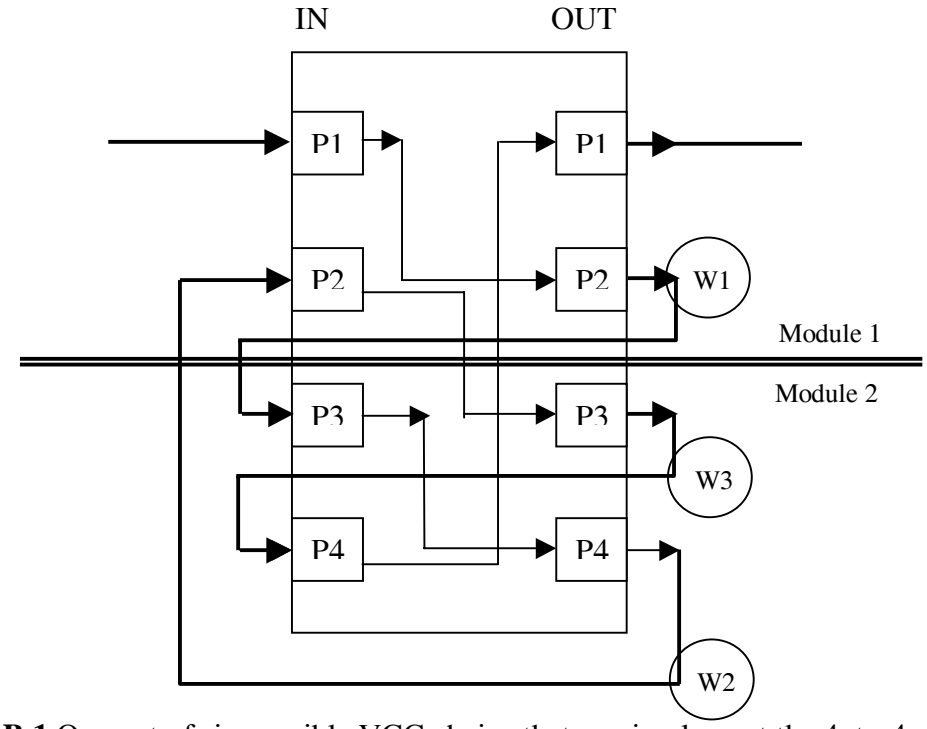


Figure B.1 One out of six possible VCC chains that can implement the 4–to-4 straight configuration with a single generator.

The four-port switch shown in Figure B.1 consists of two modules with two ports each. The measured performance may depend upon the number of times the VCC chain passes from one module to the other and may be different for different configurations.

At the end of this appendix, the pseudocode for a computer program is presented that allows generating a standardized port order for all connection configurations. This methodology (pseudocode) generally creates VCC chains that cross the modules as often as possible while still keeping the whole process simple.

B.2. Implementation of External Connections

The methodology for implementing the external connections consists of the following three steps:

1. Numbering the ports
2. Identifying the ports connected to generators and analyzers
3. Numbering the wires

These steps are now explained.

Step 1. Numbering the Ports:

Consider a switch with several modules of different port types. The ports could be different in speed and/or technology. Each module may have a varying number of ports. For example, a switch may have two modules of eight and six 155 Mbps single-mode fiber ports, respectively, another module with eight 155 Mpbs UTP ports and a fourth module with six 25-Mbps UTP ports. In order to number these ports, the first step is to group the modules of the same port type, then generate a schematic of modules placed one below the other. The schematic should be drawn such that the modules inside the group are arranged in a decreasing order of number of ports. Then the switch ports are numbered sequentially inside the groups, column wise, starting from the top left corner of the schematic. Numbering of each group continues the numbering of the previous group. This port numbering helps in creating VCC chains that cross modules as often as possible. The port numbers obtained this way are represented by **Pi** in this appendix.

Figure B.2 shows an example of port numbering. The modules are divided into three groups. The first group consists of 155-Mbps single-mode fiber modules, the second group consists of 155-Mbps UTP module, the third group consists of 25-Mbps UTP module. The ports of the first group are numbered sequentially along the column from P1 through P14 as shown in Figure B.2. The ports of the second group are then numbered sequentially as P15 through P22. The ports of the third group are numbered similarly as P23 through P28.

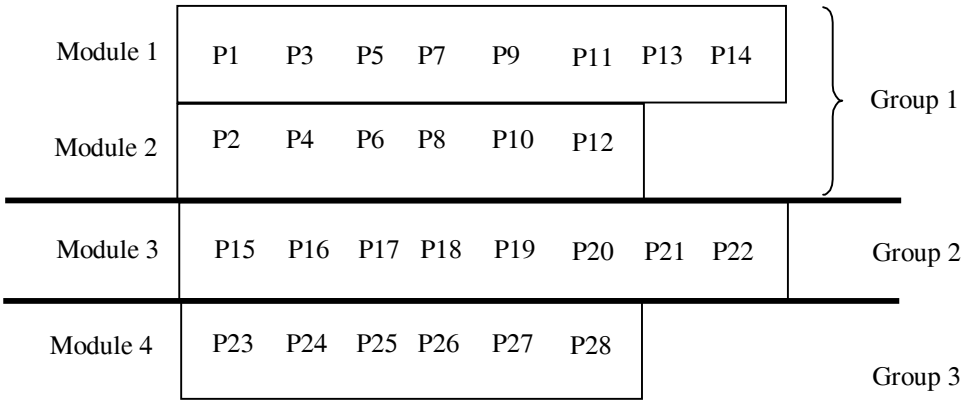


Figure B.2 Example of port numbering.

Step 2. Identifying the ports connected to the generators and/or analyzers:

In general it is possible to design a scalable configuration for any given number of generators and analyzers. These can be connected to any input/output ports. However, the starting/ending port should be chosen in such a way to avoid the case of having only one port left over in a group. This is necessary because that port cannot be connected externally to any other port. This condition does not apply if the switch allows loopbacks.

Step 3. Numbering of Wires:

After the selection of input and output ports, the remaining ports have to be connected in pairs formed by the output of one port and the input of another port. In connecting the port pairs and in numbering the respective wires the following rules are applied:

1. In each group start with the first output port available (that has not been externally connected yet). Increase the port number by one until a port is found whose input is available. This input is connected to the output of the ouput port chosen previously.

   If a scaleable configuration with loopback is desired and is allowed by the switch, the output of a port can be connected to the input of the same port. The rest of the methodology of this appendix applies to this case also.

   This is continued until all output ports have been connected to other input ports or to analyzers.

2. The external connections formed above are numbered sequentially as W1, W2, ...The only restriction is that the end of wire Wi and the beginning of W(i+1) must be different ports. If the next external connection begins with the same port as the end of the previous wire, the next external connection is skipped for this round and may be included in the next round. In general, several rounds may be required to number all the wires. The restriction also applies to the last wire. Thus, the port at end of the last wire should be different from the port at the beginning of the first wire. If this happens then swapping the labels of the last two wires may solve the problem.

The following example illustrates this step.

Consider the (n-1)-to-(n-1) straight configuration required for the background traffic in latency measurement. Suppose the switch has two modules with four ports each of the same speed and technology as shown in Figure B.3.

Step 1. There is only one group, because all ports are of the same speed and technology. The ports are numbered as shown in Figure B.3.

Step 2. For the foreground traffic: P2 IN is arbitrarily selected to be connected to the generator and P1 OUT is connected to the analyzer. For background traffic: P1 IN is connected to the generator and P2 OUT is connected to the analyzer.

Step 3. The first output port available is P3 OUT. It is connected externally to P4 IN. P4 OUT is then connected to P5 IN, and so on. Finally, P8 OUT is connected P3 IN. Figure B.4 shows these external connections.

The next step is to number the wires. The first wire connecting P3 OUT to P4 IN is labeled W1. The next wire connects P4 OUT to P5 IN. However, it cannot be labeled W2 because its input port is the same as the output port of the previous numbered wire W1. So this wire is skipped in this round. The next wire connecting P5 OUT to P6 IN is labeled as W2. The next wire connecting P6 OUT to P7 IN has to be skipped for the same reason. The wire connecting P7 OUT to P8 IN is labeled W3. The wire connecting P8 OUT to P3 IN is skipped. This finishes the first round. The unlabeled wires are considered in the second round. The first unlabeled wire connecting P4 OUT to P5 IN is labeled as W4. The other two remaining wires are labeled as W5 and W6, respectively. The only problem with the labels is that the ending port (P3) of the last wire W6 is the same as the beginning port of the

first wire W1. To avoid this conflict, the labels on wire W5 and W6 are swapped. The resulting wire numbers are as shown in Figure B.4, which also shows the internal PVCs for a latency measurement test. The construction of these internal connections is explained next.

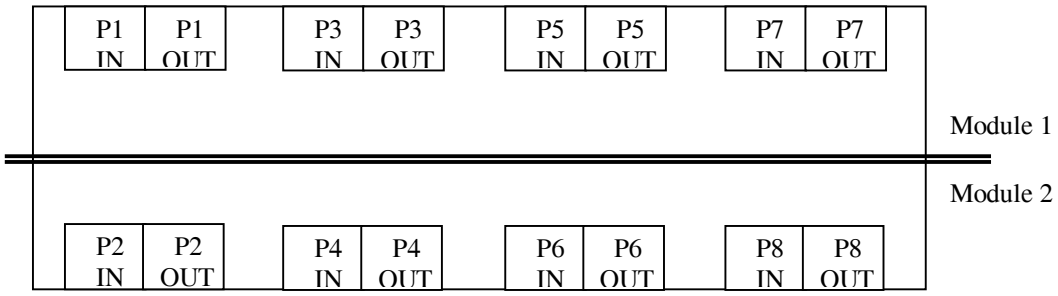


Figure B.3 Port numbering of a switch with 2 modules and 4 ports on each.

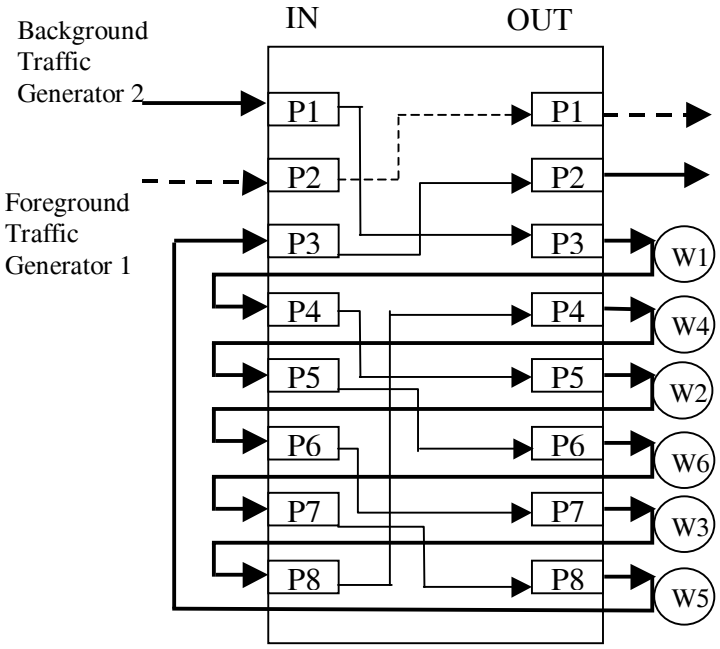


Figure B.4 A 7-to-7 straight configuration with one-generator for the background traffic.

B.3. Implementation of Internal Connections.

All VCC chains are represented by a three-dimensional matrix **CH(i, j, k)**. Matrix index **i** represents the interconnection order among the wires**.** Index **k** represents the generator number and index **j** represents the chain number starting at that generator.

The input ports of all VCC chains are represented by the matrix **CHin(j, k),** where j, k have the same meaning as explained above. In similar way the output ports of the VCC chains are represented by **CHout(j, k).** CHin(j, k) = Px (CHout(j, k) = Px) means that the input (output) part of port Px is used as input (output) port by the j^{th} chain of generator k.

One row **CH(*, j, k)** of the matrix represents a single VCC chain. For example, in Figure B.4, the VCC chain from generator #2 starts at P1, passes through wires W1, W2, W3, W4, W5, W6, and exits at P2, the matrix CH has the following entries: CH(1, 1, 2)=W1, CH(2, 1, 2)=W2, CH(3, 1, 2)=W3, CH(4, 1, 2)=W4, CH(5, 1, 2)=W5, CH(6, 1, 2)=W6 and CHin(1,2)=P1, CHout(1,2)=P2.

The number of intermediate wires in the k^{th} chain is denoted by **NW(k).** In the case of Figure B.4, NW(2) = 6.

For latency measurements, two types of traffic are used: foreground and background. Therefore, at least two VCC chains are required. In order to avoid interference with the foreground traffic, the background VCC chains may or may not use the input and output port of the foreground traffic. If the background traffic does use these ports then it should only be in the directions opposite to that used by the foreground traffic. In our example, Figure B.4, the foreground traffic uses ports P2 IN and P1 OUT as input and output ports, respectively. The background traffic also uses these ports but in the opposite direction, i.e. P1 IN and P2 OUT as input and output ports, respectively.

The remainder of this section is devoted to showing how to obtain scalable configurations the throughput and latency measurements. In all cases, the numbering of ports and wires discussed in Section B.2 is used. The algorithm to implement the internal connections consists of three simple rules:

1. The chains generally go from wire i to wire i+1 unless the wire has already been fully used by other chains.
2. After generating jth chain, (j+1)st chain can be generated simply by adding 1 to each wire index of the jth chain.
3. If there are multiple generators, each generator uses a contiguous subset of wires as source wires. Each generator needs as many source wires as the number of VCC chains starting from it.

B.3.1 n-to-n Straight (Single Generator)

This configuration is used for throughput as well as latency measurements. The scalable versions can be obtained as follows:

a) Throughput measurements: For these tests, we need only a single chain starting from a single generator, i.e., k=1 and j=1. The chain starts from one port, goes through all other ports and exits from the starting port. Therefore, NW(1) is equal to n-1. Any port Px IN and Py OUT can be selected to be the input and output port, respectively.

Figure B.5 illustrates this case for the 2-module 8-port switch. The VCC chain has CHin(1,1) = CHout(1,1) = P1.

The application of the internal connection algorithm is simple. The wires CH(i,1,1) in the VCC chain are selected in numerically increasing order. The wires are included in VCC chain if they are not already used up. After reaching the last wire, the index (i) starts again from the beginning (from i=1).

For CHin(1,1) = CHout(1,1) =P1, the VCC chain is: P1-W1-W2-W3-W4-W5-W6-W7-P1.

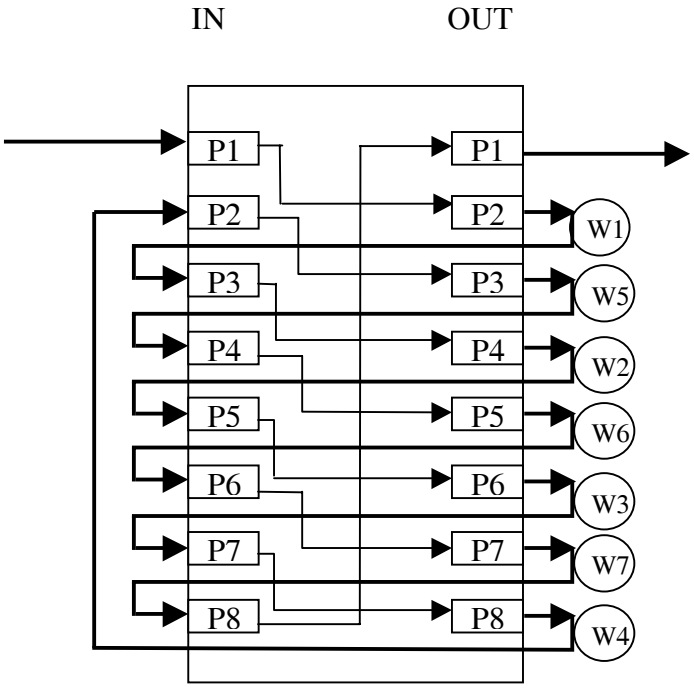


Figure B.5. The 8-to-8 straight configuration with one generator.

b) Latency Measurements: First, consider the case in which the background traffic uses the same input/output ports as the foreground traffic (but in the opposite direction). The background traffic passes through all other ports. Therefore, NW(1) is equal to n-2. The input and output ports coincide respectively with the output and input ports for the foreground.
The foreground and background generators are labeled as generator 1 and generator 2, respectively. If CHin(1,1)=P2 and CHout(1,1)=P1, the foreground chain is P2-P1 and the background chain is P1-W1-W2-W3-W4-W5-W6-P2, having CHin(1,2)=P1, CHout(1,2)=P2.. This connection configuration was presented earlier in Figure B.4.

Now, consider the case in which the background traffic does not use the input/output ports of the foreground. Generator 1 and 2 are used for background and foreground traffic, respectively. In this case, NW(1) is equal to n-3. CHin(1,1) and CHout(1,1) coincide and can be selected from any of the switch ports except CHout(1,2) and

CHin(1,2). For example, the foreground can use the chain P2-P1 and background could use P1-W1-W2-W3-W4-W5-P1. Figure B.6 illustrates this case.

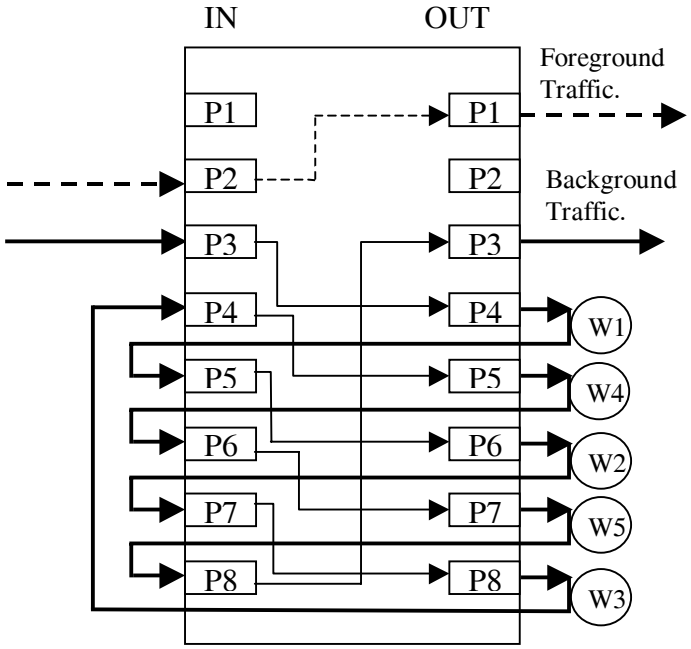


Figure B.6. The 6-to-6 straight configuration with one generator, where the foreground traffic does not share the port with background traffic.

B.3.2. n-to-n Straight (r Generators)

This configuration implements the n-to-N straight configuration with **r** generators.

a) Throughput Measurements: Each generator has one VCC chain. In all there are r VCC chains. Of the n ports, r ports are used as source/destination of these chains. The remaining ports are connected among themselves and their wires are divided among the generators as evenly as possible.

Let **p** =mod(n-r, r)

- For the first **p** VCC chains, the number of intermediate wires NW is equal to the quotient of (n-r)/r plus 1, i.e., $\lfloor (n-r)/r \rfloor + 1$
- For the remaining **(r-p)** VCC chains, NW is equal to the quotient of (n-r)/r, or $\lfloor (n-r)/r \rfloor$
- For all VCC chains, the source/destination ports may be selected from any of the switch ports Px not selected by other VCC chains as a source or destination.

As an example, consider the 8-port switch again. With r=3 generators, p equals mod(8-3, 3) = 2. So, the first two VCC chains have NW=$\lfloor (8-3)/3 \rfloor + 1 = 2$ intermediate wires, and the last chain has NW==$\lfloor (8-3)/3 \rfloor = 1$.

Figures B.7 illustrates the implementation of the VCC chains for this case. First we select the source and destination ports:
Port 1 is the input and output for the first chain, so CHin(1,1) = CHout(1,1) = P1
Port 2 is the input and output for the second chain, so CHin(1,2) = CHout(1,2) =P2
Port 3 is the input and output for the third chain, so CHin(1,3) = CHout(1,3) = P3
These selections have been made to avoid any overlap.

After applying the first three steps of the methodology we obtain the configuration shown in Figure B.7. Then we apply the VCC chain algorithm. Let us start with the VCC chain having port 1 as the source. The first available wire is W1, so CH(1,1,1)=W1, then CH(2,1,1)=W2. This VCC chain has two intermediate wires and so it is now complete. Now we continue with the VCC chain starting at port P2. The next available wire is W3 (because W1 and W2 are fully occupied by the previous VCC chain). So CH(1,1,2)=W3, and then CH(2,1,2)=W4. Similarly, for the third chain, CH(1,1,3)=W5. This VCC chain has only one intermediate wire. The VCC chain implementation is complete.

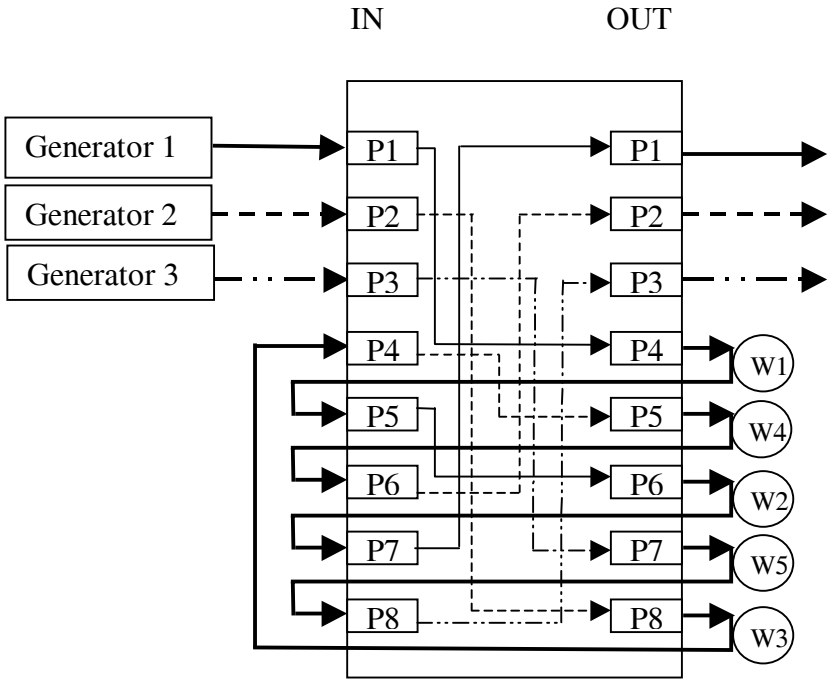


Figure B.7 Implementation of the 8-to-8 straight configuration with 3 generators .

b) Latency Measurements: Consider the case with the background traffic using the foreground ports in the opposite direction. The remaining n-1 ports are connected among themselves and their wires are evenly divided among the r background VCC chains.

Let **p** =mod(n-r-1, r)

- For the first **p** VCC chains, NW is equal to the quotient of (n-r-1)/r plus 1, i.e., $\lfloor (n-r-1)/r \rfloor + 1$
- For the remaining **(r-p)** VCC chains, NW is equal to the quotient of (n-r-1)/r, or $\lfloor (n-r-1)/r \rfloor$

- For one of VCC chains of the background traffic, the input and output ports coincide with output and input port for the foreground traffic, respectively.
- For the other VCC chains, the input and output ports can be selected from any of the switch ports Px not selected by other VCCs

After applying the first three steps of the methodology, we obtain the configuration shown in Figure B.8. Ports P1 and P2 are used by the foreground traffic as output and input ports, respectively.

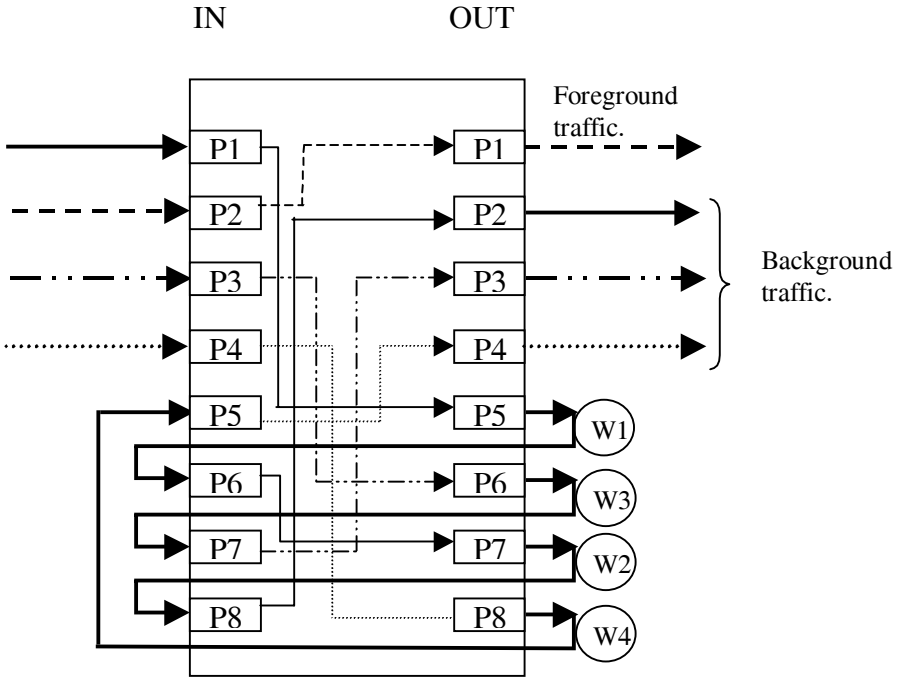


Figure B.8 Implementation of the 7-to-7 straight configuration with 3 generators for background traffic in latency measurement.

Ports P1 and P2 will be used as input and output ports (respectively) by one of the background VCC chains. The other two generators will use port P3 and P4 as the input and output ports, respectively. For the first VCC chain, NW(1) =2 and for the other two VCC chains NW(2) = NW(3) =1. The chains are: P1-W1-W2-P2, P3-W3-P3, and P4-W4-P4.

The configuration for the case when the background traffic does not share the ports with the foreground can be generated by the above procedure by considering the switch having only n-2 ports.

B.3.3. n-to-m Partial Cross (r Generators)

This is a generalization of n-to-m partial cross with 1 generator presented in the baseline. The discussion here applies also for r=1. Also, by appropriately setting r, one can obtain non-scalable (basic) configurations.

a) Throughput Measurements: This configuration has **m*r** VCC chains originating from **r,** where each generator originates **m** VCC chains. Each has a load of $1/m^{th}$ of the generator. Each intermediate wire has exactly **m** of these streams flowing through it. Again, the wires are evenly divided among the chains. However, since each chain uses only a part of the wire's capacity, the wires can be used by other chains even from other generators as well.

Let **p** =mod(n-r, r)

- For the first **p** VCC chains, the number of intermediate ports NW is equal to the quotient of (n-r)/r plus 1, i.e., $\lfloor (n-r)/r \rfloor + 1$
- For the remaining **(r-p)** VCC chains, NW is equal to the quotient of (n-r)/r, or $\lfloor (n-r)/r \rfloor$
- For all **m** VCC chains, input and output ports may be selected from any of the switch ports Px not selected by other VCC chains.

After applying the first three steps of the methodology we obtain the configuration shown in Figure B.9 for the case of 8-to-2 partial cross with 2 generators.
Note that in this case we have exchanged the number between wires W5 and W6. This is done because the output of previous wire W6, P3 coincided with the input of wire 1. So, going from W6 to W1 would have required a loopback on P3.

In this case, p =mod(8-2,2) = 0. So, the VCC chains of both generators have $\lfloor (8-2)/2 \rfloor = 3$ intermediate wires.

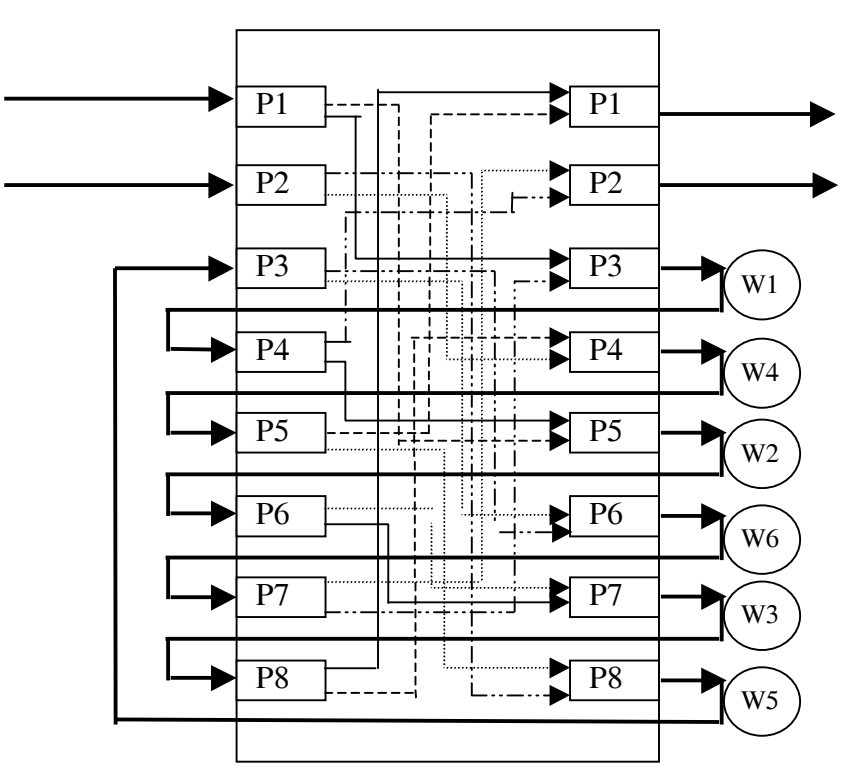


Figure B.9 Implementation of 8-to-2 partial cross configuration with 2 generators for foreground traffic

Both of the VCC chains of the first generator start and end at port P1, so:
CHin(1,1) = CHout(1,1) = CHin(2,1) = CHout(2,1) = P1.

Similarly for the two VCC chains of the other generator:
CHin(1,2) = CHout(1,2) = CHin(2,2) = CHout(2,2) = P2.

First we divide the wires among the two generators. The first generator gets W1, W2, and W3. The second generator gets W4, W5, and W6.

The first chain of the first generator is simply P1-W1-W2-W3-P1. The first chain of the second generator is P2-W4-W5-W6-P2.

The second chain from the first generator is obtained by shifting the intermediate ports of the first chain. Therefore, the chain is P1-W2-W3-W4-P1. Note that this chain is sharing wire W6 of the other generator since each chain uses only half the capacity.

The second chain of the second generator is again obtained by shifting: P2-W5-W6-W1-P2.

b)Latency measurements: Again we consider only the case of background traffic sharing the foreground ports in the opposite direction. Excluding the foreground port, the remaining n-1-r ports connected among themselves and their wires are evenly divided among the r generators.

Let **p** =mod(n-r-1, r)

- For all VCCs of the first **p** generators NW is equal to the quotient of (n-r)/r plus 1, i.e., $\lfloor (n-r)/r \rfloor + 1$
- For all VCCs of the remaining **(r-p)** generators, NW is equal to the quotient of (n-r)/r, or $\lfloor (n-r)/r \rfloor$
- For all **m** VCCs of only one generator, the input and output ports coincide with the output and input ports of the foreground traffic, respectively.
- For all **m** VCCs of all other generators, the input and output ports can be selected from any of the switch ports P_x not selected by other generators.

An example of this case is shown in Figure B.10. In this case, n=8, r=2. This gives p=mod(8-2-1,2) = 1. Therefore, NW(1)=3 and NW(2)=2.

The VCC chains of the first generator uses ports P1 and P2 in opposite directions of the foreground traffic. The VCC chains of the second generator will use port P3 as the source and destination.

The chains of the first generator are: P1-W1-W2-W3-P2 and P1-W2-W3-W4-P2.

The chains of the second generator are: P3-W4-W5-P3, P3-W5-W1-P3.

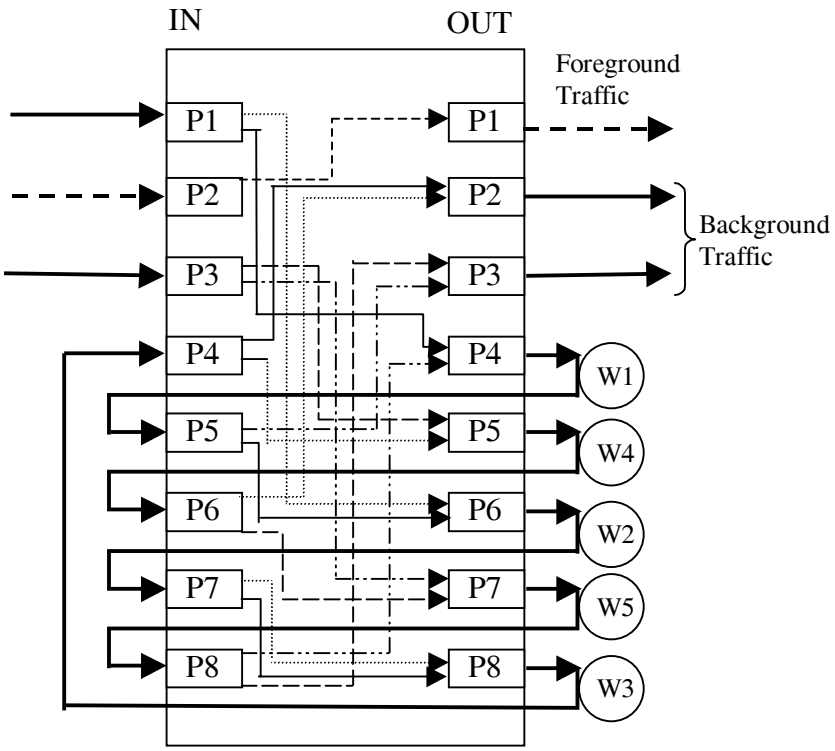


Figure B.10 Implementation of 7-to-2 partial cross configuration with 2 generators for background traffic in latency measurements.

Table B.1 summarizes the values for number of intermediate ports in various configurations of this section **B.3**. These values are used in the pseudocode of Section **B.4**.

	n-to-n straight (Single generator)	**n-to-n straight (r Generators)**	**n-to-m Partial Cross (Single Generator)**	**n-to-m Partial Cross (r Generators)**
Number of Intermediate wires NW	**See B.2.1.** **a) n-1** **b) n-2** **c) n-3**	**See B.2.2.** **$\lfloor (n-r)/r \rfloor + 1$** **or $\lfloor (n-r)/r \rfloor$**	**See B.2.3.** **a) n-1** **b) n-2**	**See B.3.3.** **$\lfloor (n-r)/r \rfloor + 1$** **or $\lfloor (n-r)/r \rfloor$**

Table B.1 Parameter values used in the algorithm to creating VCC chains for different configurations.

B 4. Internal Connection Algorithm for creating VCC Chains.

The following algorithm can be used to create VCC chains for different connection configurations and is based on the definitions given in section **B.2.** and the characteristics specified in section **B.3** and summarized in Table **B.9**.

- **NW(k)** denotes the number of intermediate wires for the VCC chains of the k^{th} generator. These values are specified in **B.2**
- **TNW** denotes the total number of wires.
- **W(f)** denotes the f^{th} wire
- **CH(i, j, k)** denotes the i^{th} intermediate wire of the j^{th} VCC chain of the k^{th} generator
- The function mod*(x, n) is equal to mod(x, n) except for the cases where mod(x, n) is equal to zero, where the function is equal to n

f = 1;
for (k = 1 to r, step 1)
{
if(k>1)

$$f = 1 + \sum_{d=1}^{k-1} NW(d)$$

for (j = 1 to m, step 1)

{ if(j>1) f = mod*(CH(1,j-1,k)+1, TNW);

for (i =1 to NW(k), step 1)
{

CH(i,j,k)=W(f);

f = mod*(f+1, TNW);

} end for i

} end for j

} end for k.

The following changes to Sections 3.1.7 and 3.2.7 of the Performance Testing Baseline Text are necessary as consequences of the acceptance of the new Appendix B, described above.

3.1.7. Guidelines For Scaleable Test Configurations

It is obvious that testing larger systems, e.g., switches with larger number of ports, could require very extensive (and expensive) measurement equipment. Hence, we introduce scaleable test configurations for throughput measurements that require only one ATM monitor with one generator/analyzer pair. Figure 3.3 presents a simple test configuration for an ATM switch with eight ports in a 8-to-8 straight connection configuration. Figure 3.4 presents a test configuration with the same switch in an 8-to-2 partial cross connection configuration. The former configuration emulates 8 foreground VCCs, while the later emulates 16 foreground VCCs.

In both test configurations, there is one link between the ATM monitor and the switch. The other seven ports~~have external loopbacks. A loopback on a given port causes the frames transmitted over the output of the port to be received by the input of the same port.~~

are externally connected. The output of one port is connected to the input of another port through a wire or fiber indicated as Wx, where x is an index. The test configurations in Figure 3.3 and Figure 3.4 assume two network modules in the switch, with ~~switch ports P0–P3~~ports P1, P3, P5, P7 in one network module and switch ports ~~P4–P7~~P2, P4, P6, P8 in the another network module. Foreground VCCs are ~~always~~preferably established from a port in one network module to a port in the another network module. These connection configurations could be more demanding on the SUT than the cases where each VCC uses ports in the same network module. An even more demanding case could be when foreground VCCs use different fabrics of a multi-fabric switch.

Approaches similar to those in Figure 3.3 and Figure 3.4 can be used for n-to-(n–1) full cross and other types of n-to-m partial cross connection configurations, as well as for larger switches. For details, see Appendix B. Guidelines to set up scaleable test configurations for the k-to-1 connection configuration are under study.

It should be noted that in the proposed test configurations, because of ~~loopbacks,~~external connections, only permanent VCCs or VPCs can be established.

It should also be realized that in the test configurations with ~~loopbacks,~~external connections, if all link rates are not identical, it is not possible to generate foreground traffic equal to the MFL. The maximum foreground traffic load for a n-port switch in those cases equals $n \times$ lowest link rate. Only in the case when all link rates are identical is it possible to obtain MFL level. If all link rates are not identical, and the MFL level needs to be reached, it is necessary to have more than one analyzer/generator pair.

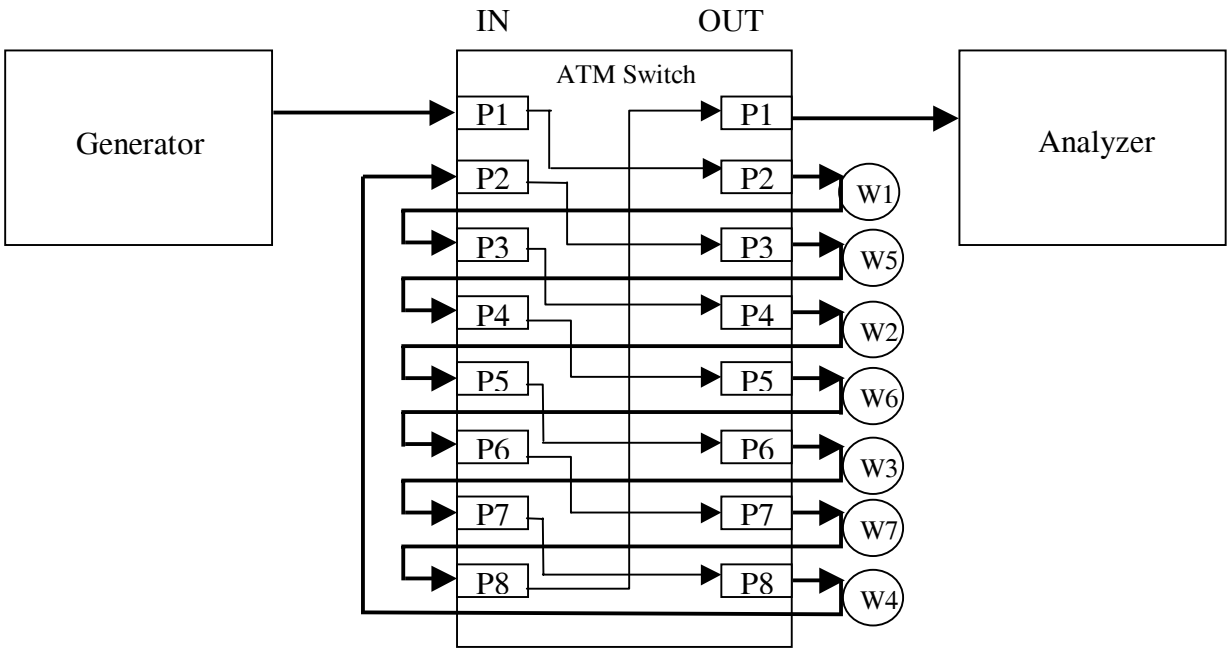


Figure 3.3: A scaleable test configuration for throughput measurements using only one generator/analyzer pair with 8-port switch and 8-to8 straight connection configuration.

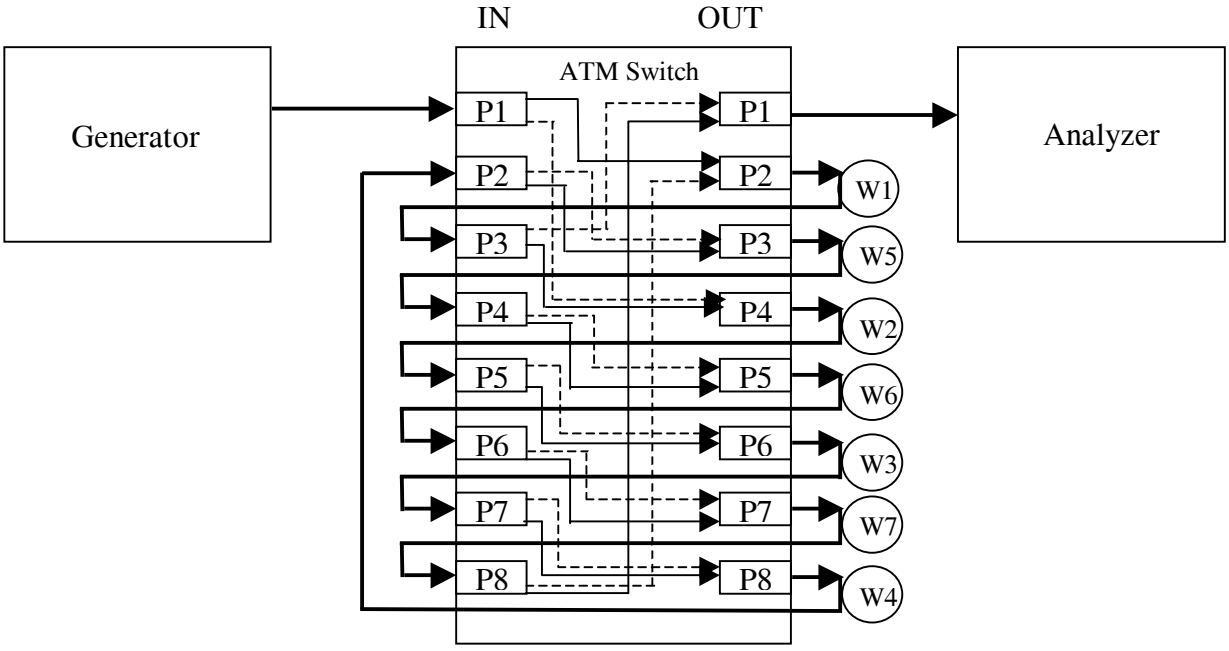


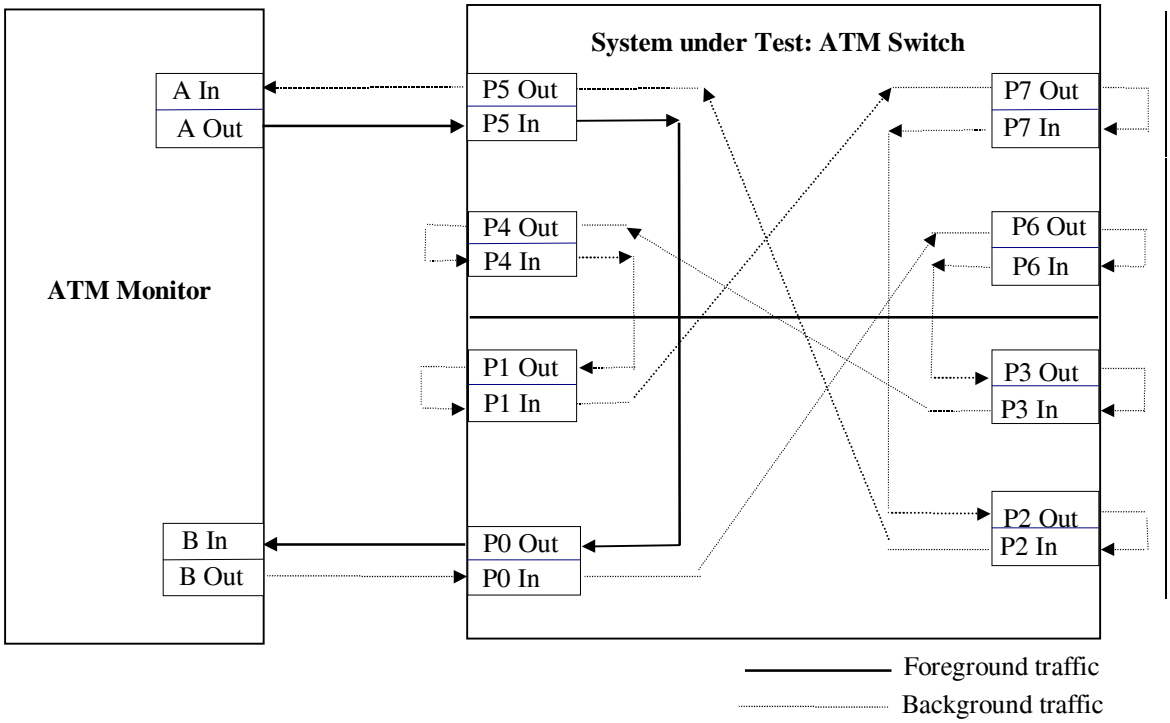
Figure 3.4: A scaleable test configuration for throughput measurements using only one generator/analyzer pairs with 8-port switch and 8-to-2 straight connection configuration.

3.2.7. Guidelines For Scaleable Test Configurations

Scaleable test configurations for MIMO latency measurements require only one ATM test system with two generator/analyzer pairs. Figure 3.5 presents the test configuration with an ATM switch with eight ports ~~(*w*=8).~~(n=8). There are two links between the ATM monitor and the switch, and they are used in one direction by the background traffic and in the another direction by the foreground traffic, as indicated. The other six ~~(*w*–2)~~(n–2) ports of the switch are used only by the background traffic and they have ~~external loopbacks. A loopback on a given port causes the frames transmitted over the output of the port to be received by the input of the same port.~~
been connected externally among them. An external connection is realized between the output of one port to the input of another port by a wire or a fiber Wx .

Figure 3.5 shows a 7-to-7 straight connection configuration for the background traffic. The ~~n to (n–1)~~w-to-(w–1) full cross configuration and the ~~n to m~~w-to-m partial cross configurations can also be similarly implemented. Recall that w =n-1.

The test configuration shown assumes two network modules in the switch with ports ~~P0–P3~~P1, P3, P5, P7 in one network module and ports ~~P4–P7~~P2, P4, P6, P8 in the another network module. Here, the foreground VCC and background VCCs are established between ports in different network modules.
~~It should be noted that in the proposed test configurations, because of loopbacks, only permanent V~~CCs or VPCs can be established.
It should also be realized that in test configurations, if all link rates are not identical, it is not possible to generate background traffic (without losses) equal to MBL. The maximum background traffic input rate in those cases equals (*n*–1) × lowest link rate. Only in the case where all link rates are identical is it possible to obtain MBL level without losses in background traffic.



~~**Figure 3.5**: A scaleable test configuration for measurements of MIMO latency using only two generator analyzer pairs with 8-port switch and 7-to7 straight configuration for background traffic~~

~~If the link rates are different, it is possible to obtain MBL in the n-to-n~~ If the link rates are different, it is possible to obtain MBL in the w-to-w straight case, but background traffic will have losses. In this case, the foreground traffic should use the lowest rate port in the switch as the input, while the highest rate port in the switch should be used as the output. The background traffic enters the SUT through the highest rate port and passes successively through ports of decreasing speeds. At the end, the background traffic exits the switch through the lowest rate port.

The scalable test configuration construction is treated in general and more details in Appendix B.

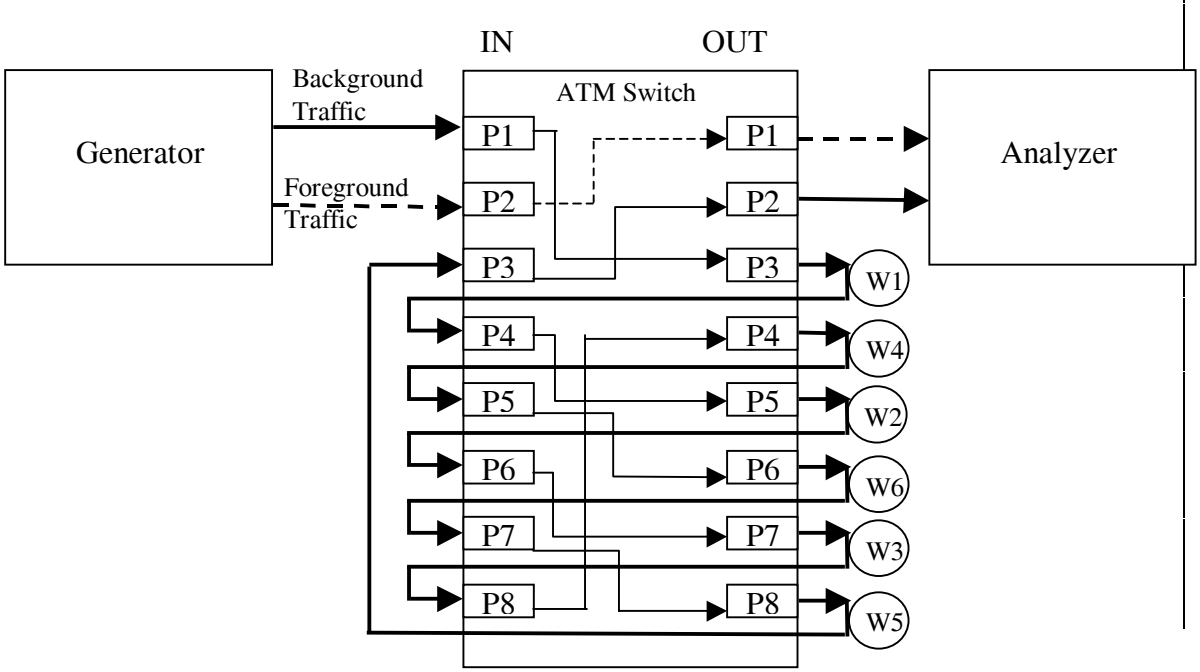


Figure 3.5: A scaleable test configuration for measurements of MIMO latency using only two generator analyzer pairs with 8-port switch and 7-to7 straight configuration for background traffic