



# How Many Channels?

## Part Two: The Effects of FICON on Channel and Device Performance

By Tom Aurand

In a previous article, we compared the properties of FICON with ESCON and examined the general effects on I/O subsystem performance. In this article, we will formulate a method for designing an optimized FICON implementation based on measurements taken from the existing ESCON configuration.

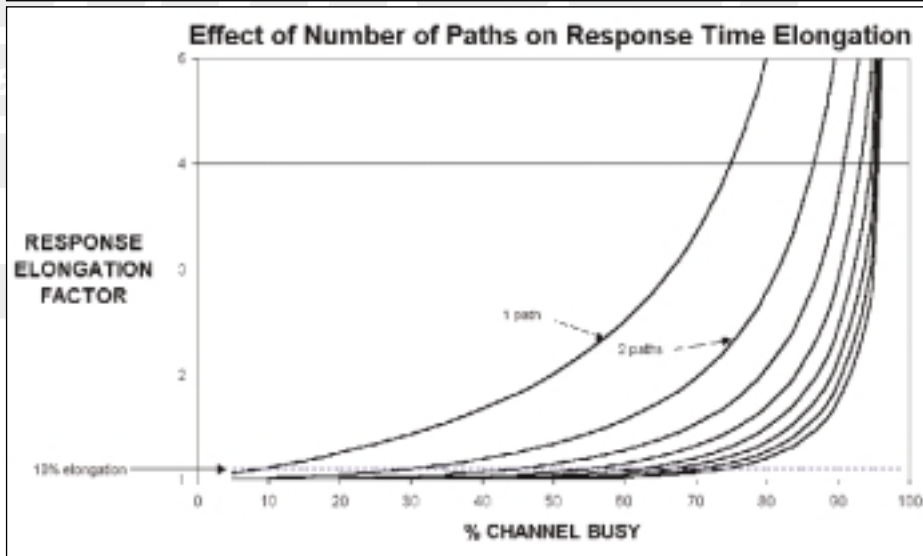
### A QT MODEL

The study of how systems of “servers” servicing “requests” behave is called Queuing Theory (QT). QT can be used to describe the behavior of a wide variety of phenomena, such as how response time is influenced by channel path utilization, how CPU wait time is accumulated, how long a job waits for a tape drive, how many batch initiators are needed, how online transaction queues work, how long I wait in the checkout line at the supermarket, how long it takes to get dial tone when I pick up the telephone, etc. QT might just as well be called the Science of Waiting. Although QT is quite complex mathematically, a basic understanding of the results is often sufficient to take the mystery out of the behavior of queuing systems, and can be used to give quite good predictions.

One remarkable result of QT is the fact that elongation of response time depends not just on percent busy, but also on the number of servers available. At a given percent busy, response time elongation decreases with more servers.

FIGURE 1 shows the QT predicted response time elongation as a function of percent busy, for various numbers of servers. Each curve represents the behavior associated with a particular number of servers. An elongation factor of 2 indicates that response time doubles from the base response (that experienced at zero percent busy), a factor of 3 indicates response

FIGURE 1: RESPONSE ELONGATION OF MULTIPLE-SERVER QUEUING SYSTEMS AS PREDICTED BY QUEUING THEORY



is tripled, etc. The reference line is at 10% elongation (factor of 1.1).

As shown by the chart, as servers are added, the percent busy can be increased while maintaining the same response elongation. One conclusion that can be drawn from this chart is that your 8-way processor can be run to 75% busy before significant response elongation occurs, while a uniprocessor experiences elongation at 10% busy! This chart is so important to understanding performance of so many aspects of data processing that it is worth printing out

and hanging on the wall for capacity planners and performance managers.

### RESPONSE TIME AND PATH BUSY

In designing an I/O configuration, one must consider not just connectivity, but also path busy. In the case of FICON implementation, reducing the number of channel paths moves you onto different response elongation curves, as indicated in FIGURE 1. This is why it is important to consider not just percent busy,

but also which of these curves describes the resulting system.

There is a wide use of rule of thumb (ROT) as applied to channel path busy, typically thought to be 30% busy as a target number to ensure adequate performance. However, as with most ROTs, individual results may vary. This is particularly true when applying ROT to channel path busy. The reason is that the number you use to measure path busy from RMF may not be your actual path busy! Why? Because RMF only views path busy from the processor side, and does not necessarily indicate what is happening between the director and the control unit. Consider the configuration in FIGURE 2. Here we have a control unit connected to two processors through a director. RMF measures 45% path busy on processor 1, and 35% path busy on processor 2. However, the path busy between the director and the control unit is 80%! RMF does not measure this directly, but it is the *effective path busy* to both processors. Delays associated with this outboard path busy are manifest as port busy delays.

This simple example illustrates the need to consider the entire path when designing a configuration, not just the processor to director connections. The percent busy used in applying the QT model of FIGURE 1 is the *highest busy* of any component of the path. So in the example of FIGURE 2, this system is operating on the fourth curve (4 paths) at 80% busy, so we can expect roughly 60% increase in response time, not the minimal increase expected at 35 and 45 percent. It is this effect of combined path busy on the back-end in multiple processor configurations that probably accounts for the low ROT of 30%.

Even more obscure is the configuration in FIGURE 3. Here we have two control units “logically daisy-chained” through a director. RMF measures the path busy at 35%, but what is the back-end path busy? We can determine this under ESCON by adding up all the device connect time on each control unit, taking this as a percent of the time interval, and dividing the result by the number of paths. Unfortunately, you cannot apply the same method to a FICON configuration, because FICON connect time includes frame-pacing delay, which does not contribute to path busy. But even without knowing the back-end path busy, we know that it cannot be higher than the front-end, so this system is operating on the 4-path curve at 35% busy, and we have negligible response elongation. This configuration vastly outperforms the configuration of

FIGURE 2: EXAMPLE OF BACK-END PATH BUSY IN A MULTIPLE-PROCESSOR CONFIGURATION

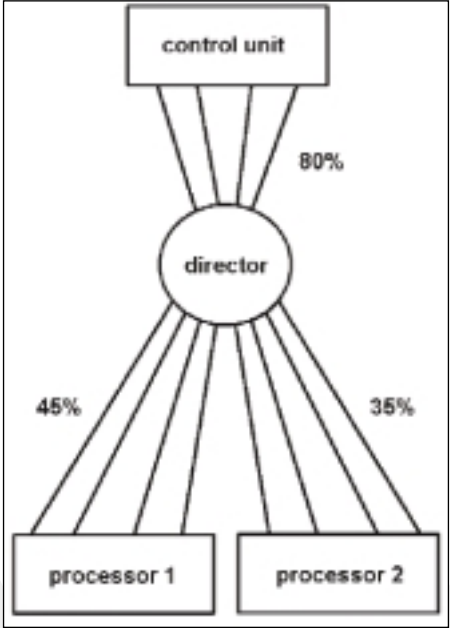


FIGURE 2, even though the RMF measured path busy is similar. This is why application of ROT regardless of the configuration is not a good idea.

To complicate matters further, a typical system is both a multiple processor and daisy-chained configuration. So, how do we know how to characterize the performance? Consider the configuration in FIGURE 4. Here, we know the front-end path busy through RMF measurements. We can determine the back-end path busy for each control unit through the method of addition of device connect time. As a check, the total connect time on both the front-end and back-end must be the same (under ESCON). So, in the final analysis, take the percent busy for each control unit on the back end, multiply by the number of paths from that control unit to the director, add these up. Then do the same on the front-end, using the RMF measured path busy times the number of paths from each processor. The answers must agree.

Once the percent path busy has been determined for each group of back-end and front-end paths, we can determine the response curve on which a given device operates, as viewed by a particular processor. To do so, we simply determine the part of the configuration that yields the greatest response elongation. So, in the configuration of FIGURE 4, devices on control unit 1 operate on the 8-path curve at 20% from processor 1, and at 40% from processor 2, while devices on control unit 2

FIGURE 3: A “DAISY-CHAINED” CONFIGURATION

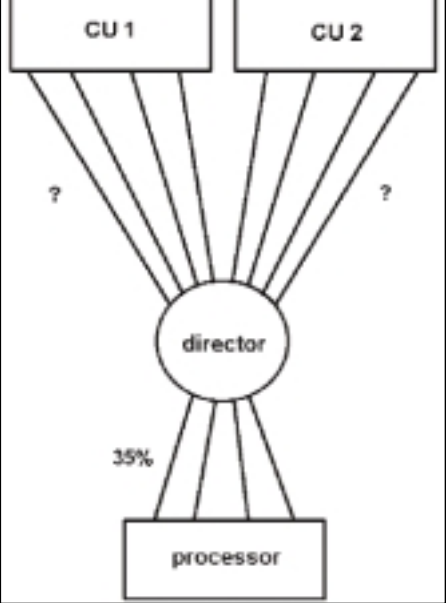
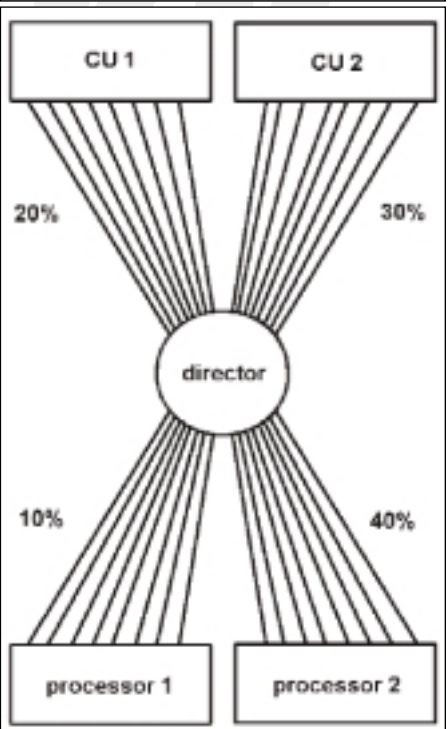


FIGURE 4: A TYPICAL CONFIGURATION IS BOTH DAISY-CHAINED AND MULTIPLE-PROCESSOR



operate at 30% from processor 1 and 40% from processor 2.

**BUT WHAT ABOUT FICON?**

The preceding discussion on calculating path busy to determine the response delay applies in principle to FICON configurations as well. However, we must consider reads

and writes separately, because reads will experience delays based on read path busy, while write delays are due to write path busy. Therefore, to evaluate the performance under FICON, we must know the read/write ratio. This can be estimated from dataset activity measurements, by multiplying block size by blocks transferred for both reads and writes, and taking the ratio. Also, remember that there may be ESCON channel “dead” time that will not carry forward into the FICON environment. So, instead of just applying the read/write ratio and adjusting to the FICON data rate, we first must determine how much actual data transfer time there is. This is taken from the same numbers used to calculate the read/write ratio; it is the sum of the byte counts, divided by the effective data rate. This is likely to be less than the ESCON connect time.

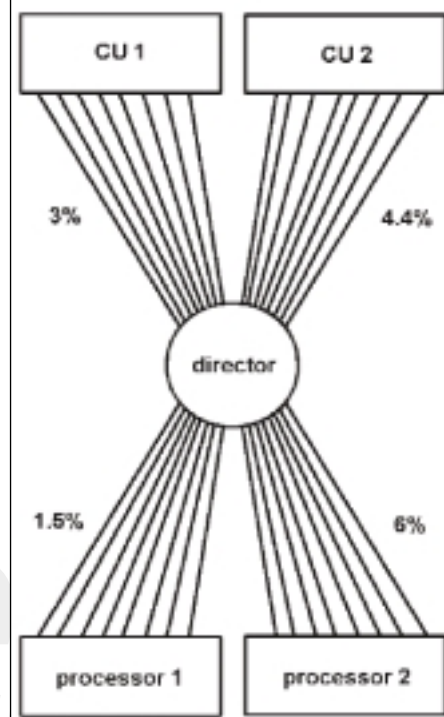
To further complicate matters, FICON introduces a new overhead not experienced under ESCON that contributes to channel path busy. This overhead has been reported anecdotally as 10%-13% path busy, even when there is no true I/O activity. These numbers are being generally passed around the industry as relating to some mysterious fixed overhead. However, I have seen FICON channels on some systems with no I/O activity showing measurements well below these numbers, in the 5% to 6% range. So it would seem that this is not a fixed overhead, but rather influenced by the configuration. I suspect it relates to the number of devices online to the channel at the time. However, to date I do not have sufficient measurement data in a wide variety of system configurations to find a good predictive correlation. At present, we can do no better than to assume that this overhead will be in the 10% to 13% range in a typical production environment.

## A CASE STUDY

Ok, now it’s time to apply all this new understanding to the real world. Suppose you have an ESCON configuration that you want to convert to FICON. You want to take advantage of the increased data rate, the improved protocol, and the reduction of channels, cables, ports and channel adapters. So, you want to answer the following questions:

- How many channel paths do I need?
- What will my response time be?
- Can I take advantage of the improved bandwidth to roll-up control units?

FIGURE 5: PROJECTED FICON PATH BUSY CALCULATED FROM ESCON PATH BUSY

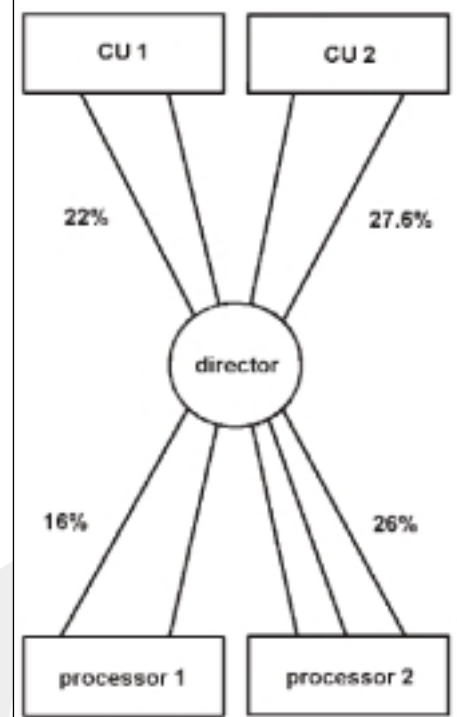


### The 1:1 swap-out

Consider the configuration in FIGURE 4. Perhaps the simplest way to approach the problem is to first determine what would happen if I just convert all ESCON to FICON, in a 1:1 swap-out. In this scenario, you need not worry about device response time, because it’s certain to improve. The objective here is just to determine the FICON path busy in all components of the configuration. To do so, I first adjust the ESCON path busy by removing the “dead” time. This number is difficult to determine, requiring adding up all the bytes transferred using data set level activity information, but for high cache-hit systems is usually less than 5%. In any case, it is “safe” to assume that there is no ESCON dead time. This assumption will lead to a worst-case calculation ensuring that you have at least enough FICON channels in the end. If you can determine the true dead time, simply reduce the ESCON path busy numbers by this amount before proceeding.

Taking the dead-time reduced ESCON numbers, we next apply the read/write ratio to split into read busy and write busy times. Once again, the read/write ratio is not a simple number to calculate (and can vary greatly from time to time and by subsystem), but you may be able to estimate based on general knowledge of the data profiles in your environment. It is “safer” to assume a greater disparity between reads and writes; 1:1 read/write ratio is the

FIGURE 6: OPTIMIZED CONFIGURATION UNDER FICON



ultimate balance, while 1:0 and 0:1 (all reads or all writes) give the “worst-case” in which you are using only half the capacity of the FICON fibers. Perhaps a compromise between these is to use a read/write ratio of 2:1. Most real-life shops can achieve this good a balance.

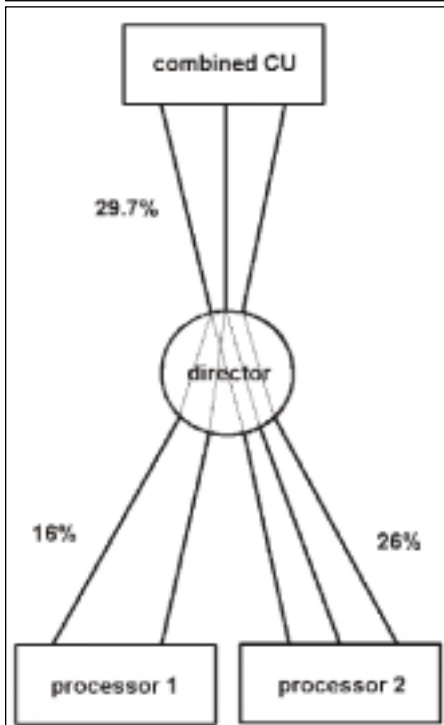
Whatever read/write ratio we choose to use, we need consider only the higher side of the equation; i.e. the number of paths we need will be determined by reads if they outnumber writes, or (unusually) writes that outnumber reads.

FIGURE 5 shows the projected FICON path busy (without overhead) as calculated from the configuration in FIGURE 4, under the assumptions that there is no ESCON channel dead time, and a read/write ratio of 2:1. Only the channel busy for reads is shown, since reads outweigh writes at 2:1. The new numbers are calculated from the ESCON numbers by applying the 2:1 ratio (multiply by 2/3), and then applying the difference in transfer rates (divide by 4.5).

### The many:1 consolidation

Now we are ready to determine how many cables we need to support I/O in each part of the configuration. To do this, we just take away one path at a time and see where we are on the chart in FIGURE 1. The goal is to reduce the number of paths as much as possible and still be below the reference line at 10% response

FIGURE 7: CONTROL UNITS COMBINED UNDER FICON



elongation. However, we must always maintain at least two paths in any group, to satisfy service continuity requirements.

Since our original configuration is eight paths, we can find the effective path busy by increasing the path busy in a ratio to eight paths. For example, confining our attention for the moment to the paths from processor 1 to the director, if we reduce the number of paths to seven, we find the new path busy to be  $1.5\% \times 8 / 7 = 1.7\%$ , then adding in the FICON overhead, we have 11.7% effective path busy. This is well within the acceptable operating range.

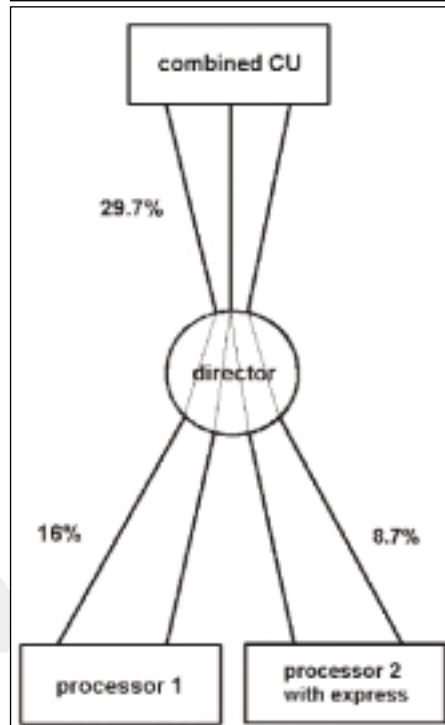
Following the same method, we determine the effective busy for the rest of the possibilities:

- ▼ With six paths effective busy =  $1.5\% \times 8 / 6 + 10 = 12\%$
- ▼ With five paths effective busy =  $1.5\% \times 8 / 5 + 10 = 12.4\%$
- ▼ With four paths, 13%
- ▼ With three paths, 14%
- ▼ With two paths, 16%
- ▼ With one path, 22%

Of these, only the one-path scenario is outside of the acceptable operational range. We therefore conclude that two paths are sufficient, yielding a path reduction of 4:1.

Applying the same method to the rest of the configuration, we find:

FIGURE 8: CONFIGURATION WITH PROCESSOR-SIDE FICON EXPRESS



- ▼ Three paths required from processor 2, at 26% busy
- ▼ Two paths required to control unit 1, at 22% busy
- ▼ Two paths required to control unit 2, at 27.6% busy

FIGURE 6 shows the configuration under FICON. The net result is not quite a 4:1 reduction in the number of paths required.

### Can I roll up control units?

The result of analysis that supports the reduction in the number of paths for an existing configuration converted to FICON leads to the possibility that more data can be consolidated behind a single control unit, without overwhelming the bandwidth to the processor. After applying the methods examined above, we can easily determine if control units can be consolidated. In fact, they certainly can (from the channel path perspective, not withstanding considerations such as device capacity restrictions and cache size), if the total number of channel paths can be combined on one control unit. In our example, we have just four FICON paths in total on the control units, so they certainly can be combined to one control unit with four paths. However, it may be possible to further reduce channel paths coincident with the consolidation. To make this

determination, we just apply the same method as before, removing one path at a time, and see where it puts us on the response chart.

After removing the overhead again, and averaging the path busy, we find each path will be 14.8% busy (without overhead). So, to find the path busy at three paths, we take  $14.8 \times 4 / 3 + 10 = 29.7\%$  busy. With two paths, we find 39.6% busy, which puts us outside of the operational range. We conclude that we can eliminate one more back-end path, and still roll-up the control units. FIGURE 7 shows the final FICON configuration, optimized for cost and performance.

### Should I implement FICON Express?

FICON express offers twice the data transfer rate as standard FICON. Although this further reduces connect time and path busy, the primary benefit is generally the opportunity to reduce paths even more. If your standard FICON configuration is already at the two-path minimum, you cannot reduce paths by using express. Unless you absolutely need the extra reduction in response time by reducing connect time further (not likely at the outset, since FICON performance is already better than ESCON), express simply adds expense to the two-path FICON configuration.

Perhaps the early benefit of express is the potential reduction of channels and processor-side director ports. Express can be implemented on the processor while standard FICON is still on the control unit. This is possible because the director can convert the transfer rate. You cannot, however, mix ESCON and FICON on the same path, because this would require the director to convert the protocol, which is beyond the capability of the director.

As an example of how express can be used to reduce processor-side paths, consider again the standard FICON configuration of FIGURE 7. There may be an opportunity to reduce the channels on Processor 2 through the use of FICON express. To make this determination, we first calculate the path busy under express if we were to simply swap-out the standard FICON channels 1:1 for express. This is very simple: just divide by two. Next we apply the same process as before, removing one path at a time, recalculating the path busy, and determining the expected response elongation using the QT chart.

FIGURE 8 shows the resulting configuration employing FICON express to reduce channels. Processor 1 was already at two paths, so express is not much benefit there. Processor 2 had three paths to the control unit (through the

director), and we determined that express would allow us to reduce the paths to two while maintaining adequate response. In this configuration, the director converts the transfer rate on the paths from processor 2 to the control unit.

The same method can be used to determine possible path reductions on the control unit side (CAs, director ports). In the example of FIGURE 8, we could reduce the three back-end paths to two by implementing express on the control unit. This could be done with or without implementing express on either processor (although this would make one standard FICON channel on Processor 2 unusable). For example, we could have a configuration with two express paths on the back end, two express paths to processor 2, and two standard paths to processor 1. In this case, the director converts the transfer rates from processor 1 to the control unit.

## CONCLUSION

FICON offers tremendous advantages over ESCON in improved bandwidth and performance. Using only a rule-of-thumb approach when planning a FICON implementation can result in missed opportunity in realizing maximum return from these advantages. Armed with some knowledge about how the architecture works, and borrowing a little from queuing theory, the capacity planner and performance manager can devise an implementation plan that is less costly and simpler to implement, and still be confident about adequate performance delivery. By utilizing the methods described here, one can test the validity of claims to high channel path reduction ratios and at the same time avoid the need to implement conservative, "safe," low-reduction ratio plans because of uncertainty of the resulting performance.



NaSPA member Mr. Aurand is founder and President of Advanced Technological Research, Inc., a software and services company marketing

product line Lexonix Technologies, specializing in batch systems tuning and mainframe performance. Mr. Aurand has over 25 years of experience in all aspects of mainframe application and systems management, from application development and programming, to operating system support, capacity planning, and hardware management. Mr. Aurand has an extensive mathematical and statistical background and has developed several modeling systems for statistical analysis of performance of mainframe CPU and peripheral devices.

<sup>1</sup> Strictly speaking, this is only an approximation. However, by using this simplifying approximation, we can use the method of graphical analysis using the QT chart, avoiding the complexity of actual QT formulas, and still arrive at acceptably accurate predictions of response elongation.

# Technical

Supporting Enterprise Networks and Operating Environments

# SUPPORT