Information Management software

Redpaper

Jeffrey Berger
Paolo Bruni

How does the MIDAW facility improve the performance of FICON channels using DB2 and other workloads?

Abstract

The Modified Indirect Data Address Word (MIDAW) facility was introduced in the IBM® z9™ processor to improve FICON® performance, especially when accessing IBM DB2® databases. This facility is a new method of gathering data into and scattering data from discontinuous storage locations during an I/O operation. This IBM Redpaper is mainly intended for industry professionals who are interested in understanding IBM z/Architecture™, and people who want to understand the types of workloads that will provide improved performance.

The MIDAW facility

The MIDAW facility is a modification to a channel programming technique that has existed since the days of the IBM S/360™ operating system. MIDAWs are a new method of gathering data into and scattering data from discontiguous storage locations during an I/O operation. MIDAWs require the IBM System z9™ server and IBM z/OS® 1.7 (or APAR OA10984 with Release 1.6). This Redpaper intends to answer this question of how MIDAWs improve the performance of FICON channels. In addition, we explain which types of applications and data sets will benefit the most from MIDAWs. The performance of any specific workload might vary according to the conditions and hardware configuration of the environment. Because DB2 is among the chief beneficiaries of MIDAWs, this paper refers to DB2 measurements for illustration.

This paper is mainly intended for industry professionals who are interested in understanding z/Architecture. MIDAWs are implemented by the Media Manager (M/M). This paper explains when M/M is used. To take advantage of M/M, users must use data set types, such as Linear Data Sets and Extended Format data sets that enable the system to use M/M. The tuning aspects of the MIDAW facility are that simple: Just make sure that the processor and operating system are at the correct level, and use data set types that enable the system to use M/M.

The use of MIDAWs will not cause the bits to move any faster across the FICON link, but they reduce the number of frames and sequences flowing across the link, which makes the channel more efficient. See Figure 1.

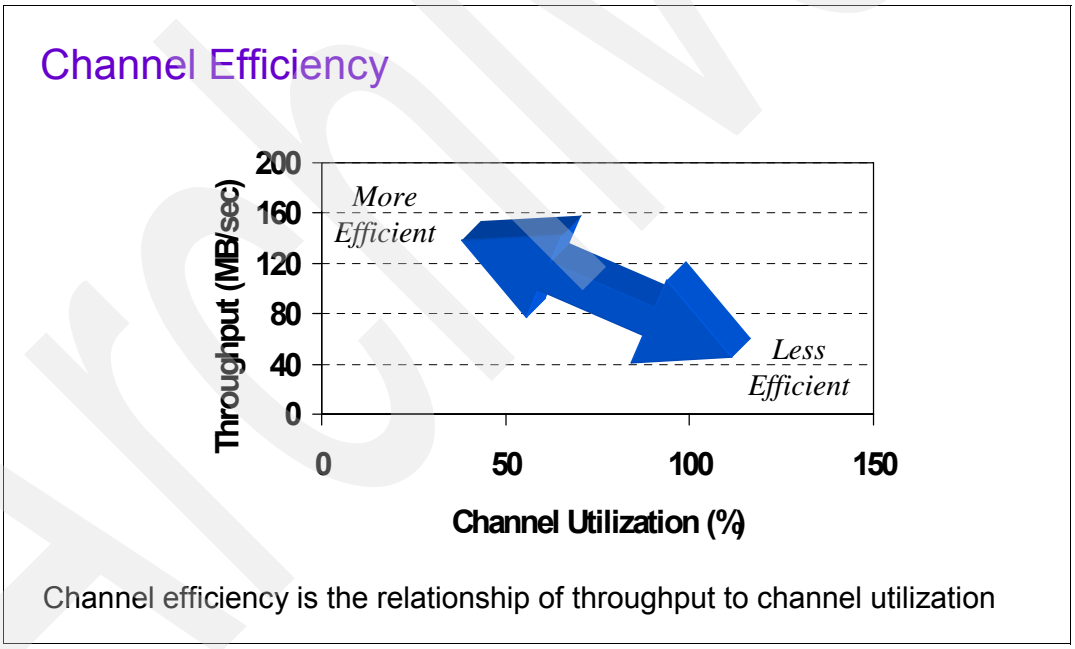

Figure 1 Channel efficiency

Channel efficiency is essentially the relationship of throughput to channel utilization. For a homogenous I/O stream of some uniform set of I/Os, the I/O rate, throughput, and channel utilization are all linearly related to each other. A channel becomes more efficient if the throughput at a particular level of utilization is increased.

Figure 2 on page 3 illustrates a graphical relationship between throughput and channel utilization for a hypothetical workload, with and without MIDAWs.

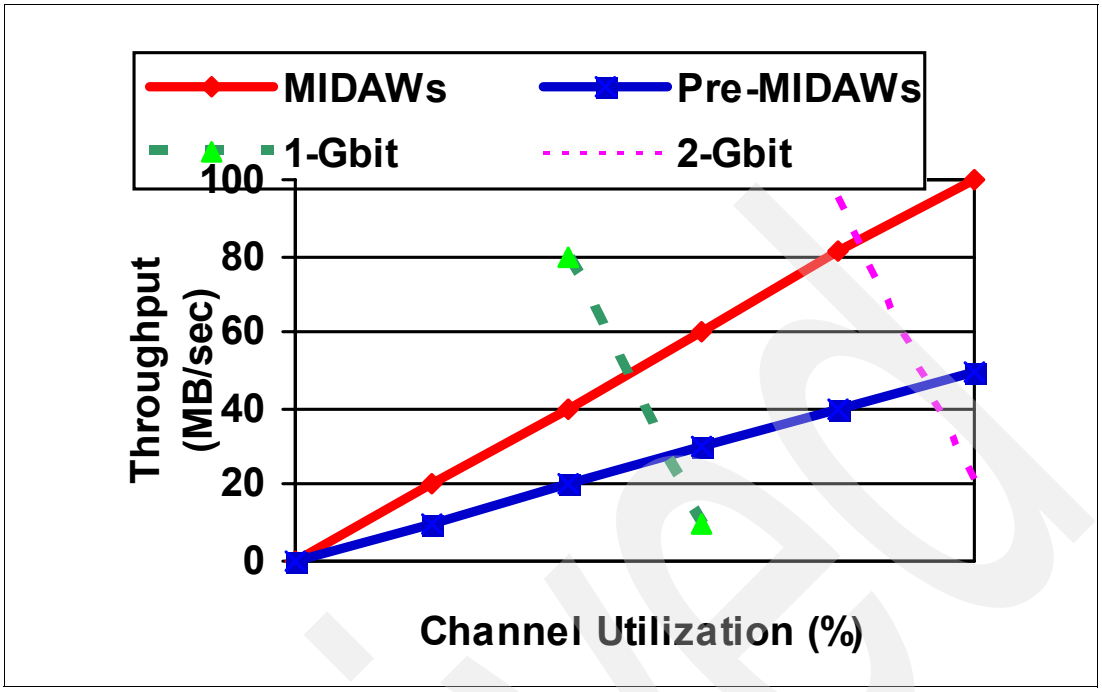

Figure 2 Throughput and channel utilization with MIDAWs

The slope of the MIDAWs line is double the slope of the pre-MIDAWs line, indicating that MIDAWs makes the channel twice as efficient. This graph also illustrates the effect of the link speed on the maximum channel utilization. Because FICON Express2 outperforms a 1 gigabit-per-second channel, the channel utilization was capped at around 52% for this hypothetical workload. A 2 Gbps channel enables the channel utilization to become fully utilized, but prior to MIDAWs the throughput at this level was still limited. MIDAWs then doubled the throughput at all levels of channel utilization in this theoretical example.

In practice I/O operations are not homogenous and it can be difficult to tell from an RMF™ report what types of channel programs are being used. Only a controlled experiment using homogenous I/Os can clearly demonstrate the channel efficiency. Most of the controlled experiments shown in this paper were done using DB2 prefetch I/Os that transfer 128 KB per I/O (in DB2 for z/OS Version 8 and prior releases).

Because the most significant performance benefit of MIDAWs is achieved with Extended Format (EF) data sets, we want to review the purpose of EF data sets. We then review Media Manager and its role in the z/OS operating system. Then we explore some technical aspects of z/OS channel programming that will enable the reader to gain some insight into what the MIDAW facility actually is, and why it improves the performance. In the remaining sections we discuss some DB2 performance measurements that were conducted in a controlled lab environment to illustrate the benefits of MIDAWs.

Extended format data sets

In 1993 IBM introduced extended format (EF) data sets. Both VSAM and non-VSAM (DSORG=PS) can be EF. In the case of non-VSAM data sets, a 32-byte suffix is appended to the end of every physical record (block) on DASD. VSAM appends the suffix to the end of every control interval (CI), which normally corresponds to a physical record. A 32-KB CI is split into two records in order to span tracks. This suffix is used to improve data reliability and facilitates other functions described below. Thus, for example, if the DCB BLKSIZE or VSAM CI size is equal to 8192, the actual block on DASD consists of 8224 bytes. The control unit

itself does not distinguish between suffixes and user data. The suffix is transparent to the access method or database.

Besides reliability, EF data sets enable three new functions: DFSMS striping, access method compression, and extended addressability (EA). DFSMS EA is especially useful for creating large DB2 partitions (DSSIZE > 4 GB). Striping can be used to increase sequential throughput, or to spread random I/Os across multiple logical volumes. DFSMS striping is especially useful for utilizing multiple channels in parallel for one data set. The DB2 logs are often striped to optimize the performance of DB2 sequential inserts.

For reasons that we shall explain later, the FICON channels have not processed the EF suffix efficiently prior to MIDAWs. As the FICON channel speeds have increased, the performance gap between EF and non-EF data sets has grown, as illustrated in Figure 3.

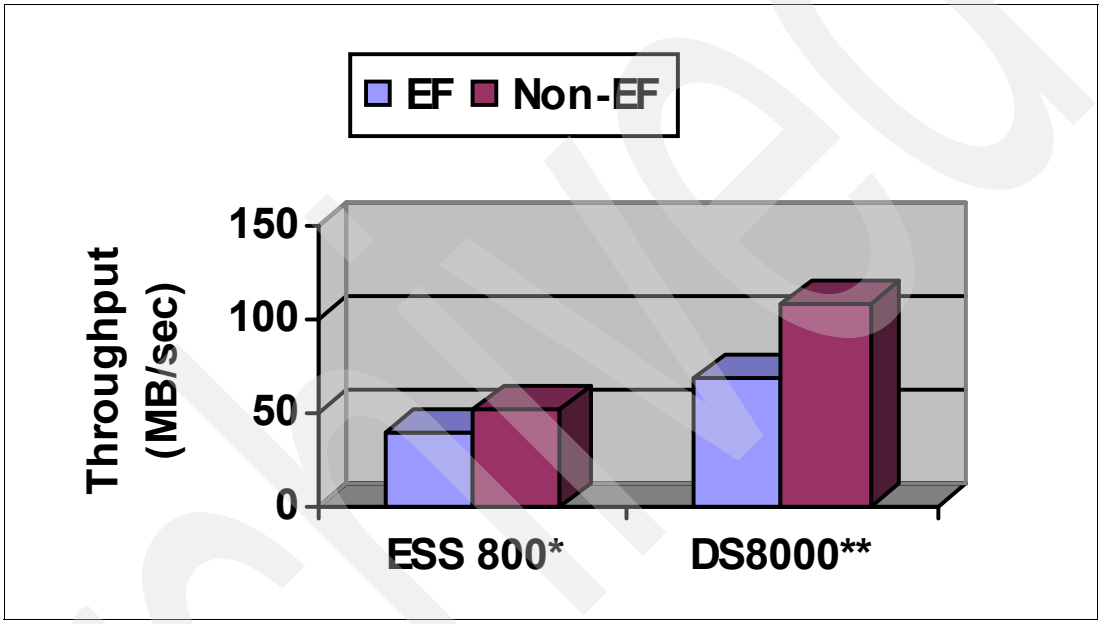

Figure 3 DB2 table scan with 4 KB pages pre-MIDAWs

Figure 3 shows the channel throughput of a DB2 table scan using 4 KB pages. The ESS 800 control unit was attached to FICON Express 1 (2002 technology) and the DS8000™ control unit was attached to FICON Express 2 (2005 technology) and in both cases a 2 Gbps FICON Director was used. The performance gap was 30% in 2002, but had grown to 58% in 2005. The creation of the MIDAW facility was largely motivated by the desire to close this EF performance gap (or EF performance “penalty”), thereby encouraging people to use the features of Extended Format data.

Media Manager

Next, we talk about Media Manager (M/M) and ECKD™ (Extended Count Key Data) architecture. The main objective of M/M is to shield middleware and most operating system components from device dependencies. MIDAWs are an example of why this is beneficial. Users of M/M will automatically benefit from MIDAWs. Nowadays M/M is so pervasive that most I/Os in a computer system are generated by M/M, and there are solutions for most applications that enable the system to use M/M. It would be best if all zSeries® shops avoid data sets where M/M cannot be used.

ECKD is a heritage architecture that z/Architecture inherited from S/360. CKD facilitates variable-length records and keyed records on DASD. M/M requires that data sets have

fixed-length records and no keys. Over the years z/OS (and its legacy systems) has changed to fixed-length non-keyed data sets in order to adapt to M/M requirements. Non-VSAM data sets are a classic example of this. When using Extended Format, the QSAM and BSAM access methods will map the logical variable-length blocks to fixed-length blocks in order to use M/M. All EF data sets, both non-VSAM and VSAM, use M/M. Prior to z/OS 1.3, VSAM non-EF data sets did not use M/M, but now they do. So, when using traditional DSORG=PS data sets it is important to use EF, but with VSAM it no longer matters.

M/M has always been used for linear VSAM data sets, such as those used by DB2, z/FS file systems, and the System Logger used by CICS®. HFS and PDSEs use M/M. M/M is always used for the VTOC index and ICF catalog.

PDSs do not use M/M (because a PDS directory contains keyed records). Consequently it is advantageous to convert all PDSs to PDSEs, including all program libraries. It is also good to convert all SAM data sets to Extended Format. BDAM cannot use M/M or EF data sets. It is better to use BSAM or VSAM in place of BDAM.

When using EF data sets, M/M supplies the storage for the 32-byte suffix and automatically appends the suffix to the user's data. The method that M/M uses to append the suffix, or strip off the suffix when reading a record, is the subject of the next section.

z/Architecture channel programs

In order to understand what MIDAWs are, and why they are important to FICON performance, it is helpful to review the channel programming architecture of zSeries, which mostly dates back to the days of S/360.

An I/O operation is represented by a *channel program*. A channel program consists of a series of CCWs (Channel Command Words) that form a chain. Media Manager uses Format 1 CCWs as illustrated in Figure 4.

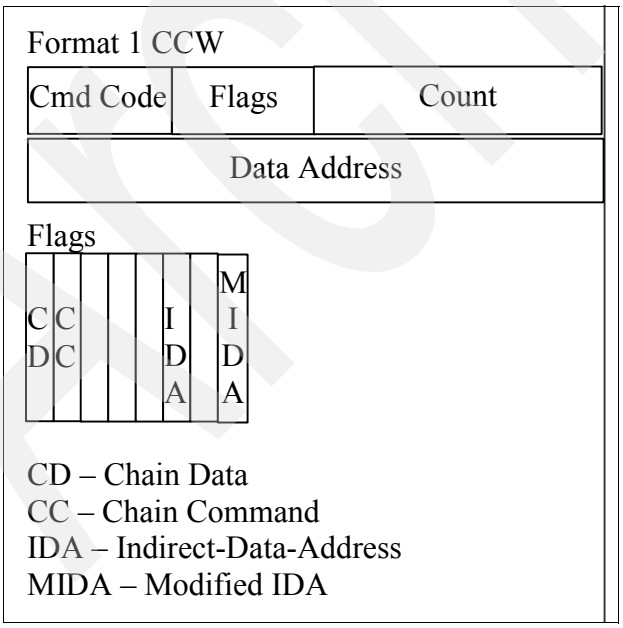

Figure 4 Channel Command Word

The command code specifies to the channel subsystem and the I/O device the operation to be executed, and the count field specifies the number of bytes to transfer. When the channel subsystem has completed the transfer of information specified by a CCW, it can fetch the next

CCW. Such fetching is called chaining, and the CCWs belonging to such a sequence are said to be chained. Two types of chaining are provided: chaining of data and chaining of commands. One flag in the CCW indicates data chaining and one indicates command chaining. The last CCW of a channel program is one where both chaining flags are off.

Unless the IDA flag is set, the data address in the CCW points directly at a continuous segment of real storage. Because ranges of virtual addresses may span discontiguous real 4 KB frames, direct addressing cannot generally be used to address more than 4 KB.

Indirect data addressing

Indirect data addressing (IDA) permits a single CCW to control the transfer of data that spans non-contiguous 4 KB frames in main storage. When the IDA flag is set, the data address in the CCW points to a list of words (IDAWs), each of which contains an address designating a data area within real storage. See Figure 5.

Prior to MIDAWs, Media Manager used Format 2 IDAWs, which enabled it to use data addresses above 2 GB (real addresses). Format 2 IDAWs designate 4 KB chunks; so there is one IDAW for every 4 KB. The number of IDAWs is determined by however many IDAWs are needed to satisfy the count field in the CCW.

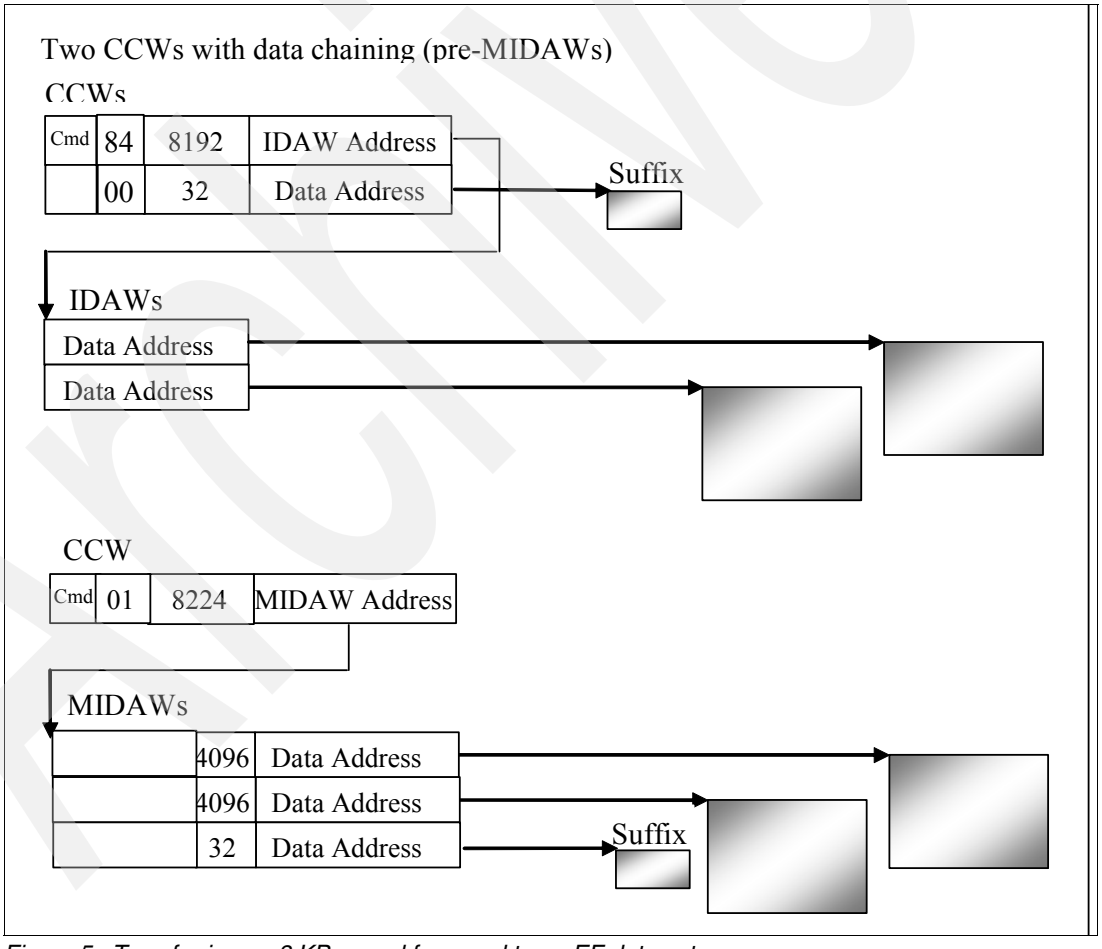

Figure 5 Transferring an 8 KB record from and to an EF data set

The first IDAW can designate any location, but all subsequent IDAWs must point at 4 KB boundaries. (Eliminating this restriction is an essential feature of MIDAWs.)

Control of IDAWs

MIDAWS data transfer is passed to the next IDAW when a 4 KB boundary is reached, but the CCW is considered complete when the number of bytes transferred is equal to the count field specified in the CCW.

For example, suppose that a CCW requests to read X'2000' (8 KB) of data and suppose the first IDAW contains X'80F00'. There must be two subsequent IDAWs on 4 KB boundaries, such as X'3A000' and X'E5000'. The channel will then read the first X'100' bytes into storage starting at location X'80F00', then read the next 4 KB into storage at location X'3A000', and then read the last X'F00' bytes into storage at location X'E5000'.

Figure 5 on page 6 illustrates how data chaining is used to transfer an 8 KB record of an EF data set on a pre-MIDAW system, assuming that the record is on a 4 KB frame boundary. Notice there are two CCWs, and the “chain data” flag (x'80') is on in the first CCW. The first CCW contains a count of 8192 and points at an IDAW list, which designates the two 4 KB frames of the data buffer. The second data-chained CCW contains a count of 32 and points directly at the suffix (which is not in 64-bit storage).

The MIDAW facility is a conceptually simple modification to IDAWs. MIDAWs remove the 4 KB boundary restrictions of IDAWs. A MIDAW is not just an 8-byte pointer; rather, it is 16 bytes, including an 8-byte pointer and a 2-byte count field. The last bit in the flag byte of the CCW indicates that the data address points at a MIDAW list. The sum of all MIDAW counts must equal the CCW count. See Figure 5 on page 6 for an illustration of MIDAWs to transfer an 8 KB record of an EF data set.

Record-level versus track-level channel commands

Channels communicate with a device over a channel *link* or *path*. Output data is received by the device and packaged into *records*, which are grouped into *tracks*. The original S/360 architecture defined “record-level” command operations, not track level operations. Although z/Architecture allows partial records to be read, the CCW count field must match the record length when updating a record.

In 1999 z/Architecture quietly introduced the concept of *track-level* command operations in conjunction with the 2105 control unit model (that is, Shark or ESS). Media Manager implemented track-level commands. When using a track-level command to read data, the DASD control unit will allow a single CCW to read multiple records. When using a track-level command to update data, the DASD control unit will allow a single CCW to update multiple records that reside on the same track, at the same time prohibiting a partial record update.

Track-level operations are beneficial because they potentially reduce the number of CCWs. As the data transfer speeds increase, the channel engine overhead of each CCW represents a larger impact. However, at first these track-level operations did not achieve what they were intended to achieve—at least not with FICON channels—because the command chaining within a track had to be replaced by data chaining. IDAWs are not applicable due to the 4 KB boundary restrictions of IDAWs.

For example, to read twelve 4 KB records from a non-EF data set on a 3390 track, Media Manager chained 12 CCWs together using data chaining. To read twelve 4 KB records from an EF data set, the Media Manager data-chained 24 CCWs together (two CCWs per 4 KB record). Using MIDAWs, M/M can transfer a whole track using a single CCW. To span multiple tracks in one channel program, command chaining was used, and still is used. However, with the advent of MIDAWs, M/M no longer uses data chaining.

If track-level CCWs were the foundation for improved FICON performance, MIDAWs represent the building that rests upon the foundation. In other words, the value of track-level CCW operations was severely constrained prior to MIDAWs.

MIDAWs improve the performance

MIDAWs and IDAWs have basically the same performance characteristics; both perform better than data chaining. There are two reasons. One reason pertains to the number of CCWs used, which in turn affects the number of frames sent across the link. By reducing the number of frames, it takes less time for the FICON channel and the control unit host adapter port to process the channel program. The second reason pertains to FICON multiplexing and the fact that MIDAWs removes some of the data-chaining activity on the link. Multiplexing only affects performance when there are concurrent I/Os on the channel.

The best way to understand these reasons is to look at some data. Figure 6 and all subsequent charts describe measurements using a FICON Express2 channel, a DS8000 control unit, and a 2 Gbps FICON link. Figure 6 shows the I/O response times of DB2 prefetch I/Os with 4 KB pages, with and without MIDAWs, for both EF and non-EF data sets. Prior to MIDAWs the response time for EF was 1.9 ms versus 1.2 ms for non-EF data sets. MIDAWs did not improve the response time for non-EF data sets, but MIDAWs did lower the response time of EF data sets such that EF and non-EF data sets performed identical to each other. The response time of non-EF data sets remained at 1.2 ms.

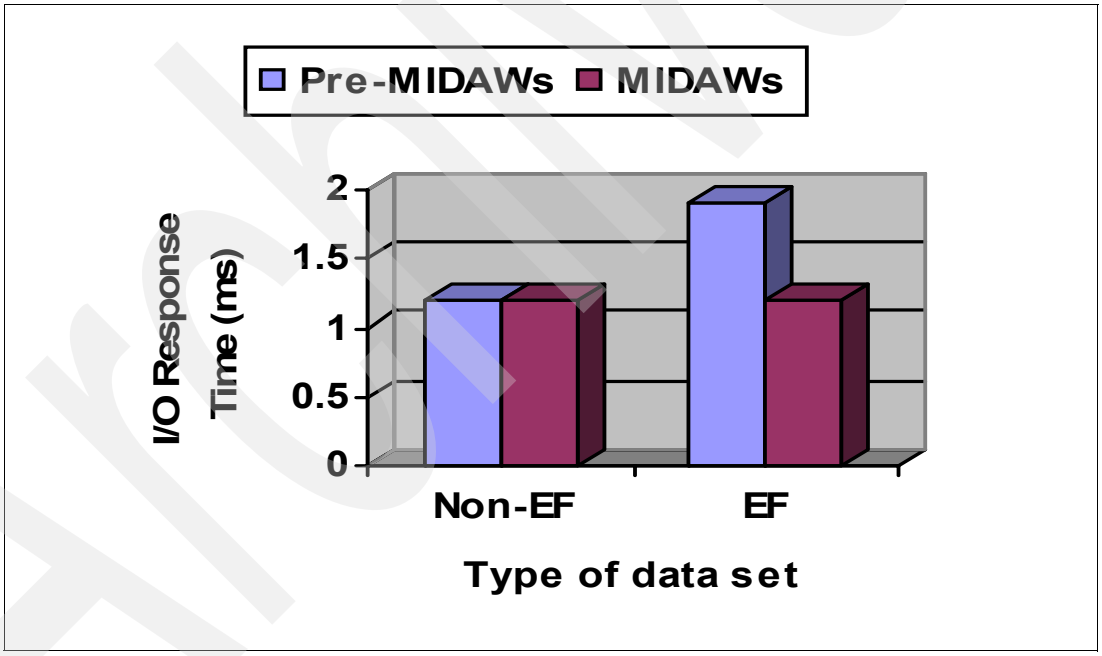

Figure 6 DB2 prefetch I/O response time

MIDAWs reduce the number of CCWs per track for EF data sets from 24 to 1. It would be easy to presume that reducing the number of CCWs is the only contributing factor to improving performance. However, MIDAWs also reduce the number of CCWs per track for non-EF data sets from 12 to 1, yet non-EF response time does not change. On the other hand, the channel utilization for non-EF data sets was reduced by half, from 51.5% to 26.2%, as shown in Figure 7 on page 9.

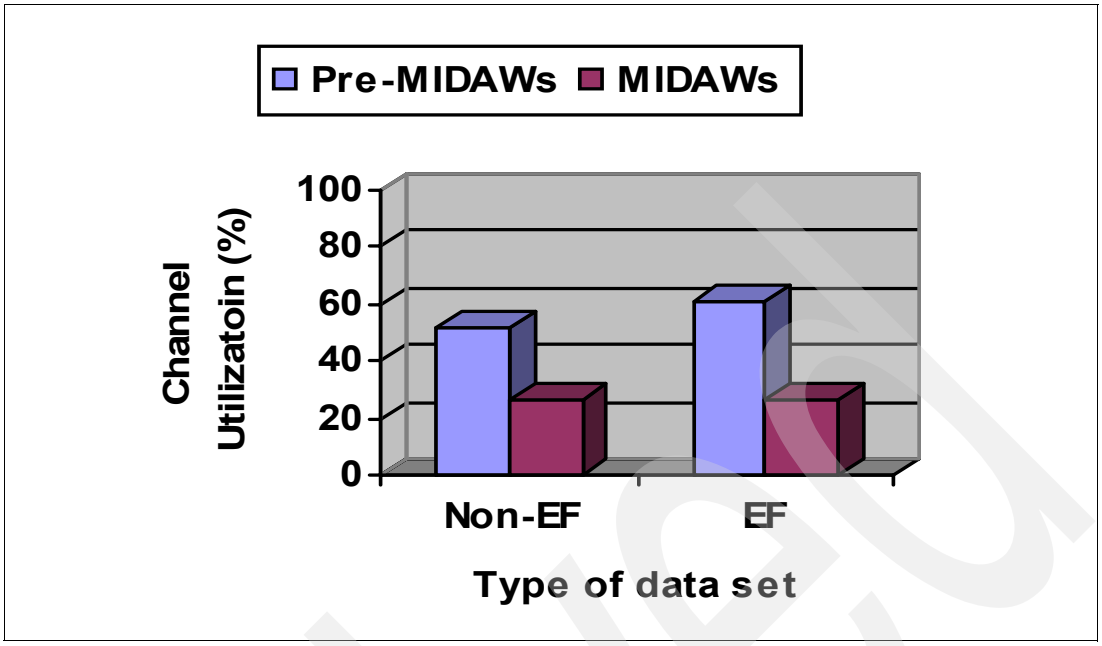

Figure 7 DB2 prefetch I/Os channel utilization

The reason why indirect CCW addressing reduces the FICON channel utilization is beyond the scope of this paper. Here we simply accept the fact that channel utilization is reduced. Yet, of what value is reduced channel utilization if the response time does not improve? The answer relates to multiplexing.

Consider the performance of EF data sets when there are multiple DB2 prefetch streams with non-EF data sets. For analyzing parallel streams, it is best to switch our metric focus to throughput in order to ignore the channel queuing effects. Look at Figure 8 to see the effect of MIDAWs on throughput with parallel prefetch streams given a single channel.

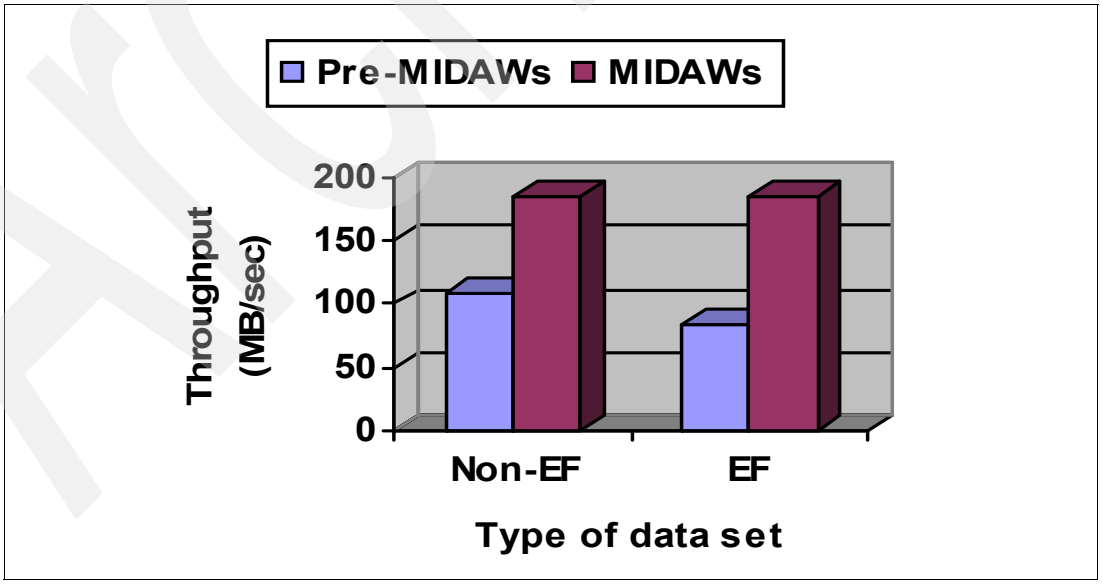

Figure 8 DB2 prefetch using parallelism: single channel

MIDAWs increased the throughput of EF data sets by 120% and increased the throughput of non-EF data sets by 70%. The non-EF results are a clue about what to expect for other types of 4 KB data sets besides DB2 data sets because M/M uses the same type of channel

programs for all data sets. As explained at the beginning of this paper, the user simply has to use data set types that are accessed through M/M.

A 2 Gbps FICON link is nominally capable of 200 MBps in one direction. With MIDAWs we achieve 186 MBps. The difference between 186 MBps and 200 MBps is the remaining effect of protocol overhead, but as we will see later in this article, both FICON Express2 and the DS8000 Host Adapter ports are capable of higher throughput when combining reads and writes.

Besides reducing the channel utilization, indirect addressing also has the effect of reducing the utilization of the control unit FICON Host Adapter port. This effect is hidden, because the system does not have any way of measuring the utilization of the Host Adapter port. The Host Adapter has to process every CCW, with or without data chaining, but indirect data addressing is transparent to the Host Adapter. MIDAWs help reduce Host Adapter utilization, even though we cannot measure it.

Next, consider a DB2 table scan with a very large degree of parallelism, but instead of using a single channel, we use eight channels connected to a DS8000—and we increase the degree of parallelism to 64. In this case, the channel utilization will be lower but the utilization of the DS8000 will be higher (compared to a single channel). Figure 9 shows the measured results. Compared to a single channel (shown in Figure 8 on page 9), the EF benefits were less but still dramatic. On the other hand, the non-EF benefits were significantly less.

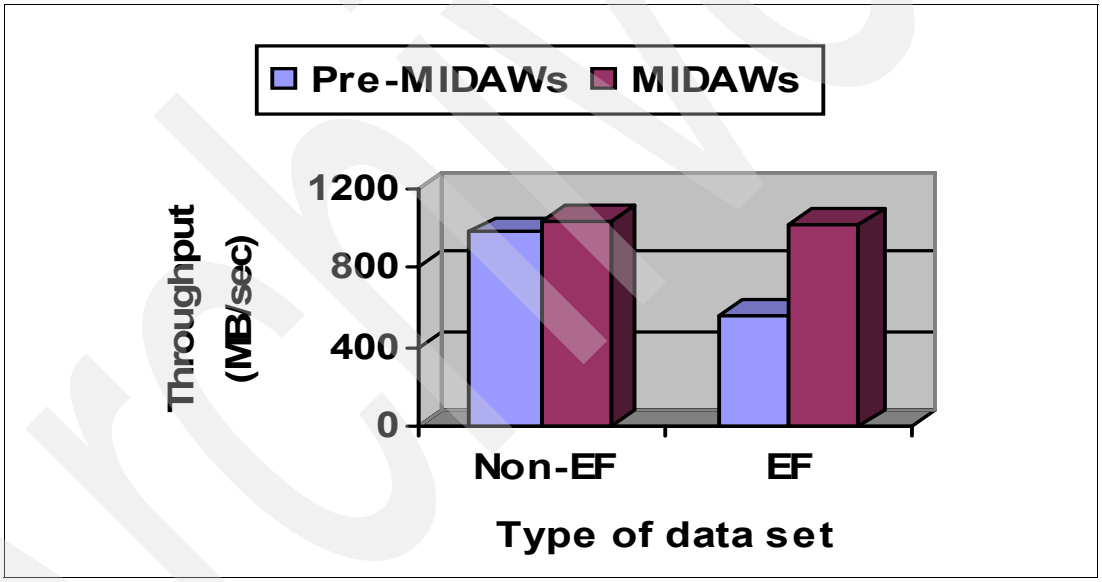

Figure 9 DB2 prefetch using parallelism: eight channels

Nevertheless, MIDAWs lowered the channel utilization for the eight channels using non-EF data sets from 64% to 36%. If the partitioned table space were spread across two DS8000s sharing the same eight channels, then we might expect the higher channel bandwidth to manifest itself in lower response time for the table scan. The lesson learned here is that improved channel efficiency does not always manifest itself in lower response time, because MIDAWs do not improve the efficiency of the control unit bus. To the extent that the control unit bus is a performance bottleneck, the benefits of MIDAWs will be lower for non-EF data sets, but the benefits using EF data sets will still be quite strong.

Block size effects

MIDAWs also affect the relative performance of different block sizes. The 3390 format on DASD wastes about 500 bytes for inter-record gaps. For 4 KB records or record sizes that are

a multiple of 4 KB, these gaps waste about 15% in DASD space and also add 15% to the time to sequentially read records from a disk into the control unit cache. Physical disk performance is irrelevant if the data permanently resides in cache, or if the channels are faster than the speed at which data can be sequentially read from the disks. However, the block size can also affect FICON channel performance.

Figure 10 shows the I/O response times for a DB2 prefetch I/O using different page sizes for EF data sets.

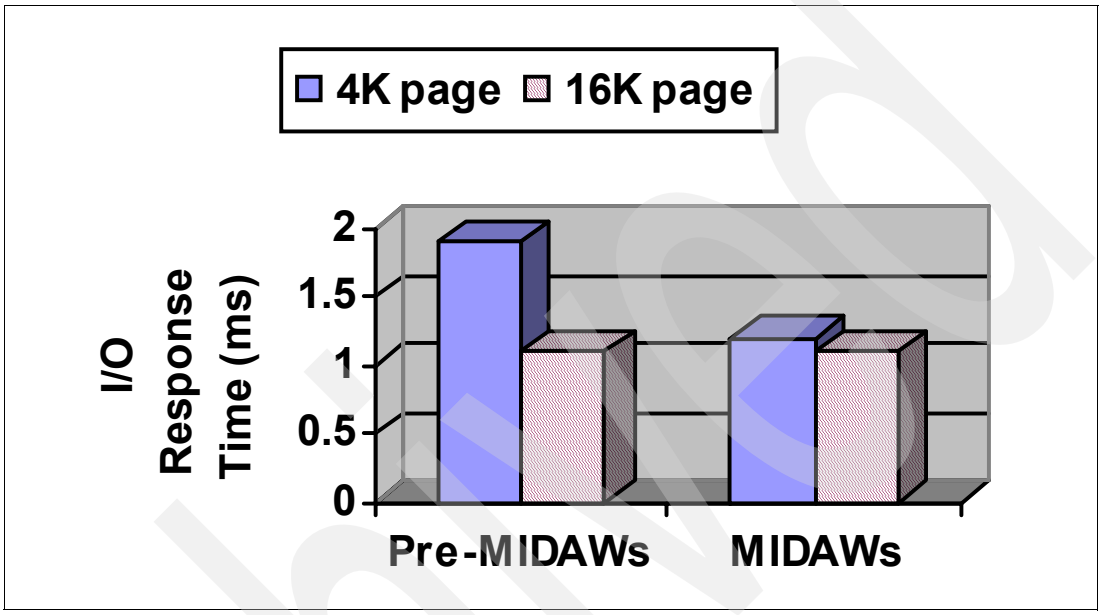

Figure 10 DB2 prefetch EF data sets

Prior to MIDAWs, the response time was 1.9 ms for 4 KB pages and 1.2 ms for 16 KB pages. Using MIDAWs, the response time for 4 KB pages dropped to 1.2 ms, but the 16 KB pages dropped only slightly to 1.1 ms. Thus there was very little difference between the performance of 4 KB pages and 16 KB pages. With non-EF data sets, not shown here in a chart, the 16 KB response time was 1.0 ms.

When using MIDAWs and a non-EF DB2 data set, the channel programs for a prefetch I/O look identical for different page sizes. That is, there are 12 MIDAWs per track in all cases. Hence, the record size has become irrelevant to channel performance. That the page size affects the I/O response times is now entirely due to control unit effects. In other words, besides eliminating the EF performance penalty, MIDAWs also eliminates the small block performance channel penalty, except to the extent that the record size affects the number of bytes that fit on a track. Small blocks are usually better for database transaction workloads, because the buffer hit ratio for random access tends to be higher with smaller blocks.

Writes

Up to this point we have discussed only reads, but MIDAWs and track-level CCWs are also important for writes. Because there are basically three types of write CCWs in ECKD, we consider each type individually.

The first type of write CCW is called an *update write*. A data set has to be formatted before update writes can be used. Update writes are often used by databases, but are not commonly used by the QSAM or BSAM access method. Most writes that occur because of SQL processing in DB2, including the DB2 logs, are update writes. The type of chaining that is

used for update write channel programs is exactly the same as for reads. Hence, MIDAWs affect the performance of update writes in the same manner that they affect reads.

The other two types of write I/Os are both format writes. The first of these is a *record-level format write* (which is sometimes called *Write CKD*), and the second of these is a *track-level format write*.

Write CKD operations have existed since S/360. Write CKD operations are used by all of the access methods when writing to a data set (when the data set is opened for “output” rather than “update”). Likewise, DB2 utilities use Write CKD operations when writing to a table space. Prior to MIDAWs, data chaining was used to chain the 8-byte count field to the data portion of a record, but command chaining was used to span records. Media Manager replaces MIDAWs for the data chaining, but the command chaining remains. Thus, MIDAWs help only partially with Write CKD operations. For example, an I/O that formats twelve 4-KB blocks on a track will require 12 CCWs instead of 24 CCWs for a non-EF data set or 36 CCWs for an EF data set.

Track-level operations are used by M/M when requested by M/M caller to *preformat* some tracks of a data set. DB2 uses this M/M service when preformatting a DB2 log or a DB2 page set. DB2 preformats a pageset asynchronously when an insert of a row causes the high used RBA (relative byte address) to increase. The chaining that is used for track-level CCWs is much the same as for sequential reads and update writes. That is, no chaining is used any longer within an individual track. Thus, preformat I/Os will benefit from MIDAWs in much the same way as DB2 prefetch I/Os and sequential update writes.

Other I/O configurations

We have been considering the performance of MIDAWs in the presence of FICON Express2 channels connected via a 2 Gbps link to a DS8000 control unit. MIDAWs should help older I/O configurations, but to a lesser degree. As shown in Figure 4 on page 5, the EF performance penalty was less when the I/O configuration was slower. Consequently there is not as much to gain on the older I/O configurations.

MIDAWs do not affect ESCON® performance at short distances. MIDAWs might improve the performance of EF data sets with ESCON at extended distances, but because ESCON does not support multiplexing, non-EF data sets will not improve.

DB2 table scans

We have previously discussed DB2 prefetch I/Os, but we did not actually talk about how prefetch I/Os affect the elapsed time of DB2 table scans or index scans. Of course, there is no effect if the table scan is CPU bound, but even I/O bound queries are subject to CPU effects. Because DB2 schedules prefetch I/Os to be done under an asynchronous service task, there is scheduling overhead as well as time to “page fix” and “unfix” the I/O buffers. However, DB2 introduced the ability to use long-term page fixing to avoid the dynamic page fix costs. As the I/O speeds increase, it becomes more important to use long-term page fixing. Figure 11 on page 13 illustrates the effects of long-term page fixing on the time to scan 1 GB of an EF data set, with and without MIDAWs. Prior to MIDAWs, long-term page fixing reduced the elapsed time by 13.5%. With MIDAWs the elapsed time reduction was 19% and we read a gigabyte of 4 KB pages in about 11 seconds.

Large pages do not require MIDAWs to achieve excellent I/O performance, but large page sizes already experienced profound performance improvements from FICON Express2 and the DS8000. MIDAWs are really targeted at the 4 KB pages. Consequently, DB2 large objects (LOBs) do not benefit from MIDAWs.

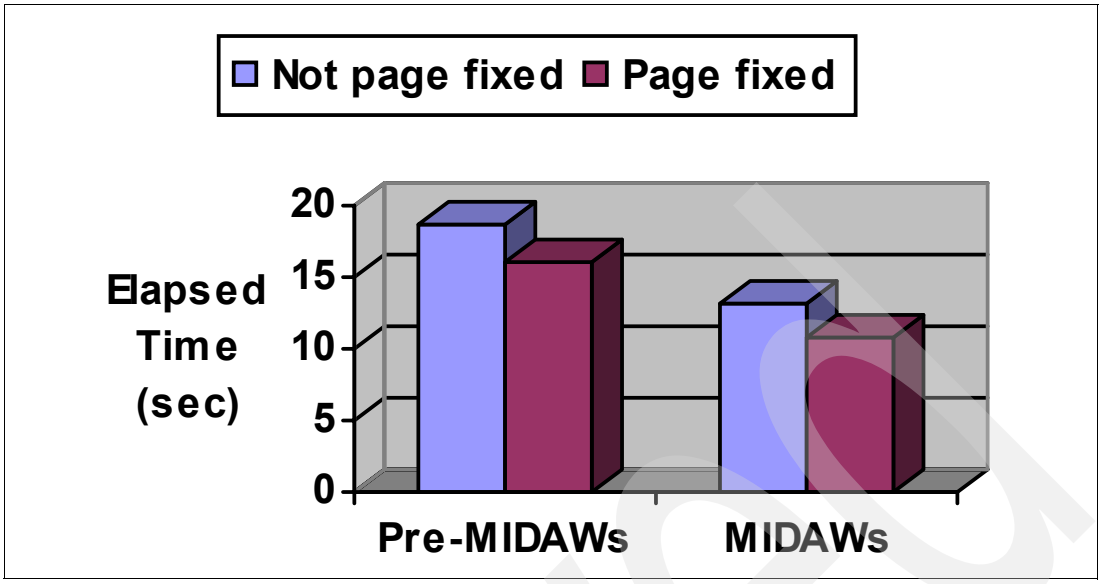

Figure 11 DB2 table scan of 1 GB on System z9-109

DB2 logging

DB2 logs always use 4 KB pages. Most applications do not stress the sequential log I/O bandwidth of the DB2 log, but mass sequential inserts might stress the log I/O bandwidth if the row sizes are large and the commits are infrequent. Sequential inserts or updates using small row sizes also might stress the log I/O bandwidth if the log latch and the CPU are not a bottleneck. Using 4000-byte rows, sequential inserts will generate lots of log data with minimal CPU time. Such a program was measured, and the log throughput was determined, as shown in Figure 12.

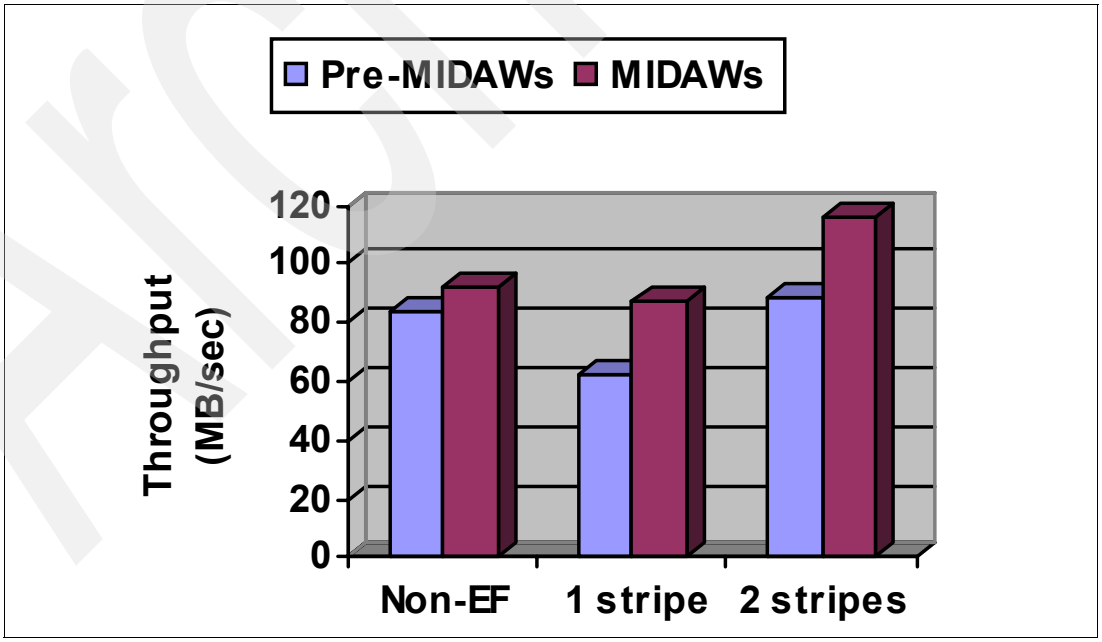

Figure 12 DB2 log throughput

Prior to MIDAWs, the maximum log throughput using the DS8000 and FICON Express 2 was 84 MBps, and striping the log increased the bandwidth only slightly. Nobody should ever

allocate their logs as EF with a single stripe, because there is no advantage to doing so. (DB2 does not support Extended Addressability for log data sets.) Nevertheless, Figure 12 on page 13 shows how a single-stripe EF log would perform, so that we can compare two stripes to one stripe. Given a log with two stripes, Figure 12 on page 13 shows that MIDAWs increased the log bandwidth by 31%, reaching 116 MBps with two stripes.

DB2 utilities

Most DB2 utilities tend to be CPU bound, but at the same time they do a lot of sequential I/O. Consequently MIDAWs will help reduce the FICON channel utilizations, but do not necessarily reduce the elapsed time of all DB2 utilities.

Two DB2 utilities in particular are not CPU bound: the Copy utility and the Recover utility. Generating image copies can be a daily or weekly activity for some shops, and achieving good Recover performance is important for both performance and availability. MIDAWs will significantly improve the performance of both of these utilities when the page size is 4 KB.

Figure 13 illustrates the performance of the Copy utility using data sets of 4 KB page size. MIDAWs increase the throughput for non-EF data sets by 13% and for EF data sets by 83%.

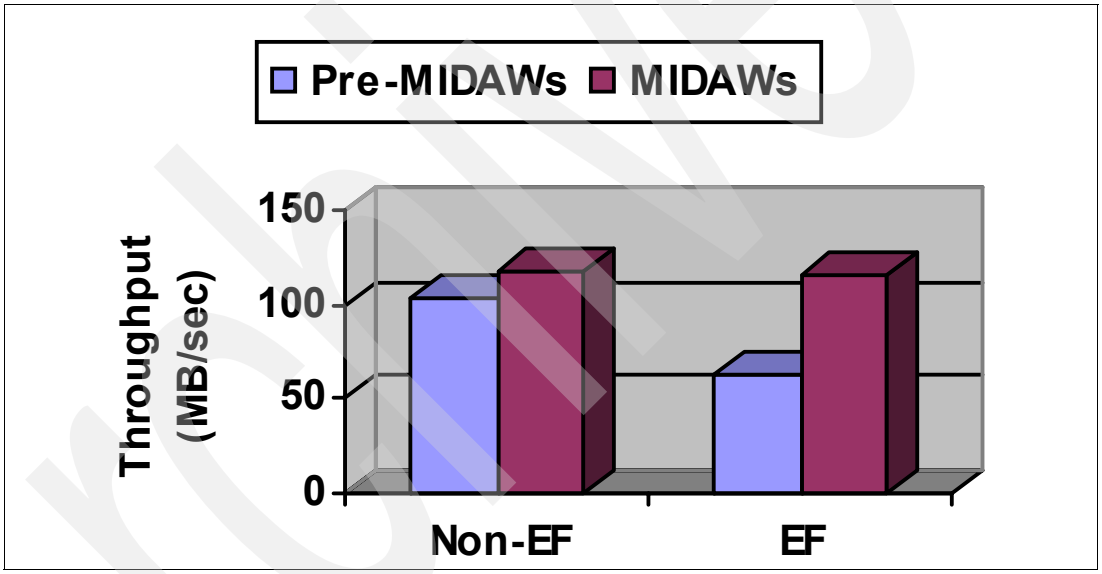

Figure 13 DB2 Copy EF 4 KB data sets

Similarly the restore phase of the Recover utility (which is the reverse of Copy) improved by 10% and 41% respectively for non-EF and EF data sets. Recover did not improve as much as Copy due to a limitation in the maximum size channel program built by Media Manager — a limitation that might be relieved at a future date.

Using DFSMS striping for DB2 data sets

The value of using DFSMS to stripe data sets with 4 KB records has been limited by the performance problems of EF data sets. The MIDAW facility removes these limits and increases the performance advantages of DFSMS striping.

However, the value of striping is still limited for DB2 table spaces, because the size of a DB2 prefetch I/O is limited in DB2 Version 8 to 128 KB for queries. Given two stripes, the 128 KB is split into two smaller I/Os of 64 KB. Measurements of a table scan using MIDAWs showed that the elapsed time of a table scan using two stripes was 25% less than one stripe, but we

do not see the 50% reduction we might hope for. The limited number of prefetch buffers is one reason why DB2 parallel table scans outperform DFSMS striping. So, whenever partitioning is possible, it is still the preferred solution over striping. Partitioning is the best method for optimizing the performance of I/O-bound queries and utilities.

Sequential inserts are a special type of workload that benefits from striping. To optimize the performance of sequential inserts, it is beneficial to stripe both the DB2 log and the table spaces.

Summary

The MIDAW facility achieves superior performance for a large variety of workloads by improving the efficiency of the channel subsystem. Because MIDAWs are used only by the Media Manager, and MIDAWs benefit only small record sizes, only certain types of data sets are beneficiaries. Some examples of data sets that are accessed through Media Manager are VSAM data sets (including all linear data sets), Extended Format data sets, and PDSEs. The most benefit occurs with Extended Format data sets that have small block sizes. Because DB2 depends on Extended Format data sets to stripe the logs, or to enable data sets to be larger than 4 GB, DB2 is a major beneficiary.

Disclaimer

The performance measurements that are described in this paper were performed in a controlled environment in an IBM laboratory. These measurements are intended to illustrate the behavior of IBM products, but actual performance that a user will experience will vary depending upon considerations such as the hardware and software configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve the performance characteristics that are stated here.

This publication was produced in the United States. IBM might not offer the products, services, or features discussed in this document in other countries, and the information might be subject to change without notice. Consult your local IBM business contact for information about the product or services that are available in your area.

References

IBM z/Architecture Principle of Operations, SA22-7832-04

Glossary

BDAM	Basic Direct Access Method
BLKSIZE	Block size parameter
BSAM	Basic Sequential Access Method
CICS	Customer Information Control System
CI	Control Interval
CKD	Count Key Data
CPU	Computer Processing Unit
DASD	Direct Access Storage Device
DB2	IBM relational database system
DCB	Data Control Block
DFSMS	Data Facility System Managed Storage
DSORG=PS	Physical Sequential Data Set Organization
EA	Extended Addressability
EF	Extended Format
ESCON	Enterprise Systems Connection

ESS	Enterprise Storage System; a class of IBM disk storage control units
FICON	Fiber Connection
Gbps	Gigabit per second
HFS	Hierarchical File System
IDAW	Indirect Data Address Word
IMS™	IBM transaction and hierarchical data base management system
I/O	Input/OutputOSAM; a special access method supplied with IMS
LDS	Linear data set
PDS	Partitioned Data Set
PDSE	Partitioned Data Set Extended
QSAM	Queued Sequential Access Method
RMF	Resource Measurement Facility
SQL	Structured Query Language
VSAM	Virtual Storage Access Method

The team that wrote this Redpaper

This Redpaper was produced a team of specialists from around the world working with the International Technical Support Organization, San Jose Center.

Jeffrey Berger is a member of the DB2 for z/OS performance department in IBM Silicon Valley Laboratory. For most of his 26 years at IBM, Jeff has worked on system performance of IBM mainframes, specializing in DASD storage and database systems, both hardware and software. Jeff has contributed several patents and several papers, which were published by Computer Measurement Group and the DB2 IDUG Solutions Journal.

Paolo Bruni is an Information Management software Project Leader at the ITSO, San Jose Center. He has authored several Redbooks™ about DB2 for z/OS and related tools, and has conducted workshops and seminars worldwide. During Paolo's many years with IBM, previously as an employee and now as a contractor, his work has been mostly related to database systems.

Acknowledgements

The origins of the MIDAW facility date back to the first introduction of FICON channels in 2000 when Jeff Berger first noted that EF Data Sets did not perform as well as non-EF Data Sets. It was quickly noted that data chaining had more of an impact on FICON architecture than on ESCON architecture. Harry Yudenfriend, a Distinguished Engineer at IBM Poughkeepsie, identified the best solution and led the effort to implement the solution. That solution was MIDAWs. Harry also recognized that the benefits of MIDAWs went well beyond Extended Format Data Sets.

The authors also thank Catherine Cronin, who lent her expertise in the area of FICON channel architecture. Both Harry and Cathy made valuable editorial comments about the paper. Thanks also to all of the many people in IBM Poughkeepsie who lent assistance with the MIDAW measurements.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

CICS®
DB2®
DS8000™
ECKD™
ESCON®
FICON®
IBM®
IMS™
Redbooks™
Redbooks (logo) ™
RMF™
S/360™
System z9™
z/Architecture™
z/OS®
zSeries®
z9™

Other company, product, or service names may be trademarks or service marks of others.