IBM Systems and Technology Group
Thought Leadership White Paper

May 2013

Faster Oracle performance with IBM FlashSystem

Executive summary

This whitepaper discusses methods for improving Oracle® database performance using flash storage to accelerate the most resource-intensive data that slows performance across the board.

To this end, it discusses methods for identifying I/O performance bottlenecks, and it points out components that are the best candidates for migration to a flash storage appliance. An in-depth explanation of flash technology and possible implementations are also included.

The problem of I/O wait time

Often, additional processing power alone will do little or nothing to improve Oracle performance. This is because the processor, no matter how fast, finds itself constantly waiting on mechanical storage devices for its data. While every other component in the "data chain" moves in terms of computation times and the raw speed of electricity through a circuit, hard drives move mechanically, relying on physical movement around a magnetic platter to access information.

In the last 20 years, processor speeds have increased at a geometric rate. At the same time, however, conventional storage access times have only improved marginally (see Figure 1).

The result is a massive performance gap, felt most painfully by database servers, which typically carry out far more I/O transactions than other systems. Super fast processors and massive amounts of bandwidth are often wasted as storage devices take several milliseconds just to access the requested data.

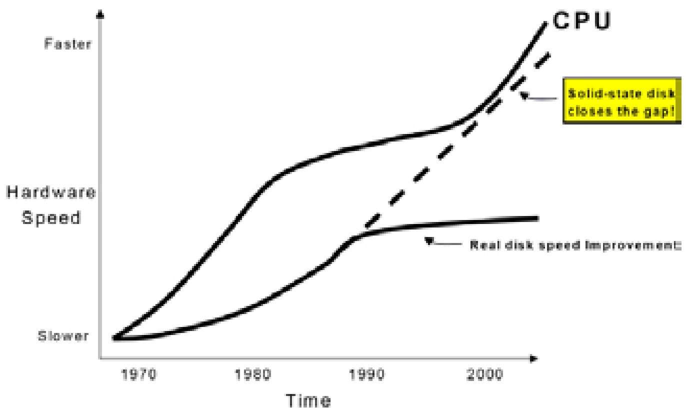


Figure 1: Comparing processor and storage performance improvements

When servers wait on storage, users wait on servers. This is I/O wait time. Flash storage systems are designed to solve the problem of I/O wait time by offering 250 times faster access times (.02 milliseconds instead of five) and 1333 times more I/O transactions per second (400,000 instead of 300) than a hard disk drive. Admittedly, multiple hard disk drives can be stacked to obtain thousands of IOPS, but it soon reaches a point of diminishing returns where the costs of power, floor space and air conditioning become prohibitive. In a test by a SAN manufacturer 496 disk drives were required to reach 100,000 IOPS in a RAID0 configuration, obviously something which is not desired.

Traditional approaches to Oracle performance

Decreasing application performance under heavy user loads is not a new story for most enterprises. The last three years have seen dramatic changes in demands placed upon database servers. While the number of users of database system has increased, so has the average amount of data stored in databases. Additionally, the demand for more complex business analysis has increased the complexity of the work done by database servers. The combination of more users, greater volume of data and more complex queries has frequently resulted in slower database response. The knee-jerk reaction to this problem is to look at two likely sources for database performance problems:

- Server and processor performance: One of the first things that most IT shops do when performance wanes is to add processors and memory to servers or add servers to server farms.
- SQL statements: Enterprises invest millions of dollars squeezing every bit of efficiency out of their SQL statements. The software tools that assist programmers with the assessment of their SQL statements can cost tens of thousands of dollars. The personnel required for evaluating and iterating the code costs much more. Dozens of consulting firms have appeared in the last decade that specialize in system tuning, and their number one billable service is SQL tuning.

In many cases, the money spent in these two pursuits can be significant, whereas the return is often disappointing. Server performance and SQL tuning alone does not often detect the true cause of poor database performance: the gap between processor performance and storage performance. Adding servers and processors will have minimal impact on database performance and will compound the resources wasted, as more processing power waits on the same slow storage. Tuning SQL can result in performance improvements, but even the best SQL cannot make up for poor storage I/O. In many cases, features that rely heavily on disk I/O cannot be supported by applications. In particular, programs that result in large queries and those that return large data sets are often removed from applications in order to protect application performance.

When system administrators look to storage they frequently try three different approaches to resolve performance problems:

- Increase the number of disks: Adding disks to JBOD (just a bunch of disks) or RAID is one way to improve storage performance. By increasing the number of disks, the I/O from a database can be spread across more physical devices. As with the other approaches identified, this has a trivial impact on decreasing the bottleneck.
- Move the most frequently accessed files to their own disk: This approach will deliver the best I/O available from a single disk drive. As is frequently pointed out, the I/O capability of a single hard disk drive is very limited. At best, a single disk drive can provide 300 I/Os per second. Flash storage systems are capable of providing hundreds of thousands of I/Os per second within a single 1U appliance.
- Implement RAID: A common approach is to move from a JBOD implementation to RAID. RAID systems frequently offer improved performance by placing a cached controller in front of the disk drives and by striping storage across multiple disks. The move to RAID will provide additional performance, particularly in instances where a large amount of cache is used. However, in order to reach high IOPS and meet user concurrency requirements the required number of hard disk drives soon becomes prohibitive.

Introduction to flash storage

Strictly, a solid state disk (or SSD) is any storage device that does not rely on mechanical parts to input and output data. However, SSD has come to mean a form-factor solid state device meant to take the place of an existing HDD, Flash storage is not to be confused with form-factor technology. Form factor SSDs use traditional infrastructure connections and controllers that were designed for hard disk drives and their high latency, low throughput limitations. Flash storage systems are designed from the flash chip up using fast FPGA controller technology to minimize latency and maximize bandwidth.

IBM® FlashSystem™ only uses the highest quality Flash available: single level cell (SLC) and enterprise multi level cell (eMLC). Most SSDs utilize less reliable, lower endurance MLC flash. eMLC flash has 10 times the life of MLC, and SLC flash has 33 times the working life of MLC, technology. MLC flash has a lifetime of 3000 write operations per flash storage cell location (designated as p/e cycles), eMLC has 30,000 and SLC over 100,000. Figure 2 shows this relationship between the various types of flash chip.

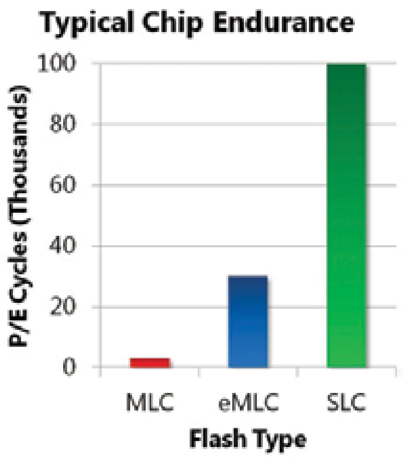


Figure 2: P/E cycles for various flash types

All-flash storage systems

All-flash storage systems offer higher capacity than any earlier forms of memory storage. This is because the all-flash storage systems do not require the additional batteries to allow flushing of the DDR cache during power outages and does not include large amounts of expensive DDR memory. Instead, a small amount of DDR is used to act as buffering for the flash

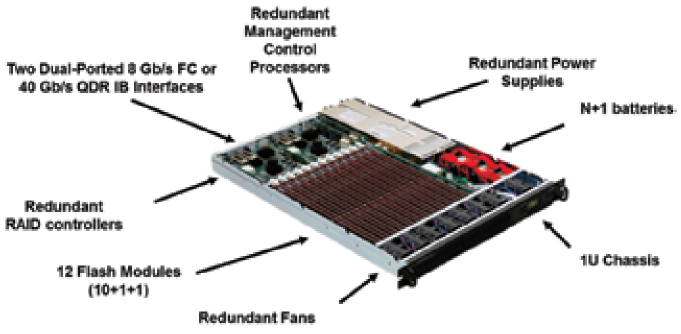

Redundant Management Control Processors
Two Dual-Ported 8 Gb/s FC or 40 Gb/s QDR IB Interfaces
Redundant Power Supplies
N+1 batteries
Redundant RAID controllers
12 Flash Modules (10+1+1)
1U Chassis
Redundant Fans

Figure 3: Example flash storage appliance architecture

for writes and to act as a meta-data repository during operation. Small batteries are used to provide electricity during loss-of-power situations to allow the flush of the small cache and metadata areas to Flash. With an All-Flash solution, 20 terabytes of addressable, high availability storage fit into a 1U form factor. Figure 3 shows the architecture of an all-flash flash storage appliance.

Identifying I/O wait time

Looking at operating system performance is the best way to identify I/O wait time. The tools to evaluate operating system performance vary by operating system. The following text gives some idea of the tools available.

Windows based systems

For Microsoft® Windows® operating systems the best tool for system performance analysis is **PerfMon**. Unfortunately, **PerfMon** does not provide actual I/O Wait Time statistics. It does, however, include real time processor performance levels. *Processor: % Processor Time* measures the actual work being done by the processor. If a system is hit hard by transactions and yet *% Processor Time* is well under 100 percent it is possible to infer severe I/O wait time. Systems that implement solid state disks will typically show high *% Processor Time* numbers.

As an example, two screen shots are included from Windows Performance Monitor. The tested system has dual Intel® Xeon® 2.8 GHz processors, 1 GB RAM and is running Windows Server® 2000.

Figure 4 shows the *Processor: % Processor Time* for a Server 2000 system with Intel's IOMeter performing 100 percent random writes to a hard disk drive. Here, you can see that the processor utilization averages around 1.8 percent. However, if you try to

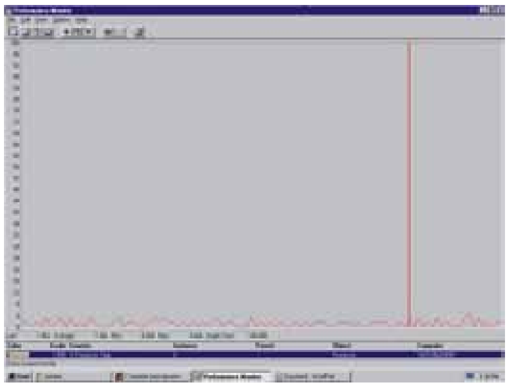


Figure 4: Processor performance when writing to hard disk

run additional applications on this system, the processor utilization would only marginally increase because the processor is tied up waiting on I/O from the hard disk drive. In this example, IOMeter shows that on an average there were 150 writes per second (150 IOPS) to the disk drive.

Figure 5 shows the same system and same access specifications in the IOMeter program running against FlashSystem. In this example, the processor averages 68 percent utilization. The IOMeter shows that 37,000 writes per second are going to the RamSan (37,000 IOPS). The flash storage has provided a 3677

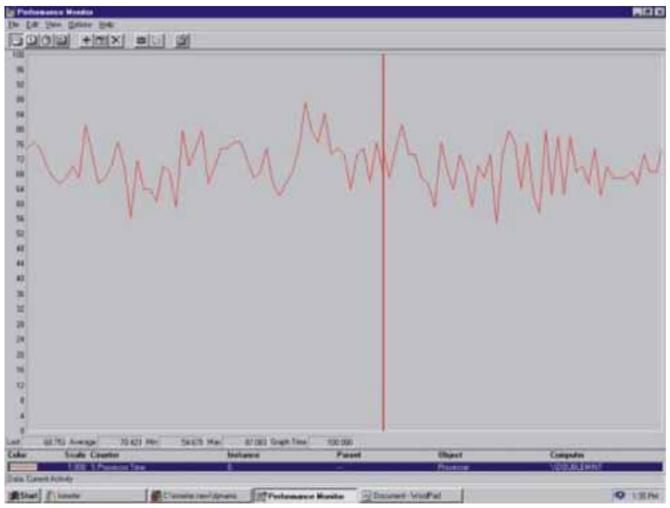


Figure 5: Processor performance when writing to FlashSystem storage appliance

percent increase in server CPU utilization with total saturation of the Fibre Channel Host Bus Adaptor, and since this performance, higher than the best RAID, is only a fraction of RamSan's capabilities, several similar servers could be hooked up to receive similar results.

In addition to processor indicators, Texas Memory Systems, an IBM Company recommends looking at the Physical Disk: "Average Disk Queue Length" and "Physical Disk: Disk Bytes per Second" to detect bottlenecks in the disk subsystem. If these values are consistently high, consider moving files that are located on that disk to the solid state disk. A Disk Queue Length greater than 3 indicates a problem. Texas Memory Systems has developed an extensive whitepaper with suggestions for collecting and analyzing Windows Performance data from the PerfMon utility. Additionally, contact your IBM® sales person for a free I/O analysis of your report.

UNIX

For UNIX operating systems, the following commands are useful: `top`, `iostat` and `sar`. Depending on the command, you will receive slightly different output.

The top command, when executed on a Solaris system, produces results that have the following format:

```
load averages: 0.09, 0.04, 0.03 16:31:09

66 processes: 65 sleeping, 1 on cpu

CPU states: 69.2% idle, 18.9% user, 11.9% kernel, 0.0%
iowait, 0.0% swap

Memory: 128M real, 4976K free, 53M swap in use, 542M
swap free
```

The key is that this command provides the `% iowait` for the system. It is important to note that `top` provides a snapshot of performance.

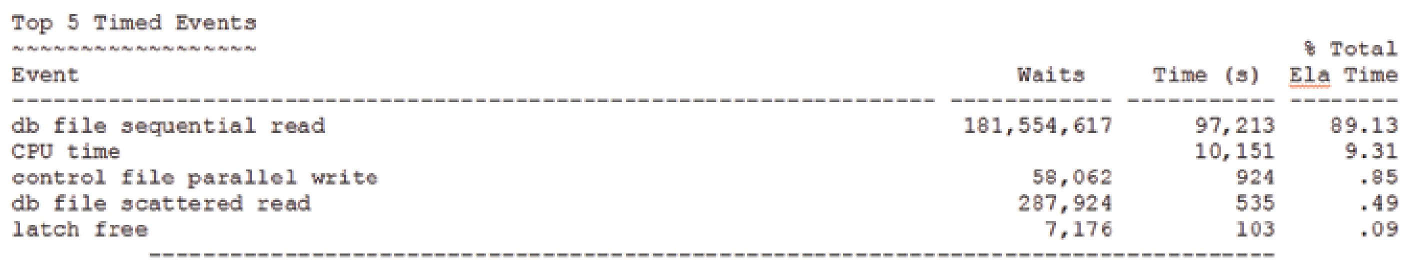
It is also reasonable to look at the vmstat command. This command will tell you how frequently your system is paging to virtual memory (disk). If you have frequent paging, it makes sense to consider adding RAM to your system or using flash storage appliance for paging. Paging to disk is another way that hard disk drives can introduce bottlenecks into system performance.

Oracle

Every version of Oracle since version 8.1.7.2 comes with the Statspack utility to monitor database performance. In version 10g, Oracle introduced the Automatic Workload Repository (AWR) along with Automatic Database Diagnostic Monitor (ADDM) as an extra cost option to their Enterprise Manager Tool partly in response to the growing complexity and cost of managing and tuning large databases. While AWR/ADDM was a significant enhancement to Statspack, even the best database tuning cannot fix slow disk subsystems. A Statspack report or AWR report must be captured during peak performance periods and can provide I/O related statistics to assist in determining which files would benefit from placement on solid state disks.

Oracle uses multiple database (or dirty buffer) writer processes to write changed data and rollback/undo data to disk, as well as log writer processes to write redo log data and archive log processes to write archived logs to secondary storage locations. However, each user process in Oracle does its own read from the disks; in large Oracle systems this can mean hundreds if not thousands of concurrent disk read requests. This requirement for large amounts of concurrent read access to the storage system is usually poorly understood by system administrators and is a major cause of disk contention issues in improperly configured disk RAID setups.

With regard to Oracle IO issues, the latest version of Oracle has added to the demands placed upon storage. In the good old days, Oracle had a database, some redo logs, archive logs and backups. Now, along with these, Oracle provides flashback functionality which while greatly enhancing the ability of the database DBA or developer to react to changes, it is one more area where data is being stored. The change from rollback segment to the new undo tablespace was a great benefit, but there are undo tablespaces in production systems that exceed 800 GB and rapid IO is critical to healthy system performance. In-memory-undo (IMU) was added in Oracle10g and has greatly reduced undo related disk operations, however, there are bugs which require turning IMU off and some special features (such as the logging required by change data capture) that will turn IMU off silently.



```
Top 5 Timed Events
~~~~~~~~~~~~~~~~~~                                                                        % Total
Event                                                           Waits    Time (s) Ela Time
-------------------------------------------------------------- ------------ ----------- --------
db file sequential read                                         181,554,617      97,213    89.13
CPU time                                                                         10,151     9.31
control file parallel write                                          58,062         924      .85
db file scattered read                                              287,924         535      .49
latch free                                                            7,176         103      .09
          -------------------------------------------------------------------------------
```

Figure 6: Sample Oracle Statspack

As mentioned at the beginning of this paper, one of the ways that system performance improvement was approached was to add more systems. Oracle's clustered database, Real Application Clusters (RAC), is one approach to add more systems to meet the demands of some of these massive database systems. One key point to this architecture is that all of these individual computers may be adding more computer power, but they all access the same shared disk systems increasing any existing IO stress.

The "Top 5 Timed Events" is the first place to look to begin understanding if the database is I/O bound. Figure 6 shows the top five events from a sample Statspack report.

The "Top 5 Timed Events" provide a snapshot of the database activity during the time period that a Statspack report covers. If these top events show that a majority of the database time is spent handling disk I/O, then Flash storage could provide a dramatic performance improvement. Table 1 provides a partial list of common events that indicate flash storage must be investigated and the database components that will benefit from use of a flash storage system.

Event	Description
db file sequential readl	The sequential read event is caused by reads of single blocks by the Oracle Database of a table or index. This is generally caused by an index read. The amount of time spent waiting for this event can be greatly reduced by moving the indexes to flash storage.
db file scattered read	The scattered read event is caused by reads of multiple blocks by the Oracle Database of a table or index. This is generally caused by a full table scan of the data tables. The amount of time spent waiting for this event can be greatly reduced by moving some of the data files to flash storage.
CPU time	This is the amount of time that the Oracle database spent processing SQL statements, parsing statements or managing the buffer case. Tuning the SQL statements and procedures, or increasing the server's CPU resources generally best reduce this event. It is an event that is generally not helped by flash storage.
log file parallel write	This event is caused by waiting for the writes of the redo records to the redo log files. This event can be greatly alleviated by using flash storage for all copies of the redo logs.
log file sync	This event is caused by waiting for the LGWR to post after a session performs a commit. This can be tuned by reducing the number of commits. Placing the redo logs flash storage can also alleviate this wait.
log file single write	This event is caused by waiting for the writes of the redo records to the redo log files. This event can be greatly alleviated by using flash storage for some or all copies of the redo logs.
free buffer wait	This wait occurs when a session needs a free buffer and cannot find one. A slow DBWR process that cannot quickly flush dirty blocks from the buffer cache can cause this. Moving the files that are receiving the majority of the writes to flash storage can help to alleviate the wait event. If poor I/O does not cause this wait write capacity, you can tune your instance by increasing the buffer cache.
control file parallel write	This wait is caused by waiting on writes to the control files. Moving the control files onto flash storage can help alleviate this wait.

Event	Description
buffer busy waits	The primary cause of these waits is contention for a block that is being used in a non-sharable way so that a read/write cannot be performed until the process that is using it is complete. Increasing the speed of the IO subsystem by using flash storage can alleviate this.
direct path read	This wait event is caused by reads that skip the database buffer. If there are a number of sorts and hashes taking place, then this can be caused by slow access to the TEMP space. Moving the TEMP space to flash storage can reduce this event.
direct path write	This wait event is caused by writes that skip the database buffer. If there are a number of sorts and hashes taking place, then this can be caused by slow access to the TEMP space. Moving the TEMP space to flash storage can help reduce this event.

Table 1: AWR events

Components to move to a flash storage system

Once you determine that your system is experiencing I/O subsystem problems, the next step is to determine which components of your Oracle database are experiencing the highest I/O and in turn causing I/O wait time. The following database components must be looked at:

Entire database: There are some databases that should have all their files moved to flash storage. These databases tend to have at least one of the following characteristics:

- High concurrent access: Databases that are being hit by a large number of concurrent users should consider storing all their data on flash storage since, as we know from the previous section, each user process in Oracle does its own disk reads. This will ensure that storage is not a bottleneck for the application and maximize the utilization of servers and networks. I/O wait time will be minimized and servers and bandwidth will be fully utilized.
- Frequent random accesses to all tables: For some databases, it is impossible to identify a subset of files that are frequently accessed. Many times these databases are effectively large indices.

- Small to medium size databases: Given the fixed costs associated with buying RAID systems, it is often economical to buy a flash storage system to store small to medium sized databases. A RamSan-710, for example, can provide 1 terabyte of database storage for less than the price of most enterprise RAID systems.
- Large read-intensive databases: Given the fixed costs associated with architecting a RAID system for performance: buying a large cache and buying a lot of spindles for striping, it is economical and much faster to buy a RamSan-820 pure flash solution in order to accelerate large read-intensive databases. A single RamSan-820 can scale to 20 TB of capacity in a 1U form-factor.
- Database performance is the key to company profitability. There is some subset of databases that help companies make more money, lose less money or improve customer satisfaction if they process faster. Flash storage will help make these companies more profitable.

Redo logs: Redo logs are one of the most important factors in the write performance of Oracle databases. Whenever a database write occurs, Oracle creates a redo entry. Redo logs are used in sequence with the best practice configuration using mirrored redo log groups, a minimum of two groups is required. Each redo entry is written to the two or more mirrored redo logs. Oracle strongly encourages the use of mirrored redo logs so that a backup redo log is available in the event of a failure. The operation is considered committed once the write to the redo logs is complete. Redo logs are used with linear output, if desired; the administrator can also configure redo logs to automatically archive. Archiving makes a copy of a filled log to another location before it can be reused. Archiving can be another source of waits in a slow disk based system.

The redo logs are a source of constant I/O during database operation. It is important that the redo logs are stored on the fastest possible storage. Writing a redo log to flash storage is a natural way to improve overall database performance. For additional reliability, it is useful to use mirroring to mirror flash storage systems.

Indices: An index is a data structure that speeds up access to database records. An index is usually created for each table in a database. These indices are updated whenever records are added and when the identifying data for a record is modified. When a read occurs, an index is consulted so that Oracle can quickly get to the correct record. Furthermore, many concurrent users may read any index simultaneously. The activity to the disk drive is characterized by frequent, small and random transactions. Under these conditions, disk drives are unable to keep up with demand and I/O wait time results.

By storing indices on flash storage, performance of the entire application can be increased. For online transaction processing (OLTP) systems with a high number of concurrent users this can result in faster database access. Because indices can be recreated from the existing data, they have historically been a common Oracle component to be moved to flash storage.

Temporary tablespace: When there cannot be enough server memory allocated, temporary segments are used to support temporary data during certain Oracle operations. The temporary tablespace segments support complex sort, hash, global temporary table and bitmap index operations.
Other SQL execution operations wait on temporary operations to complete, therefore it is only logical to place the temporary tablespaces on as fast a storage system as possible such as flash storage to reduce waits.

When complex operations occur they will complete faster if the temporary tablespace is moved to flash storage. Since the I/O to the temporary tablespaces can be frequent, disk drives cannot easily handle them.

Undo data: In databases with a high number of concurrent users undo segments can be a cause of contention. Undo data is created any time an Oracle transaction changes a record. In other words, if a delete command is issued, all the original data is stored in memory or in an undo segment stored in an undo tablespace until the operation commits. If the transaction is rolled back or undone, then the data is moved from memory or the undo segment back to the tables from where it was removed.

Because there is a possibility that the undo tablespace is hit with every change operation, it is useful to have the undo tablespace stored on flash storage. This will provide fast writes when the update transaction is created and will make the involved undo segment available more quickly for the next operation.

Frequently accessed tables: It is estimated that only 5 - 10 percent of data stored in OLTP systems is frequently accessed. These tables typically account for a large percentage of all database activity and thus I/O to storage. When a large number of users hit a table, they are likely going after different records and different attributes. As a result, the activity on that table is random. Disk drives are notoriously bad at servicing random requests for data. In fact, the peak performance of a disk drive drops as much as 95 percent when servicing random transactions. When a table experiences frequent access, transaction queues develop where other transactions are literally waiting on the disk to service the next request. These queues are another sign that the system is experiencing I/O wait time.

It makes sense to move the frequently accessed tables to flash storage. Flash storage performance is not impacted if performance is random. Additionally, flash storage systems by definition have faster access times than disk drives. Therefore, application performance can be improved up to 25 times if frequently accessed tables are moved to flash.

Notes

Notes

For more information

To learn more about IBM FlashSystem, please contact your IBM sales representative or IBM Business Partner, or visit the following website: **ibm.com**/storage/flash

TSW03191-USEN-01