Building More Competitive Devices on Windows Mobile

Windows Mobile is Microsoft's embedded operating system for battery-operated mobile devices including smartphones, PDAs, enterprise mobile computers (handheld scanners), and many other device types. While much of its OS core is shared with Windows Embedded CE (previously called Windows CE), Microsoft has added a common shell, UI enhancements, better phone features and a common application interface.

A large number of OEMs building very different devices choose Windows Mobile for its strong enterprise integration (Microsoft Exchange server for email), legendary flexibility, huge application developer ecosystem, streamlined SDK for developing apps, and the support that Microsoft provides for development, marketing and sales. These attributes have given Windows Mobile strong growth over the last four years, with several OEMs making devices that sold well over a million units.

Lately though, Windows Mobile has been experiencing a slowdown in growth in the Smartphone market, primarily due to increased competition from emerging operating systems like Mac OSX and Google's Android, which have created a wave of interest due to a dramatically enhanced user experience (iPhone) and with open source code and royalty-free licensing (Android). Apple has already taken a large chunk of the market and consumer mindshare with its iPhone product. If you are an OEM who believes the benefits of Windows Mobile are key to your customers, how can you optimize your product to better compete?

In this paper we will provide a high-level overview of the competitive landscape and discuss options that may allow OEMs using Windows Mobile to differentiate themselves from the competition by improving the user experience through performance gains in the data storage software stack.

Windows Mobile's Competitive Situation

Gartner recently released their analysis on the worldwide smartphone market. Even though Windows Mobile is not limited to the smartphone market, this segment represents a sizeable majority of its use. Analyzing the data allows us to get a snapshot of the challenges faced by Windows Mobile in today's marketplace.

Worldwide: Smartphone Sales to End Users by Operating System 4Q08 (Thousands of Units)

	4Q 2008		4Q 2007		
Company	Sales	Market Share(%)	Sales	Market Share (%)	Growth 4Q07-4Q08 (%)
Symbian	17,949.1	47.1	22,902.5	62.3	-21.6
Research In Motion	7,442.6	19.5	4,024.7	10.9	84.9
Microsoft Windows Mobile	4,713.9	12.4	4,374.4	11.9	7.8
Mac OS X	4,079.4	10.7	1,928.3	5.2	111.6
Linux	3,194.9	8.4	2,675.9	7.3	19.4
Palm OS	326.5	0.9	449.1	1.2	-27.3
Other OSs	436.9	1.1	411.3	1.1	6.2
Total	**38,143.3**	**100.0**	**36,766.1**	**100.0**	**3.7**

Source: Gartner (March 2009)

While Windows Mobile made significant gains in the early part of the decade, the data shows that its growth is slowing down. This can be attributed to several factors, increased competition and platform shortcomings being the two primary ones. More specifically, developers commonly express the following two frustrations:

1. **User interface**: The Windows Mobile UI contains several remnants of its old days where users had to navigate using a stylus. Even though improvements have been made in the application of touch screen technology, Windows Mobile lags behind the competition on this aspect.

2. **Device performance and responsiveness:** Windows Mobile users report sluggish performance and lack of responsiveness when doing common data related tasks. Performance has also been known to significantly degrade over time.

More than a few Windows Mobile OEMs have tackled the UI issue by adding their own interfaces. HTC, Sony Ericsson, Samsung all have their proprietary user interface overlaid on top of the Windows Mobile shell. Microsoft has also responded to this issue by announcing new releases with a better user experience. Windows Mobile 6.5 (slated for availability in Q2 2009) will feature a new UI for the homescreen. Windows Mobile 7, expected in 2010 will add more significant UI changes.

Unfortunately, the same progress has not been made on performance issues by either Microsoft or the device OEMs. In the past, manufacturers have tried to mitigate these issues by adding faster hardware (faster CPU, more RAM, more expensive flash parts, etc), but that has had the undesired effect of increased cost and reduced battery life, two elements also under increased pressure for improvement. Our experience shows that responsiveness of these devices can be enhanced significantly by using a more optimized and efficient software stack. One of the most fundamental areas for performance improvement is the software related to data storage.

Software Stacks

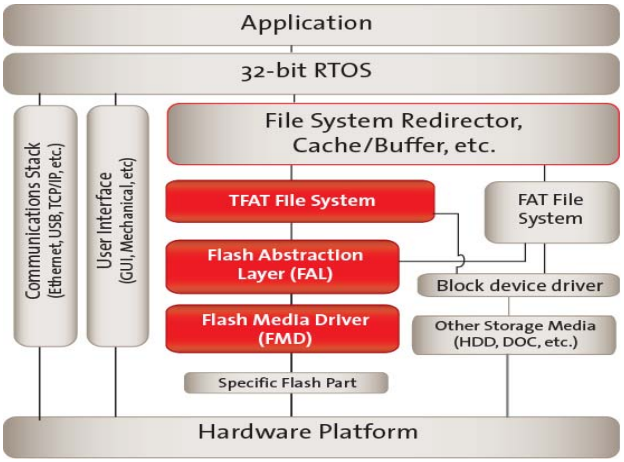
The OS is comprised of several important stacks that provide the functionality required by the device. Some of the prominent stacks are

1. Phone: support for phone bands, phone technologies (GSM, CDMA, etc)
2. Networking: TCP/IP, Ethernet, data networks (EDGE/3G, WiMax), Bluetooth
3. Applications (Browser, media player, database, etc)
4. Audio/Video processing

One interesting note is that most of these stacks operate autonomously and serve mostly self-contained functions. For example, the phone stack is utilized when you are making a call but not otherwise. The Bluetooth stack will be used only when you are transferring data via Bluetooth. Improving performance for any of these stacks will only improve responsiveness for that particular function set, but will not affect the performance of other functions. Hence, investing in optimization stack by stack offers limited return on investment from a user perspective. This is not to say it's not a good idea to invest in these areas, just that you need to have realistic expectations for the results.

The data storage stack, on the other hand, is used by a variety of device functions. Nearly all activities will have some element of data storage. For example, when you are browsing the web, the browser is constantly writing files to the browser cache. When you are transferring files using most any protocol or method, files are being read and written to the disk. Therefore, optimization of the data storage stack will improve performance for all other stacks that utilize it, leading to a more wide-reaching performance improvement.

Data Storage Stack on Windows Mobile



The data storage stack is usually comprised of 3 major elements:

1. Storage device driver
2. Block device layer
3. File system

For Windows Mobile (and Windows Embedded CE), the flash media driver (FMD) performs the task of input and output of data to a flash memory device. The FMD contains all of the device-specific code necessary for read, write, and erase commands to the flash memory device. If you change the flash part, you will need a different FMD. The FMD can be used with the flash

abstraction layer (FAL) to create a block driver. You can also link the FMD with a boot loader so that the boot loader can flash a run-time image.

The FAL enables hard-disk style, sector read/write access to the flash, rather than the low level FMD operations of reading/writing raw flash pages and erasing whole blocks.

Using the FAL enables a file system to run on top of flash. It is the file system which allows applications to read/write data in high-level file constructs like File Open/Close/Delete, etc. External storage devices (like SD cards) can be formatted using any file system (most commonly used one is FAT), but Windows Mobile requires a transactional file system for resident storage to ensure that the internal data has some protection against data corruption. The only transactional file system option provided by Microsoft for Windows Mobile is Transactional-FAT, commonly referred to as TFAT.

Note: *Data storage stack can also include block device drivers for removable storage cards, USB flash devices, etc. Though not covered in the scope of this paper, options for improving performance will exist there as well.*

MS Flash (FAL +FMD)

Since FAL and FMD are almost always used together, for the rest of this paper we refer to them collectively as MSFlash.

Flash has grown in popularity as the data storage medium of choice for embedded applications. Its advantages include; its non-volatile nature, low power requirements, high density, and rugged characteristics. Flash memory is also versatile, capable of serving as a read-only memory for both the storage and execution of program code, as well as a disk drive for the storage of code and data files.

However, flash memory has several characteristics that make it difficult to use in some applications; its inability to overwrite data without a prior erase operation, its requirement for special programming algorithms, and a limited lifetime consisting of a finite number of write/erase cycles. Some flash memory technologies such as NAND, ship from the factory with bad blocks and may develop more of these bad blocks over time. The ability to dynamically correct errors and manage replacement blocks is crucial to the successful use of NAND flash parts.

When evaluating effectiveness of a Flash Manager, we have to consider the following important functions it performs

1. Reliability in managing flash memory. This includes such functions as:

    a. Bad Block Management
    b. Error correction on reads
    c. Minimizing read and write disturb
    d. Atomic sector writes

2. Extending flash lifetime: Especially important with respect to NAND flash with its limited erase cycles. After ten thousand erases or so, a given erase block can go bad and no longer function. Flash managers employ sophisticated wear leveling schemes to ensure erases are spread across the disk to avoid premature failure.

3. Performance: How fast can the flash manager perform functions of sequential and random I/O? How fast does it mount (impacts device boot speeds)? These are important questions to consider when evaluating a flash manager.

In this paper, we will be focusing only on performance. For a more detail evaluation of flash management solutions for Windows Mobile, please contact Datalight.

TFAT (Transactional FAT) File System

A complete review of the different functions of a file system is out of scope for this paper, however in this section we will discuss specific considerations of TFAT which can cause interesting challenges for certain scenarios. If these scenarios apply to you, then it's important for you to do some due diligence before choosing the file system for your device.

Microsoft provides a good overview of TFAT on its MSDN site. Here are few other important considerations:

1. TFAT was designed as an add-on to make traditional FAT file systems more reliable. It was not designed from the ground up to provide 100% reliability against corruption. This has significant implications for performance.

2. TFAT requires the block device driver to provide atomic sector writes. **Not all block devices support this**, and use of TFAT on such a block device cannot prevent corruption due to power loss.

3. By default, TFAT only preserves modifications to the metadata (directory structure and file allocation table). It does not preserve actual file data unless the TransactData flag is set. Setting the TransactData flag makes the file system commit every write operation to the disk which can significantly slow down the device.

Datalight Storage Stack for Windows Mobile

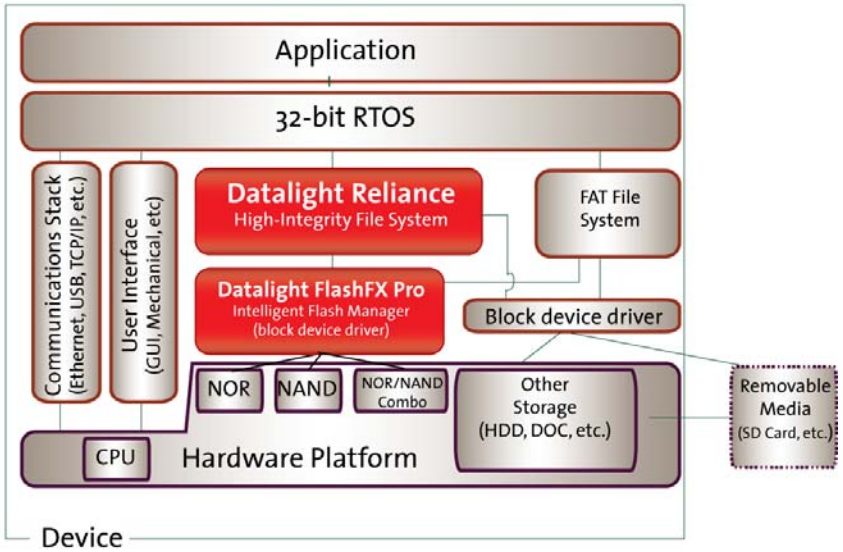
Datalight has been in the business of developing storage stacks since 1993, and was one of the first software companies to support Flash memory. Over the last 15 years, we have seen tens of millions of devices shipped with our storage stack. Because our sole focus is on this aspect of embedded software, we have fine-tuned the flash manager and file system to provide the

reliability, flexibility and performance that an embedded device needs to provide a favorable user experience.

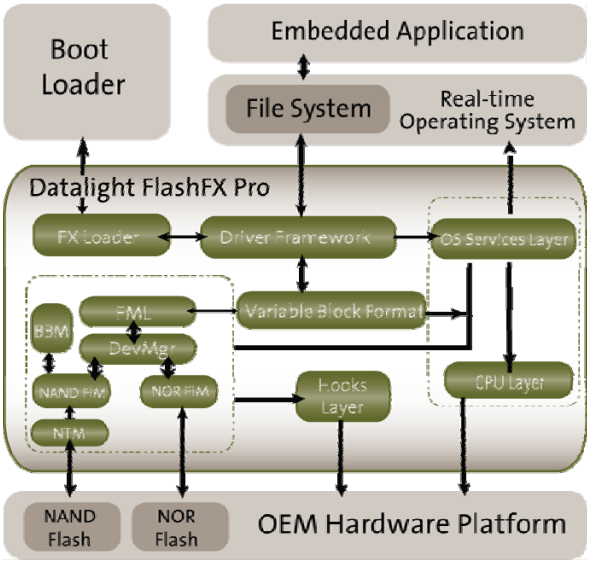
The Datalight storage stack is comprised of two products:

1. FlashFX Pro – Intelligent flash manager with built-in support for 300+ flash parts. Provides a block-device interface to the high-level applications/file systems.

2. Reliance Nitro – High-integrity file system that provides 100% reliability against data corruption at a logical level, while offering high performance and fast mount speeds.



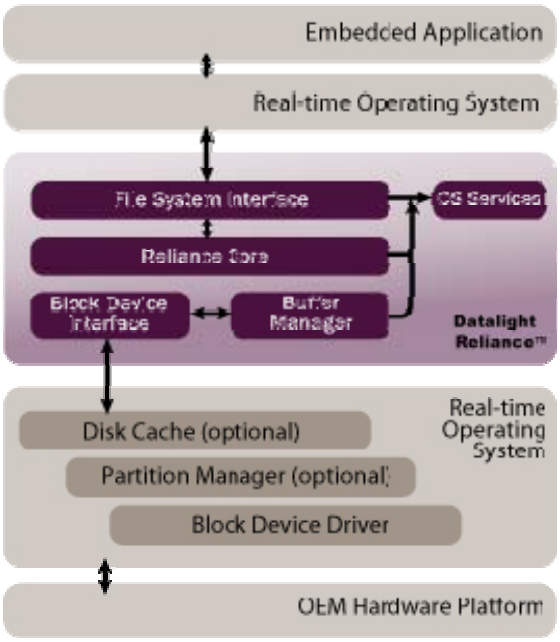
Datalight FlashFX Pro



FlashFX Pro is delivered as a highly modular product, with abstracted interfaces to the operating system, the flash hardware, and the hardware platform, as shown in the diagram below:

In terms of correlating the MSFlash and FlashFX Pro components, the MSFlash FMD layer is most akin to the FlashFX FIM and NTM layers, while the MSFlash FAL is roughly equivalent to the rest of FlashFX. For more details on Datalight FlashFX Pro, please visit http://www.datalight.com/products/flashfx/

Datalight Reliance Nitro File System



Reliance Nitro is a transactional file system that was designed from the ground up to provide 100% reliability on embedded platforms, with an architecture that provides much better performance all around. It maintains two states of the disk, a working state and a committed state, which allows it to provide consistent, logical file system reliability. Another benefit of an always-good disk state, Reliance Nitro can be mounted by just doing three logical block reads, leading to much faster system boot times.

For more details on how Reliance Nitro's transactional architecture provides 100% reliability, please review the presentation on Datalight's website.

Comparing Performance between the Two Stacks

Evaluating software stacks for performance is a complex exercise because there are multiple dimensions on which to evaluate it, and not all of them can be applied to every device. In this section, we will examine the differing aspects of performance for the data storage stack and see how they rate. Readers should determine which of the listed performance criteria are applicable to their device scenarios and decide what makes sense for their particular use case.

Performance Aspects for Data Storage

When evaluating data storage performance, the following elements should be considered:

1. **Sequential Read**: Determines how fast the data storage stack can read sequential data from the storage device. This is really important for boot speed. During boot, the device has to read the OS image from the storage device. Having high sequential read speeds hence can significantly improve boot times.

2. **Sequential Write**: Determines how fast the data storage stack can write sequential data from the storage device. This is really important for data transfer. Devices like MP3 players and smartphones need to quickly transfer data from the host computer to the device. A

device having high sequential write speeds can provide must faster user experience on these scenarios.

3. **Random Read/Writes**: Devices that write small chunks of data frequently are impacted significantly by this. Especially important for:
    a. Cache implementations
    b. Database applications
4. **File System mount speeds**: A file system needs to be mounted before it can perform any file operations. The mount process differs based on the type of file system (journaling v/s FAT v/s transactional). File system mount speeds can affect device boot times if the OS image is stored on the file system. Changes to the mount speed over time can significantly impact the user experience.
5. **File and Directory operations** (Open/Close/Delete): These are some of the most common operations on a device. Since almost all applications use the data storage via the file system layer, having a file system that is optimized for these operations can have a huge effect on device responsiveness.

Note that the first three aspects are applicable to both the block device driver and file system, but last two are only applicable to file systems.

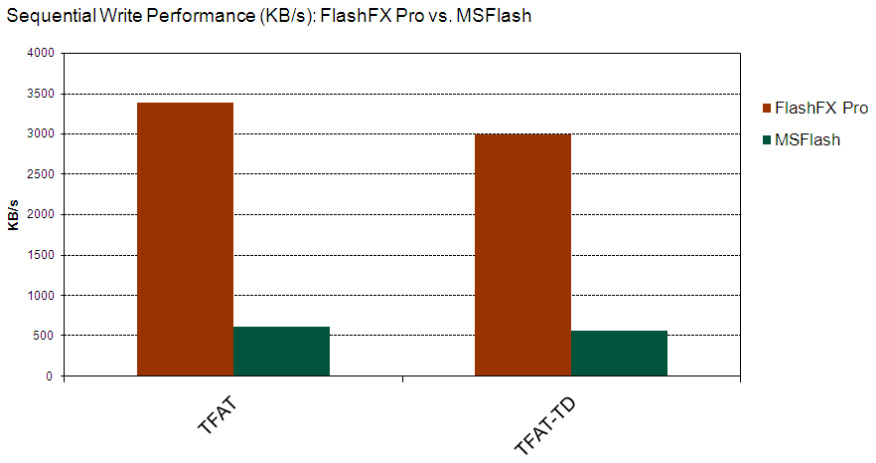
File and Directory operation performance is often neglected by developers. Most benchmarks focus of the first four aspects described above, because they are easy to quantify and compare. File operations can have a huge effect on overall device responsiveness. The time taken by the file system to create, open and delete files can add significantly to device lag depending on the use case. The following are some scenarios where this aspect is particularly important:

1. Database implementations: Many databases implement their data store as a set of files. As the database grows, the number of files grows as well. Time taken to open files (for indexing) can contribute greatly to database performance.
2. File stores: A browser (or any internet application) uses a cache to stored web pages downloaded from the internet. A mail client also maintains a file store for storing emails. As the number of files in the store increases, the file system performance can become a bottleneck for overall responsiveness of these applications.
3. Media applications (like music, video, still-camera): With large storage available on today's consumer devices and smartphones, the number of media files on any given device is on the rise. Having a file system that can operate with large number of files, can contribute significantly to the responsiveness of media applications.

Datalight FlashFX Pro compared to MSFlash

The hardware platform for all the tests described herein is the Marvel PXA320 based development platform. This platform includes resident NAND flash, and an integrated NAND Flash Controller, which includes hardware ECC calculation functionality. The platform comes with Windows Mobile 6.1 BSP.

The chart at the right shows that Datalight FlashFX Pro is approximately 4.5 times faster than MSFlash for sequential write speeds. Using FlashFX Pro can hence significantly improve Windows Mobile responsiveness for scenarios involving data transfer or any activity that requires significant writing to the disk.



FlashFX Pro with:	Random Write	Sequential Read	Random Read
TFAT	311% faster	14% faster	29% faster

The chart at the left shows how FlashFX Pro compares to MSFlash for Random Writes, Sequential Reads and Random Reads.

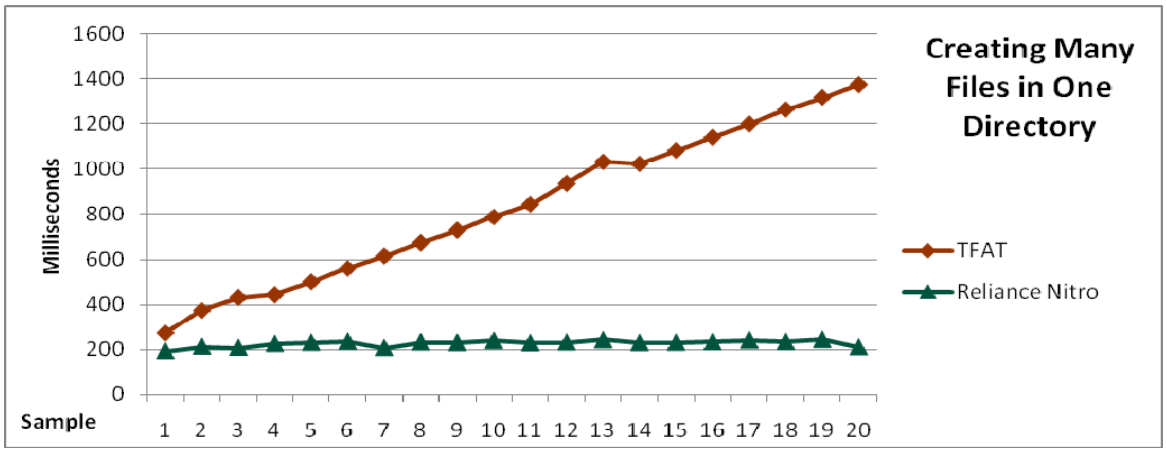
By providing a comprehensive advantage on all I/O operations, FlashFX Pro significantly enhances data storage performance on Windows Mobile.

File Operations

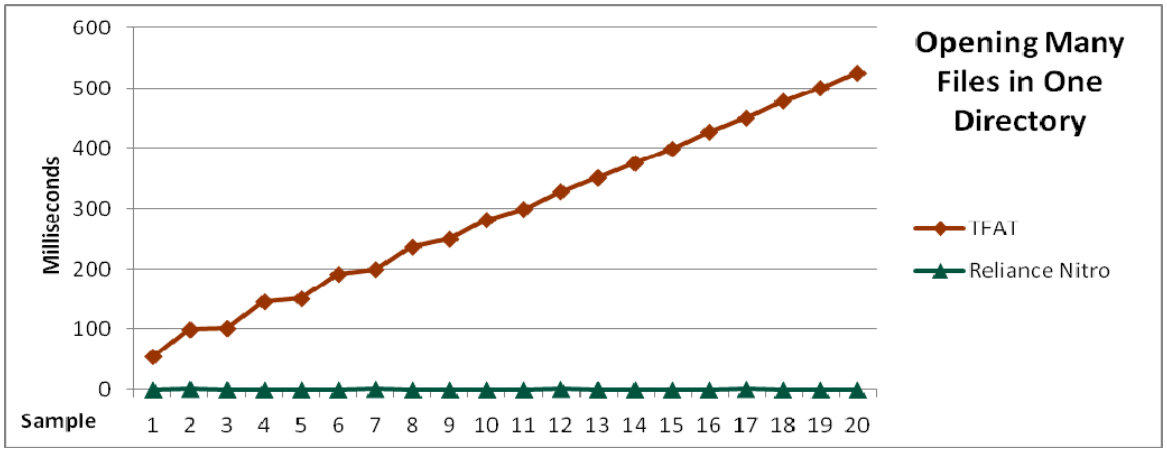
In these tests, we want to check the responsiveness of the file system in real world scenarios. The test benchmarks the time taken to Create, Open and Delete 20 samples of 50 files each (total 1000). This simulates scenarios such as:

1. Copying a lot of music files to the device (Create)
2. Sending a lot of text messages in a short amount of time (Create)
3. Indexing all emails stored on the device for faster search (Open)
4. Browsing through all available ringtones on the device (Open)
5. Browsing through pictures on your digital camera or photo frame (Open)
6. Clearing browser cache (Delete)

The graphs below plot the time taken to complete each sample set. Higher numbers mean the file system takes more time to complete the operation and performs more slowly.

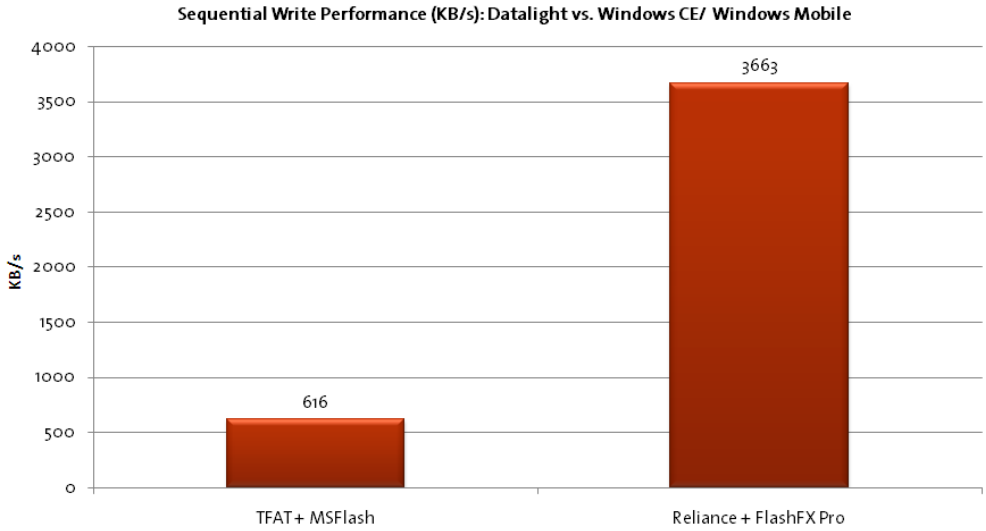


Depending on the number of files, Reliance Nitro can be 50% to 700% faster than TFAT on creating files.



The above chart is key. Due to Reliance Nitro's revolutionary tree-based allocation and directory design, the open times for each sample are below 1ms (and hence show up as 0 on the chart). So in cases where there are fewer than 50 files in the directory, Reliance Nitro is approximately **50 times** faster. At higher range, Reliance Nitro can be **500 times** faster than TFAT. Since '*file open*' is one of the most often run operations by the file system, this can translate into significant performance gains for your device.

Overall Performance of Data Storage Stack



Using the Datalight storage stack provides more than 5 times the performance of the default stack on Windows Mobile. This can provide a significant boost to your overall device performance and increase the responsiveness without needing to invest in upgrading hardware.

Conclusion

Lack of device performance and responsiveness is one of the important shortcomings of the Windows Mobile platform. Traditionally, OEMs have tried to address this using faster hardware to overcome deficiencies in the software stack. While this does help improve performance it also adds to the BOM costs and can have negative effect on battery life. Using an optimized software stack can help improve performance without adding significant costs. OEMs should evaluate several different data stacks that complement Windows Mobile and then make informed decisions based on a thorough cost-benefit analysis. Datalight's data storage stack offers significant performance differentiation for data operations and can be an important factor in differentiating your device in a world of significant competitive threats.