TeraSci - Mobi Philosophy

Anderson McDuffee

July 7, 2020

5362 Production Dr. Huntington Beach, CA 92649 714-896-0150

Web Site: www.terasci.com Email: engineering@terasci.com

Overview

The mobile device platform is designed with an extra layer in the architecture called the “Mobi”. The primary purpose of the Mobi layer is to sit between the Site Server and the mobile device platform called “ports”. It gives the ports access to the Site Server database services and shares the firmware and code sets for all ports in the Mobi.

The Mobi helps to address both architectural and logistical problems that arise from scaling the solution up when large volumes and high concurrency are required. It helps solve both classes of problems by grouping ports together into highly performant sets that optimize hardware and technology usage. In addition, the port configurations of Mobis create easily manageable sets of ports that will maximize port utilization without overwhelming operators.

Design Theory

Often times OEM solutions involve the end user providing their own Windows PC in order to flash and refurbish mobile devices using the OEM provided software. There are a number of issues with this approach. Windows PCs are large, consume a lot of space and power and are expensive due to licensing fees. They are also a pain to maintain with Windows Updates often taking up precious production time.

Each OEM has their own set of tools, if not many versions of the tools depending on the device that is being processed. There is no common design paradigm in Windows when it comes to mobile device drivers. So, invariably, not only do you need the correct version of the OEM tools you also need the correct device driver installed for everything to work. Many of these drivers will conflict with each other and for no justifiable reason considering there are 99% identical and really only differ based on USB Vendor and Product identifiers (i.e. VID and PID) that are compiled into the driver.

Then each tool has its own design and user interface. Some are user friendly, others aren't. Irrespective of how user friendly each one is, it certainly isn't easy or efficient to train operators to be productive with all the variations at hand. This is especially true if the tools are distributed across multiple physical stations that require moving devices around, further lowering productivity.

All of the above inefficiencies lead to the first major goal of a TeraSci Mobi. To provide a unified and simple one stop interface for processing any mobile device.

USB 2.0 Limitations

Probably the most significant issue with the myriad OEM solutions is that they all perform differently and overall do a terrible job of scaling to meet volume demands. In order to increase volume, without paying for more Windows PCs, end users will resort to connecting a bunch of USB hubs to increase concurrency. However, this presents a major performance bottleneck.

Currently, most mobile devices flash using USB 2.0 running in High Speed (HS) mode which provides a theoretical bandwidth of 480 Mb/s (Megabits) or roughly 60 MB/s (Megabytes). In reality, a well optimized device will saturate a single USB 2.0 connector at around 35-40 MB/s, which is a more realistic real world expectation for a USB 2.0 HS port. So, a single device really needs a dedicated USB 2.0 port.

A typical desktop motherboard has 4 dedicated USB 2.0 ports available, but only placing four devices behind each Windows PCs is an inefficient use of computing resources and floor space. So, throw a 4 port USB 2.0 hub behind each connector on the PC motherboard, right? Well, it may work, but it certainly won't be fast since all four devices on a single hub will share the 480 Mb/s upstream bandwidth, effectively running at 25% of their capability. That doesn't even factor in that cheap USB 2.0 hubs are notoriously flaky and when they decide to drop from the bus they will take all four devices with them.

Even if it is possible to get 16 devices connected to a single station with a bunch of hubs there is no guarantee that the OEM tools support flashing that many devices simultaneously, many don't. Those that do likely won't handle 16 parallel flashing threads on low end commodity hardware.

The above points are all important considerations in the second major goal of a TeraSci Mobi. To provide a flexible and modular design that achieves near linear performance scaling and isolates devices failures from creating cascading failures.

Modular Design

The internals of a Mobi were designed from the ground up to create a highly modular system that performs well, is easy to maintain and allows for a variable number of ports to best meet the requirements of our customers based on volume. Currently, there are 4, 8 and 12 port versions of the Mobi available. However, due to the modular design even a single port solution is possible.

Sticking with the TeraSci 1-to-1 philosophy, each port is specifically designed to test, flash and control only a single device at one time. Using this approach allows for a proper isolation of each port within a Mobi so that failures on one port won't impact other running ports in the Mobi. Full isolation of failures makes it much easier for the operator to correctly identify the failed device and either fix the issue or just swap to the next device. It also allows for online replacement or repair of a faulty port module since the module can be rebooted or physically removed from the Mobi without impacting the other running ports.

In fact, port modules are completely generic components and can be safely swapped with each other. In other words, nothing about port #1 makes it port #1 except for its physical presence in the first module slot of the Mobi. Furthermore, nothing about a port module ties it directly to a specific Mobi, so port modules can also be moved between Mobis assuming they are the same hardware revision. So, in many cases it is even possible to borrow an idle port from another Mobi to fix a faulty one.

Code & Firmware Versions

Although the ports are completely independent from each other, they do access code, firmware and other required resources from the Mobi over an internal network. What this means is that all ports within a Mobi will be running the same code revision. Additionally, it ensures that all ports will be flashing the same version of a device firmware.

The code revision and device firmware used by a Mobi are all managed through the Site Server database. So, all Mobis behind a Site Server will deploy and run the same version of all required code. Likewise, they will efficiently synchronize and cache all required device firmware versions from the local Site Server.

The process of deploying or updating the code and firmware versions is fully scripted such that the process can be easily run, in parallel, on all Mobis during non-business hours or in some cases during a shift change. Alternatively, the same scripted deployment process can be used to roll changes out to a single Mobi at a time in a rolling fashion.

Since code versions can be deployed to a single Mobi at a time it is possible to increment the required code versions, in the Site Server database, and deploy it to a single Mobi. The selected Mobi can then be used as an in-field staging and verification area to ensure new features or fixes are working adequately before deploying them to every Mobi in the facility. If there is an issue; simply roll back the version numbers in the database and re-deploy the selected Mobi to get it back to production operation. Otherwise, the updates can be deployed to the remaining Mobis during the next available window of non-operation.

Operator & Admin UIs

Mobis are also intended to be a point of control for operators and administrators that need to oversee the operation and maintenance of all ports within a Mobi. As such, the user interface (UI) is written using HTML, CSS and JavaScript in order to provide a clean and customizable

presentation. In addition, it allows easy remote access and observation, by TeraSci employees, when assisting or troubleshooting with issues.

The main UI is laid out to match the physical arrangement of test ports in the Mobi. Each physical test port gets a dedicated window in the UI so it is intuitive to associate UI messages and color coding's to their respective physical ports. The color coding provides a simple system by which to determine the state of a test port at-a-glance, For example, an idle port has a white background, whereas a port in an error state will have a more prominent red gradient.

In addition to the operator UI, there is also an administrative UI that allows for remote maintenance and troubleshooting of Mobi and port module hardware issues. It allows for operations as simple as forcing a hard reboot on a hung port board to more complex cases such as checking for shorts in power FETs and verifying current and voltage levels.

Summary

Using a generic Windows PC with OEM provided software comes with a number of limitations and frustrations that make it hard to setup a mobile device tester that is easy to use, performs well and scales to volume. TeraSci has taken the time to analyze and understand many pain points in order to come up with a highly modular mobile platform architecture that addresses many of the most significant issues with OEM solutions.

The Mobi design is the cornerstone to supporting flexible and scalable N-port deployments that are easy to use, easy to maintain and don't sacrifice performance. They make it possible to meet the needs of our customer whether your volume only requires 4 ports or hundreds of ports.