ECC”17

MEET [ORWL]!

THE GROUNDBREAKING SECURE SYSTEM

[ORWL]

Presenters

Gerard Duynisveld
Gduynisveld at Eltan dot com

Wim Vervoorn
Wvervoorn at Eltan dot com

WWW.Eltan.Com

Eltan information

- Founded early 1998
- Eltan is an independent engineering company with its core business x86
- Assist companies deploying PC- and Communication Technology in their products
- Phoenix Authorized Independent Developer / Distributor
- Coreboot developer since 2013
- Intel FSP fellow traveler

Eltan Technology

- Eltan provides specialized high-quality engineering services for hardware, Firmware/ Bios and operating systems as well as complete system solutions.
- Firmware/ BIOS: Eltan supplies either solutions based on Phoenix or coreboot as an Intel and AMD source code partner.
- Embedded Controller: Based on proven code and experience
- Hardware design: Custom motherboards
- System design: Full Custom Design
- Custom Development tools and Dediprog programming tools
- Windows embedded partner

What is [ORWL]

- Project owner: DesignShift
- Objective: build a very secure system as a desktop, based on open hardware and software by partnering with the best developers and builders.
- Main partners:
  - Eltan (for the firmware development and hardware consultancy)
  - Quanta (manufacturing)
  - Maxim (for the Secure Controller)
  - ST (for the keyfob)
  - Intel

[ORWL]

- [ORWL] animation

AWARDS etc.

POPULAR SCIENCE

Design SHIFT Wins Popular Science Award

October 18, 2017 – Design SHIFT, today announced that it has received a 2017 "Best of What's New" Award for its ORWL™ secure endpoint, from *Popular Science* magazine, recognizing it as one of the 100 Greatest Innovations of the Year.

View complete press release

WIRED

This Ultra-Secure PC Self Destructs if Someone Messes With It

June 23, 2017 (Wired) – A dedicated hacker can pwn your PC in like the offspring of a Mac Mini and a flying saucer, and it's nea code—of data snoops, botnet admins, or the Thought Police. S And for the paranoid, one of the software options is the securi

View complete press release

ORWL is Recognized by 2017 Edison Awards!

WHEN PROTECTION OF YOUR DATA IS CRITICAL

- Over the last few years, we saw recurring cyber attacks in various markets.
- Data stored are valuable and need ultimate security.
- Hardware is the building block of this security.

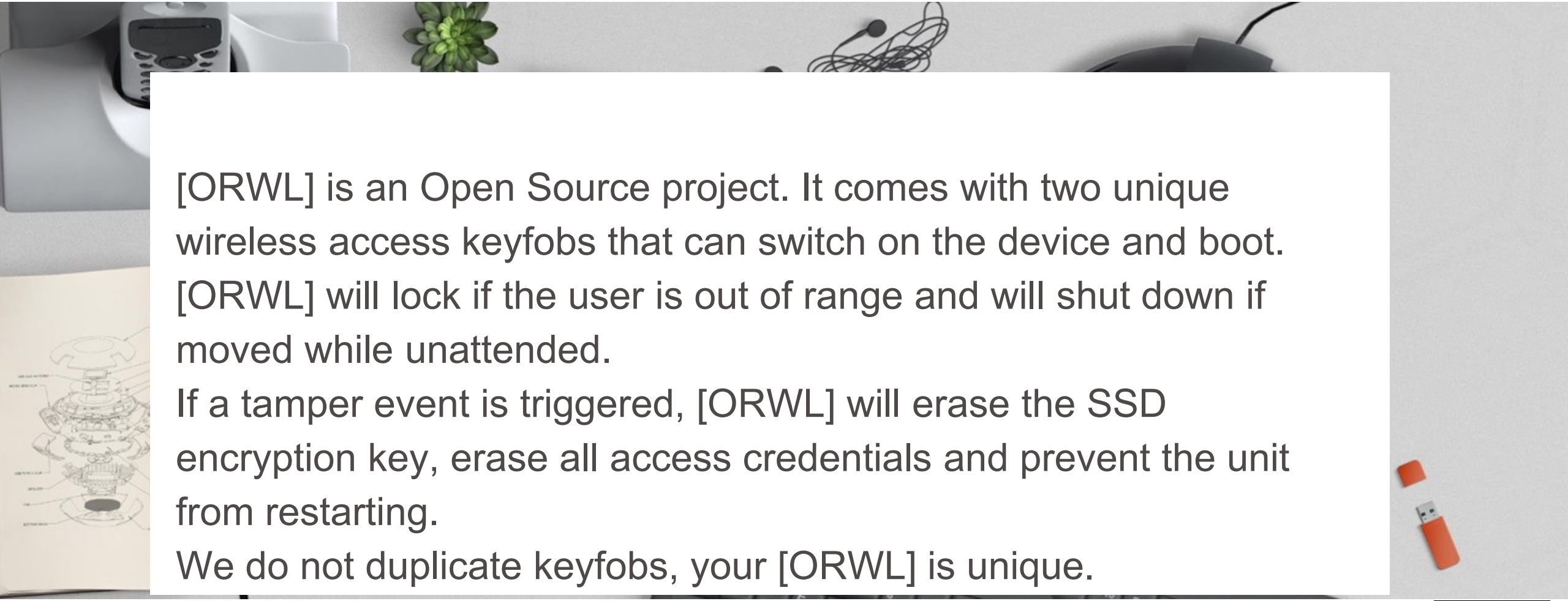
TYPES OF ATTACKS ENCOUNTERED

[ORWL] has been thought out to shield any types of physical attacks existing:

- Walk-in – Injection of a USB stick
- Sneak-in – Remote Access/ IP spoofing
- Break-in – opening the device
- Damage equipment – Altering the temperature, drilling holes...

SECURITY IS AS GOOD AS THE WEAKEST LINK IN THE CHAIN

A NEW CATEGORY OF SECURED SYSTEM



[ORWL] is an Open Source project. It comes with two unique wireless access keyfobs that can switch on the device and boot.
[ORWL] will lock if the user is out of range and will shut down if moved while unattended.
If a tamper event is triggered, [ORWL] will erase the SSD encryption key, erase all access credentials and prevent the unit from restarting.
We do not duplicate keyfobs, your [ORWL] is unique.

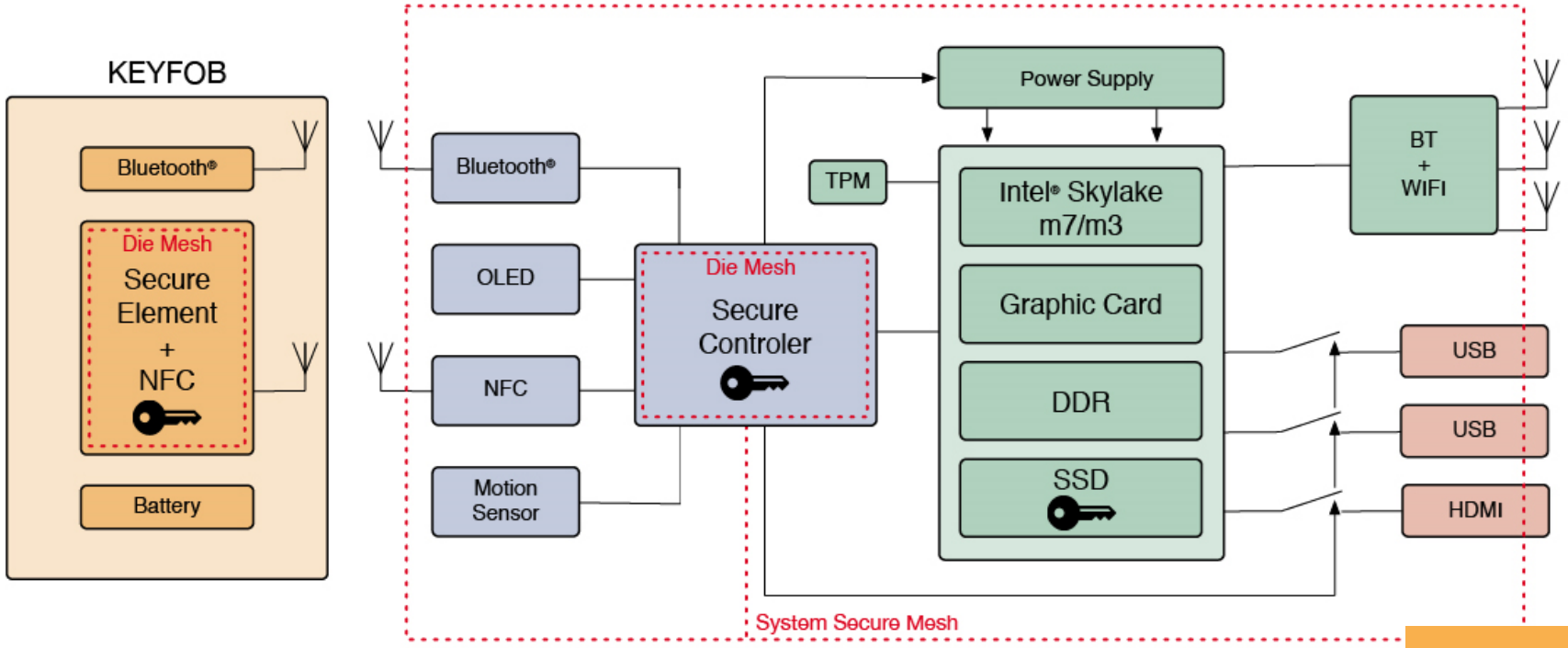
SMALL + SECURE

[ORWL]

[ORWL] is a new category of secured consumer device designed using banking technologies to protect your data. It requires both a key and a password to be used. [ORWL] will replace any computer and can also work as a secure endpoint to control access to your cloud services, as part of a secure network.

SECURITY IS AS GOOD AS THE WEAKEST LINK IN THE CHAIN

SYSTEM ARCHITECTURE SUPPORTS EXISTING STANDARDS



A POWERFUL PC

[ORWL] is a secured system running the latest X86 Intel® processor Core™ m .

[ORWL] supports Bare Metal virtualization, thanks to IOMMU VT-d.

YOU CAN RUN A VM-BASED OS.

ZONE OF TRUST

Guest devices

Trusted devices

Authentication

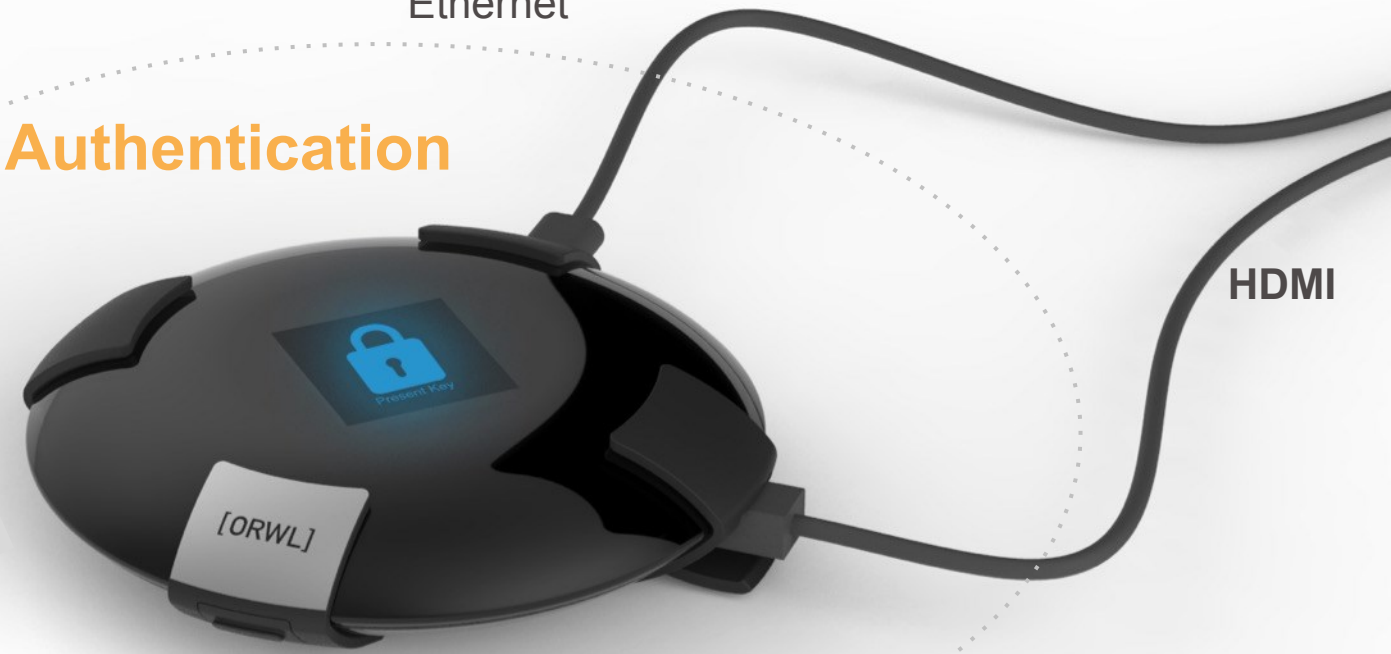
WiFi 802.11 ac
Access points
Secure WiFi pairing
IOT

BTLE / BT
Credential keys
User presence
BT/NFC secure pairing

NFC
Login
Root access

USB 3.0 type C
Keyboard/Mouse
Camera
Ethernet

HDMI



4 cm

10 m

100 m

[KEY]

[ORWL]

BTLE Beacon

NFC + Secure element

Access right / Presence control

Range < 10m

Hidden UUID
UUID encrypted
Synchronized timetable

USB
Power + Data

USB
Mouse keyboard

micro HDMI 1.4
4k Audio/Video

AES over NFC

Private/Public key pairing
Synchronized time table

WiFi 802.11AC 2x2 Dual band
BT4.0

USB 3.0

USB 3.0

HDMI

Intel 'Skylake' Core M5/M7

64-512GB SSD

2/4/8 GB LPDDR3 1600MHz

NFC Reader

Crypto Key manager with tamper sensor

Power control

White OLED

Motion sensor

Active Mesh Sensor

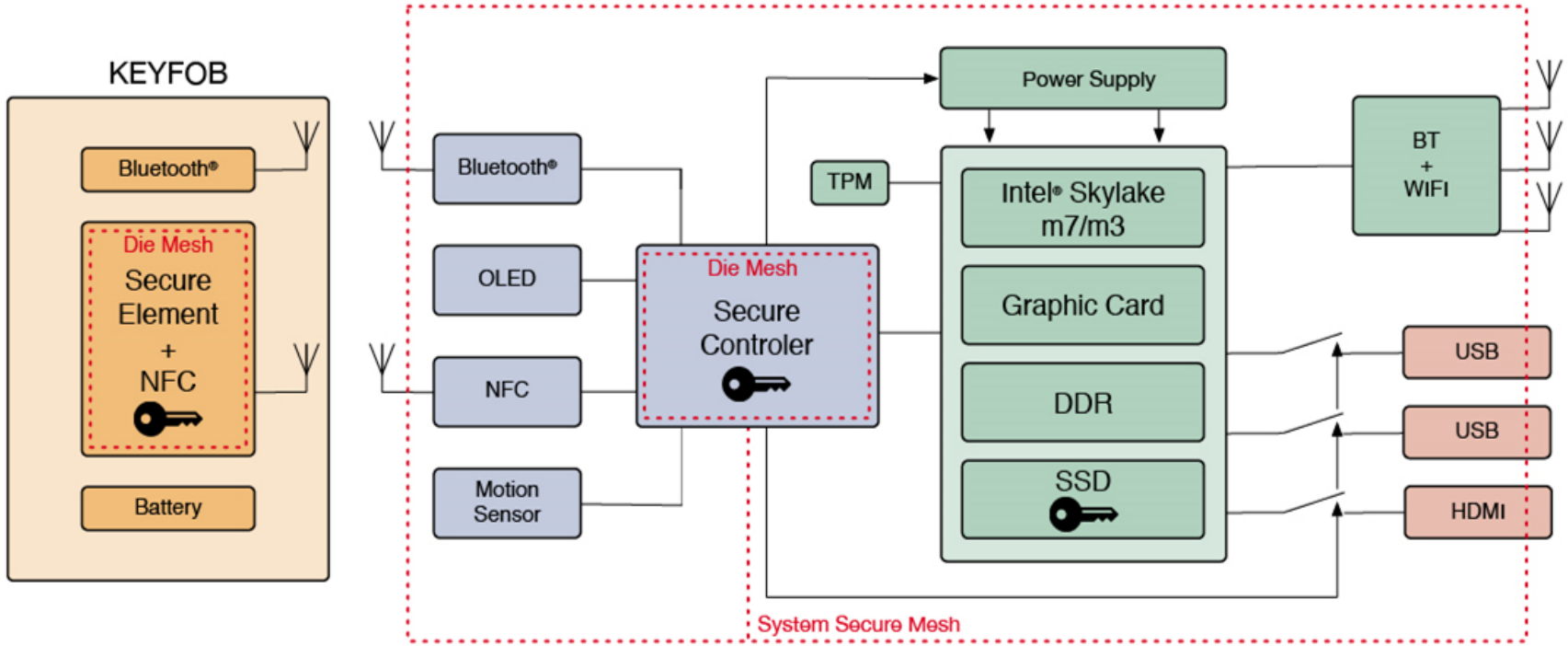
Architecture

- Open hardware and software

- Hardware based on Intel Skylake notebook reference
- Firmware Coreboot/ Tianocore UEFI payload
- Standard Operating systems Linux/ Windows/

- Mechanical

SYSTEM ARCHITECTURE SUPPORTS EXISTING STANDARDS



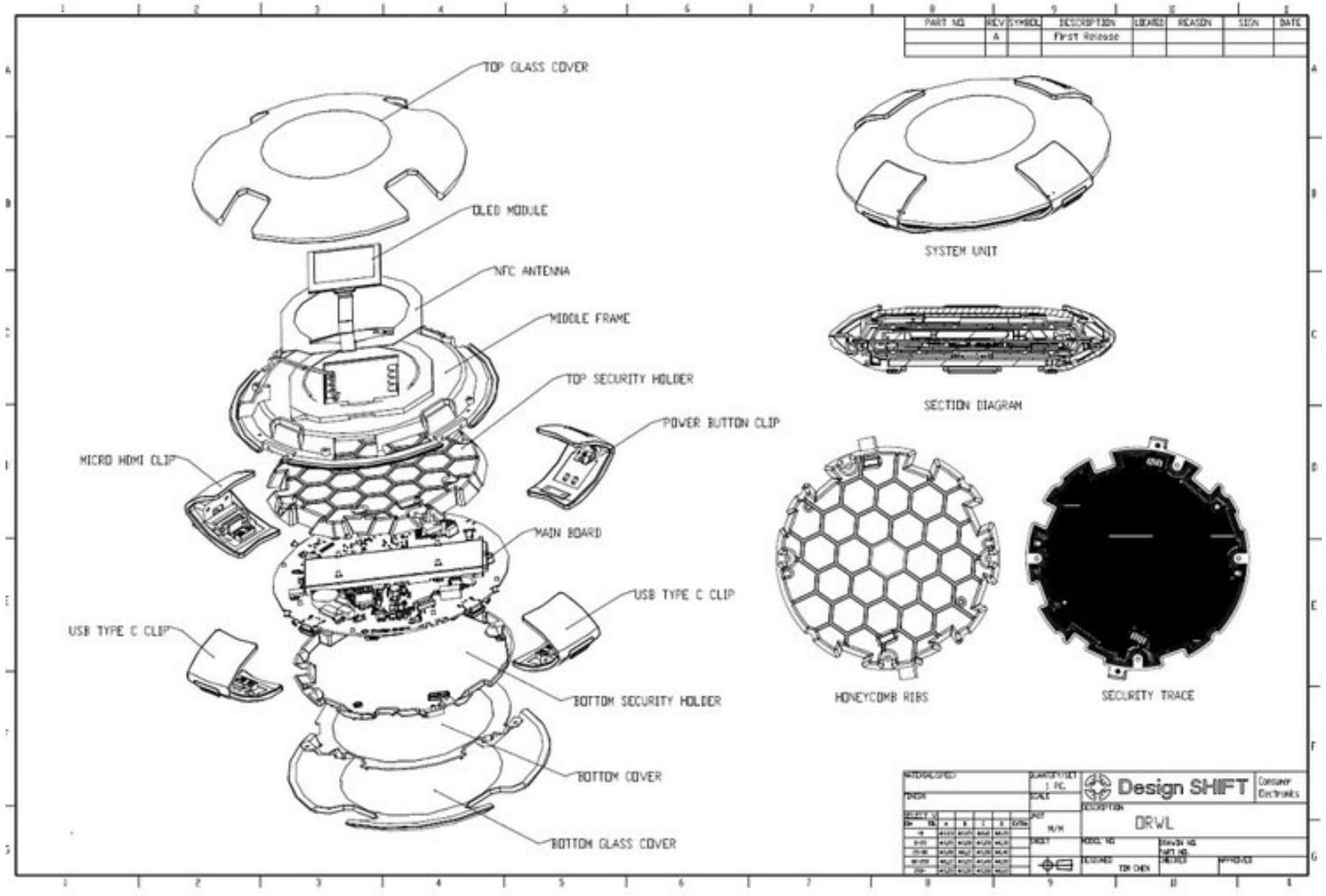
[ORWL] is a secured system running the latest X86 Intel® processor Core™ m .

[ORWL] supports Bare Metal virtualization, thanks to IOMMU VT-d.

YOU CAN RUN A VM-BASED OS.

A POWERFUL PC

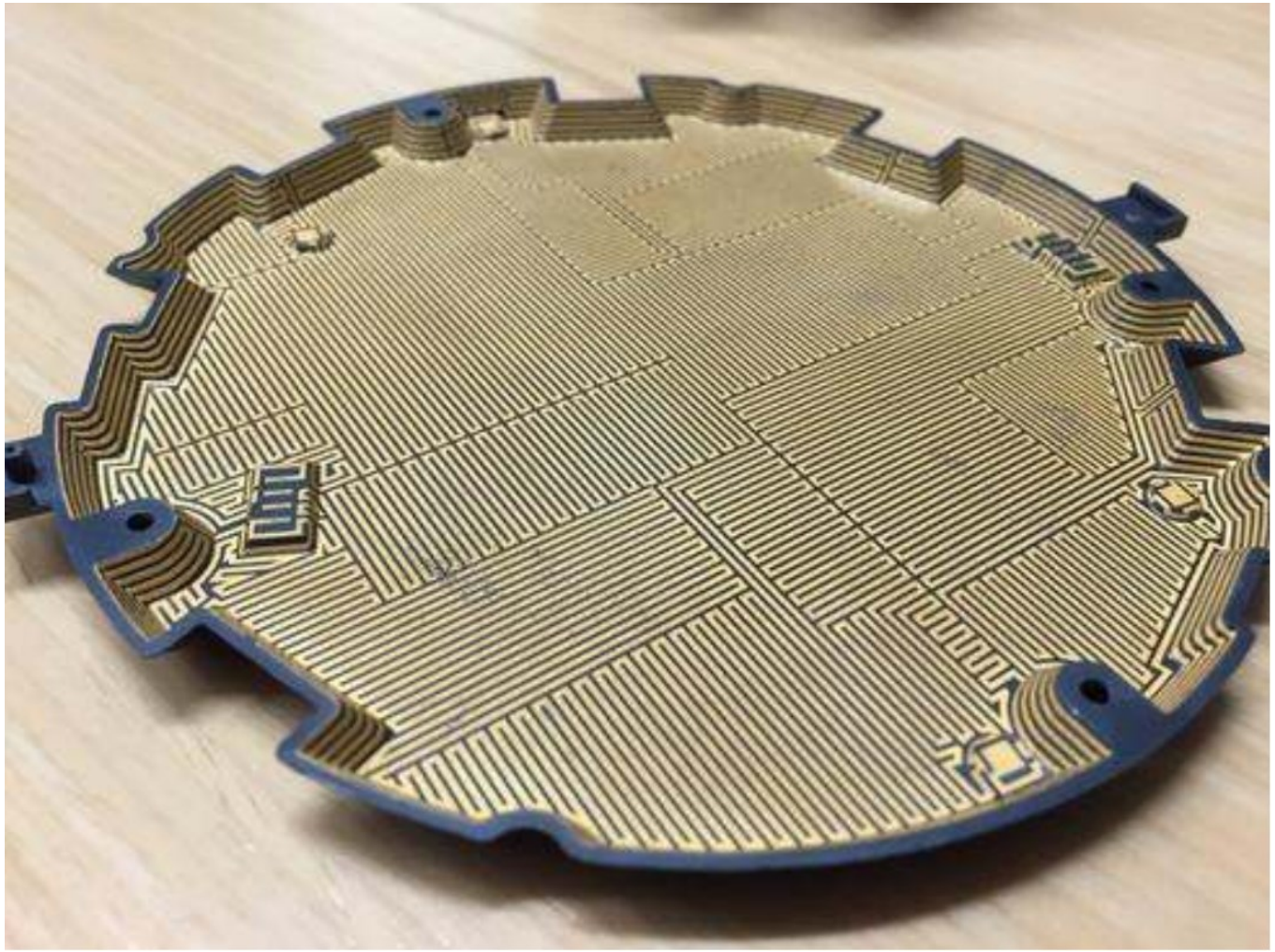
Mechanical



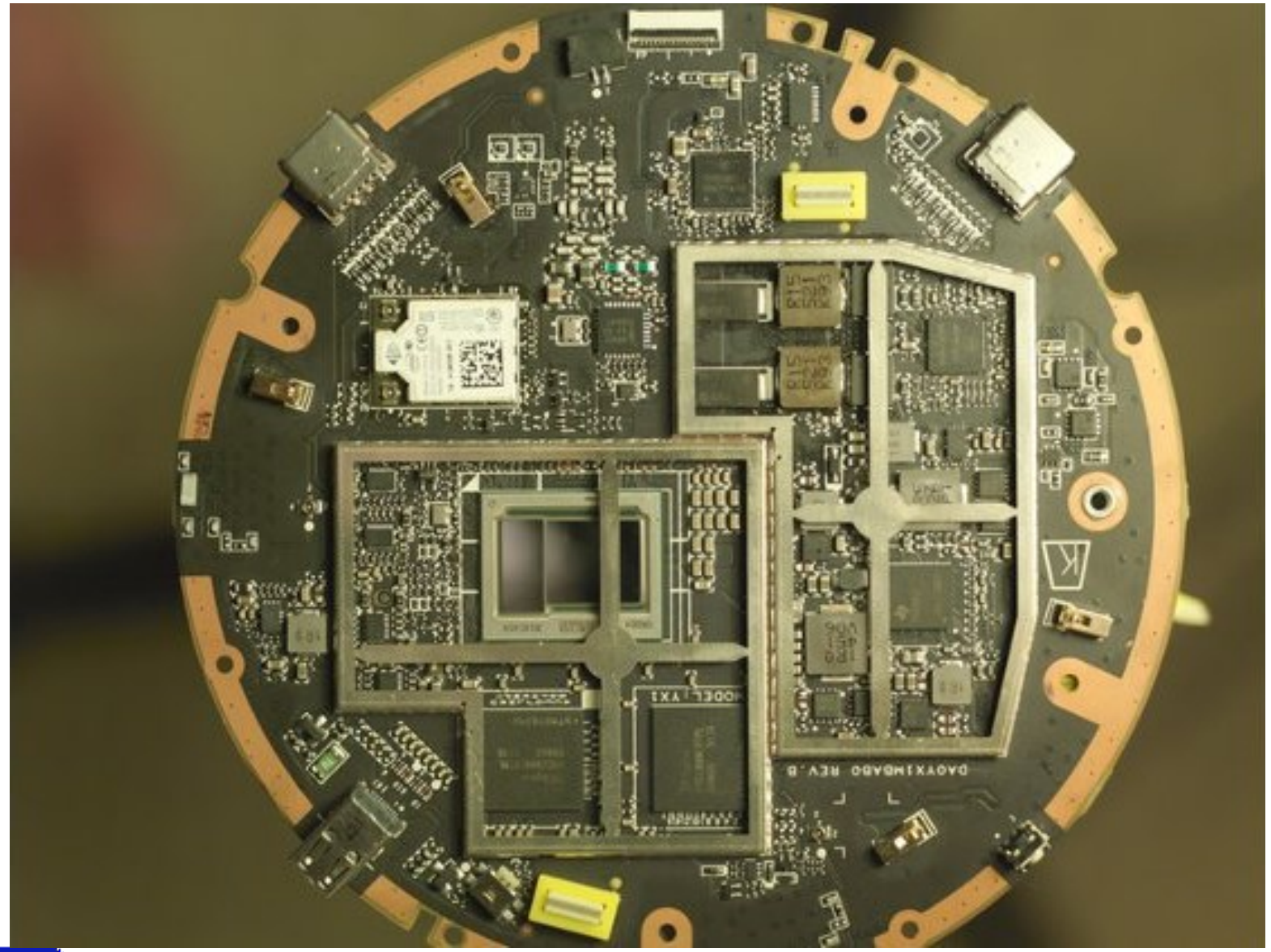
Exploded

PCA with mesh

The mesh inside



PCA

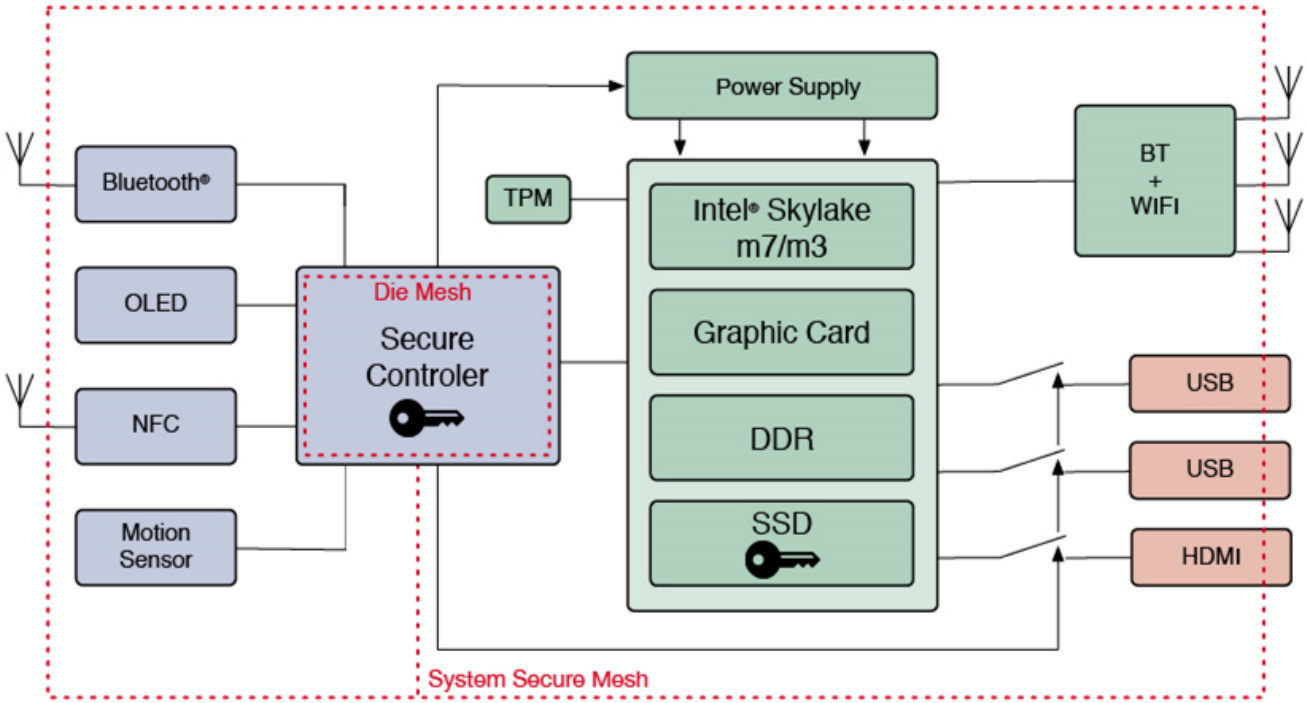


Secure Keyfob

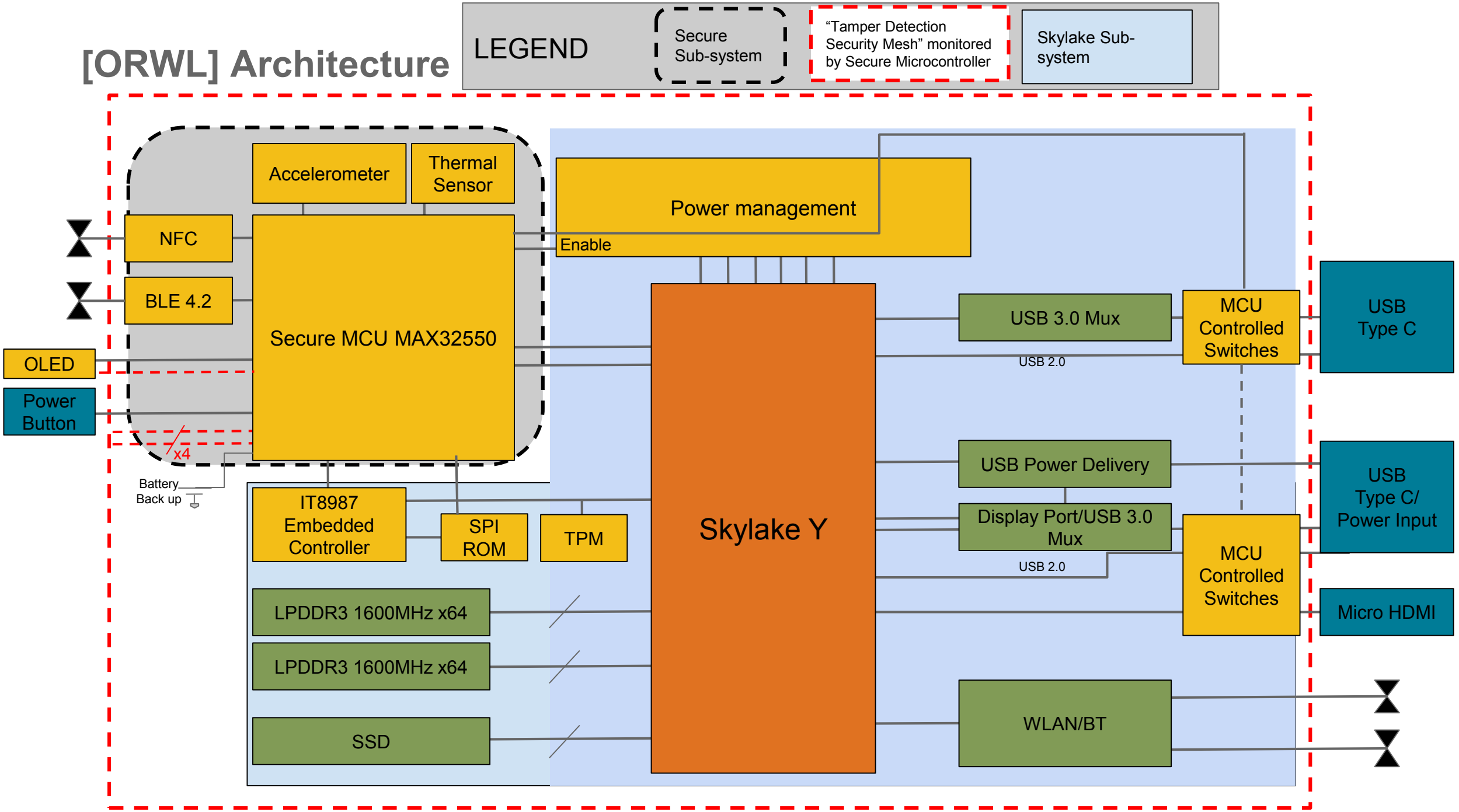
- Built around the STMicroelectronics ST54D
  - Runs in card emulation mode
  - Uses NFC to authenticate using the same authentication technology as smart cards
  - Adds BlueTooth (BLE) for proximity monitoring

The [ORWL] system

- Contains two main sub systems
- A Secure Controller (Maxim MAX32550)
- A PC based on the Intel Skylake m3 to m7 SoC

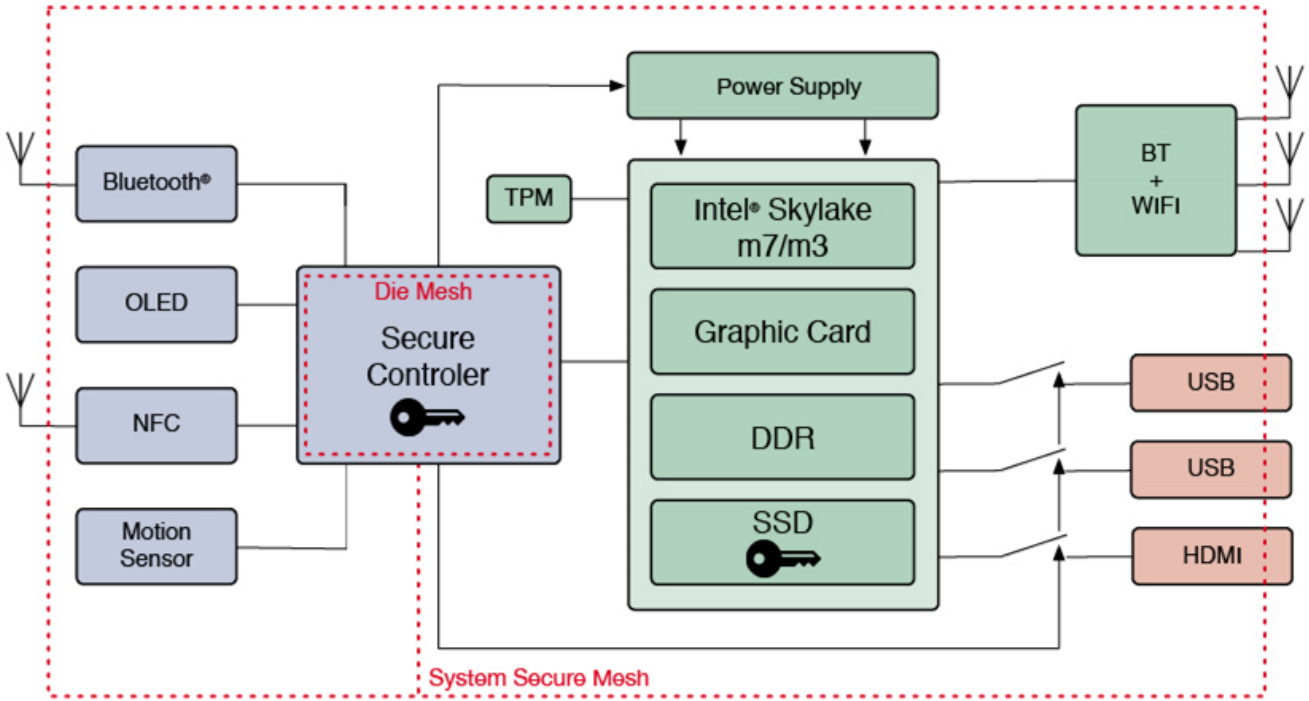


[ORWL] Architecture



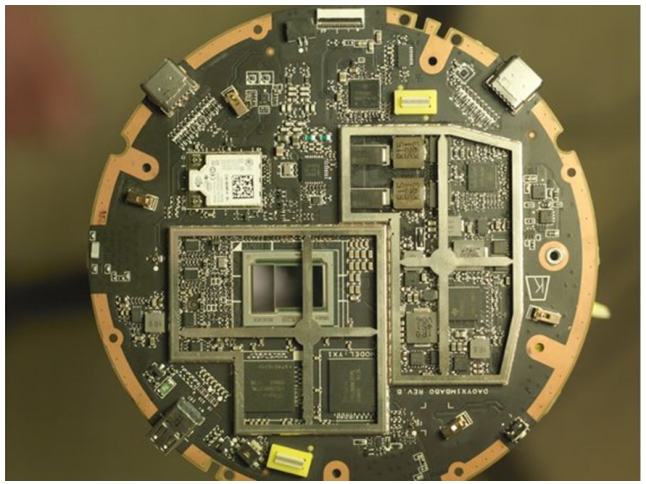
Secure Controller

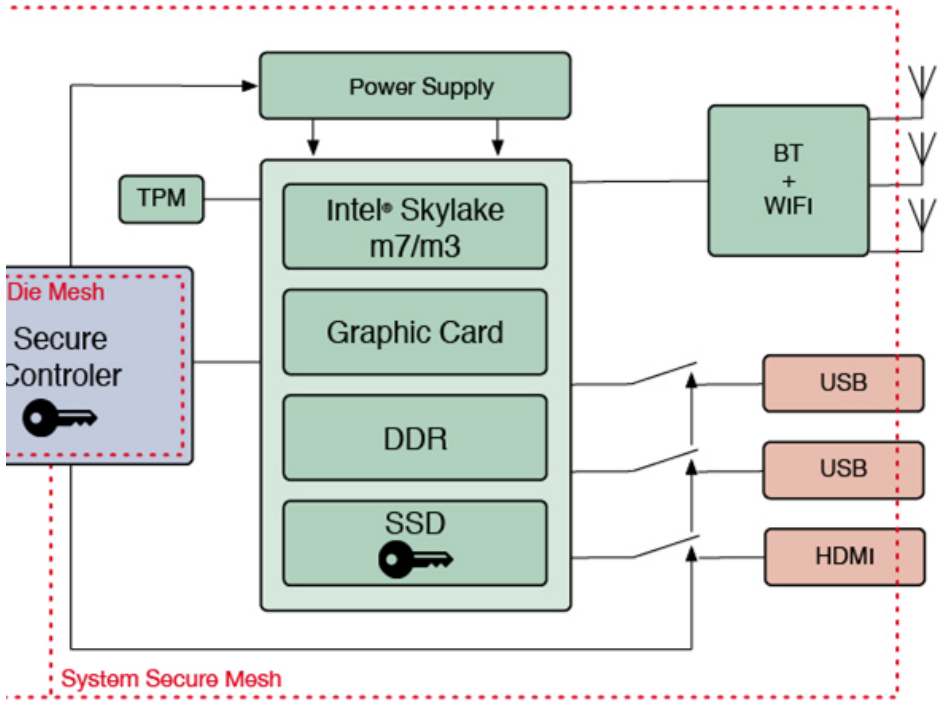
- Maxim 32550 running FreeRTOS
  - Acts as the center of the security solution
  - NFC for authentication
  - before the system can be turned on
  - BlueTooth for the proximity detection
  - Motion sensor to detect the system is moved
  - Monitors mesh integrity
  - Stores main product keys
  - Real time clock
  - Verifies flash integrity before switching on
  - Can disable USB and HDMI ports
  - Can request the system to enter sleep
  - Can force the system off
  - Uses OLED screen for UI
  - Bios can interact with this subsystem



PC subsystem

- Skylake Y based system supporting m3 to m7 processor
- Based on notebook reference design
- Interfaces to the outside using
- 2 * USB-C ports with power delivery
- 1 micro HDMI port for the display
- BlueTooth controller
- Wifi controller
- Self encrypting SSD
- 8 GB onboard memory

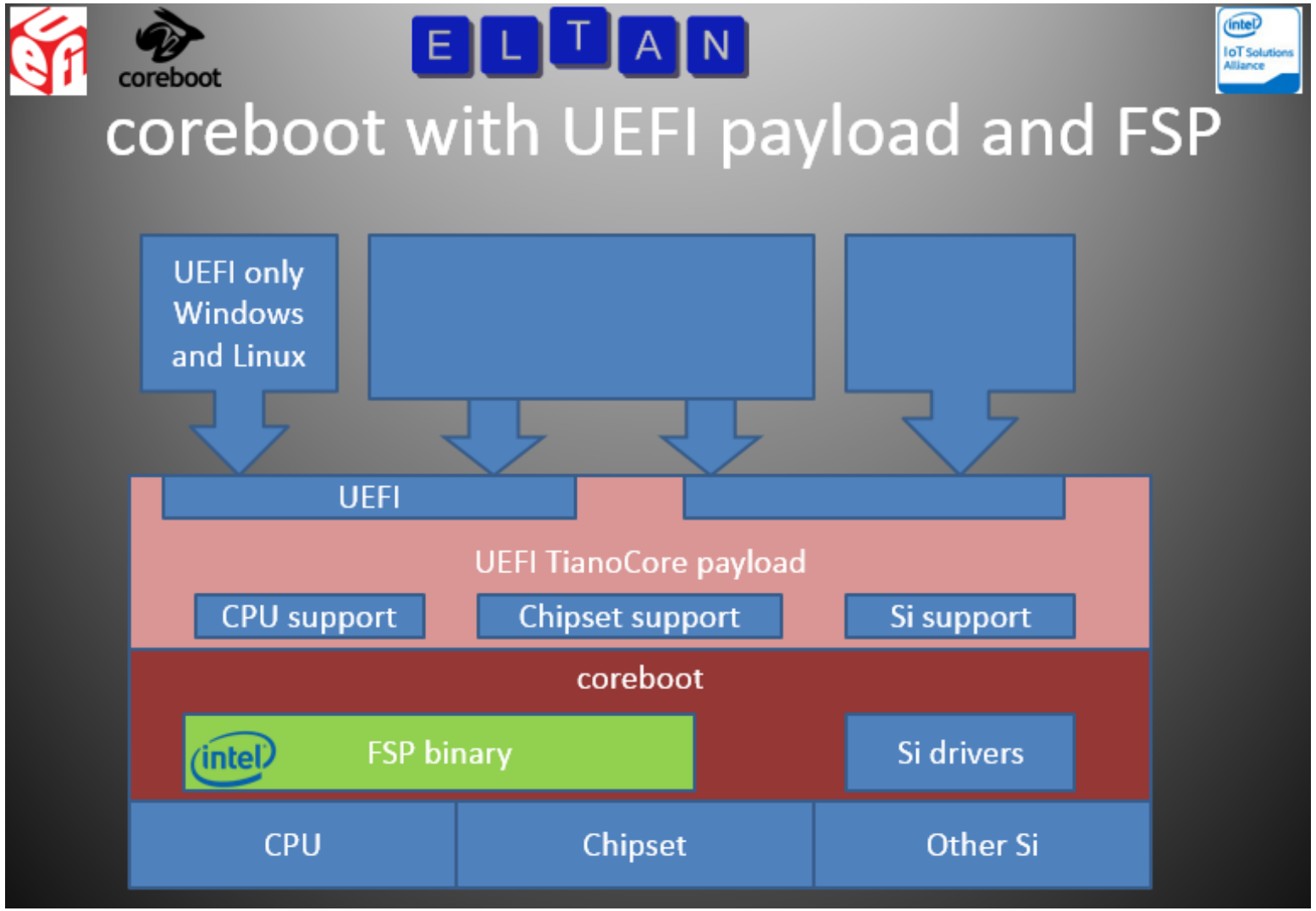




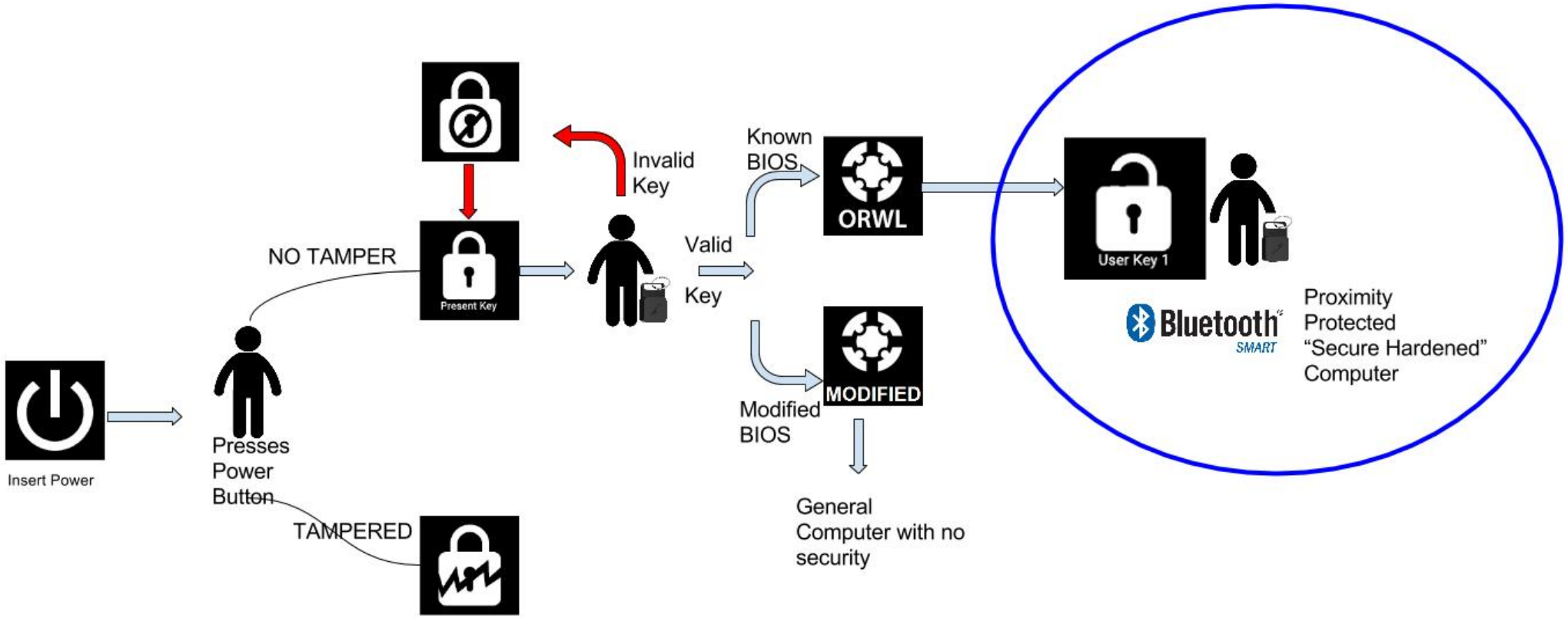
Firmware architecture

- Requirements:
- As much as possible Open Source solutions
- Should be able to run Ubuntu Linux, Qubes OS and Windows 10 using the same firmware
- Should use as little as possible of the Intel ME functionality
- Supports verified / secure boot
- A preboot UI to allow changing settings to the Secure Controller should be made available
- Because of this coreboot combined with a TianoCore Payload is used

Firmware architecture



Product Usage



[ORWL] features

To address the [ORWL] requirements the features have to be addressed in either the coreboot part, the tianocore payload or both.

System State
To allow a proper flow from PCB assembly to the actual use of the system as the end user a number of system states have been defined to make sure the system automatically behaves as expected in the defined situation.

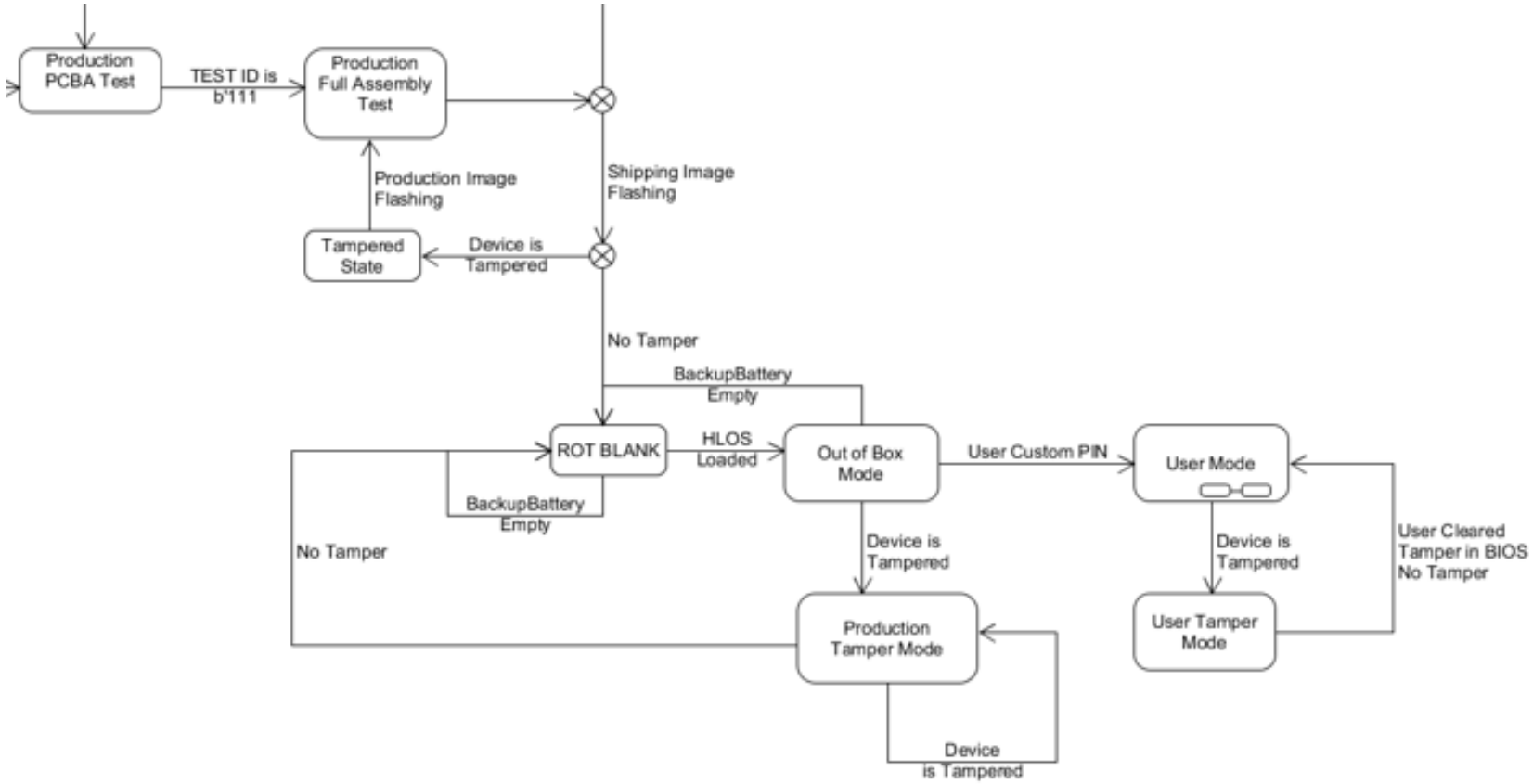
The main reason for this is that the Secure Controller initially only has the unprotected boot code and none of the keys that define it's personality. Further more the keyfobs will only be paired when the system arrives at the end user so this is when the user will be able to authenticate for the first time.

Depending on the state the system will be able to boot from USB only or from SATA only. Also the secure boot will be enforced or not. Flash update will be enabled or not etc.

Additionally the system can start in a tampered state in this case the all that can be done is display the reason for the tampering and the system will shutdown.

[ORWL] features

System State



[ORWL] features

USB booting
When the system is in the user state it will not be able to boot from USB during production state this is the only boot option. The user can enter SETUP to request a USB boot. When this is requested a USB boot can be performed once after that it will be disabled again.

This functionality is implemented in TianoCore payload. We modified the USB driver to make sure no block device driver is loaded on a USB device unless the system is in production state or the USB boot is enabled. This method also prevents booting from a USB device from the UEFI shell.

Flash Update
To perform the flash update we decided to use flashrom to make use of this convenient we decided to create a small Linux distro packed in a self contained efi executable to perform this update. The user can request the system to enter flashmode. When this is done system will reboot with unprotected flash and will only be able to boot this executable which is signed with a dedicated key.

The TianoCore boot- and security policy have been adapted to support this mechanism. Coreboot has been adapted to leave the flash unprotected in flash state.

[ORWL] features

Verified boot
The verified boot implementation is implemented chain that starts at the Secure Controller. This validates an initial part of the coreboot image. The coreboot image then validates the additional stages, payload and other relevant data areas. A manifest that is used for that is generated during build time. Once the TianoCore payload is started this implements secure boot and allows booting a trusted bootloader.

The coreboot part of this verified boot is fixed and can't be disabled. The user may decide to disable or modify secureboot for the TianoCore part. This is similar to a standard UEFI bios implementation. The policy has been customized to support the specific behavior required in e.g. production and flash state.

Tamper protection
The secure controller stores the system secrets and will discard these when the system is tampered. When this happens all data on the system will be discarded as well and can't be recovered. Also the system will enter a dedicated boot path that only allows notifying the user of the event in a user friendly way.

[ORWL] features

Secured SSD

The [ORWL] uses a self encrypting SSD and stores the data used to unlock the disk in the Secure Controller. This data is not stored anywhere else not even to allow unlocking of the disk when resuming from S3. This has been done to fully support the functionality that will cause the system to lock the disk when the system has been tampered.

To support this coreboot interacts with the Secure Controller very early to support S3 and TianoCore Payload does this as late as possible to allow booting. To recover an SSD from a somehow failing system it is possible to erase the device and thereby unlocking it.

The user can also erase the disk from SETUP and request the Secure Controller to generate new security information. The result will be a clean system. Normally it is possible to restart into SETUP from the OS. We noticed that doing this and trying to perform a secure erase to the SSD will fail as the Microsoft drivers issue the Freeze Lock command that prevents use of this command until a power cycle of the drive is performed.

Disable Wifi

A user can disable Wifi completely from SETUP as well. If this is done the PCIe lanes the Wifi module is connected to will be disabled so the device will not be visible to the OS or any other software. This is handled in coreboot.

[ORWL] features

Proximity and motion protection
The [ORWL] supports a proximity and motion protection feature. In SETUP the user can decide if this should be enabled and how this should behave. This includes items like the distance between the [ORWL] and keyfob that is allowed before the protection becomes activated. This can be between 5 and 30 meters. The user can also define the action that should be taken such as system should enter S3 or system should shutdown. The user can also indicate the USB should be disconnected or the screen should be blanked.

The Secure Controller deals with the actual detection and will perform the notification to the system to perform actions. Normally an attempt will be made to perform these actions gracefully. The Secure Controller will generate ACPI events and the ASL code contains the handlers to deal with that. If this somehow doesn't lead to the expected result the Secure Controller is able to intervene and e.g. turn the system of by removing the power.

[ORWL] features

Prevent manipulating of the time
Coreboot will retrieve the system time from the Secure Controller each time it boots. The runtime UEFI API to set the time is also adapted to update the time in the secure controller as well as in the PC RTC. As even UEFI operating systems don't use the UEFI API all the time coreboot will also synchronize the time to Secure Controller when the system shuts down.

The mechanism allows sanity checking within the Secure Controller and discarding a change if this seems out of bounds.

Authentication
The user authenticates using the keyfob, this can be done with and without entering a pin. User can decide if entering a pin to start the system is required or not for other items this is pre-defined.

The authentication process itself is completely handled by the Secure Controller and coreboot is not aware of the authentication that happens before the system is started. For other situations like the authentication before entering SETUP or the UEFI shell the TianoCore payload will initiate the authentication use the result.

Coreboot implementation

The coreboot implementation used is mainly structured as any other coreboot port but there are specific items that needed to be addressed.

Related to the Skylake Y support
Coreboot for the Skylake uses the Intel FSP as the main component for the hardware initialization
As the FSP is released by the Intel IOTG group the reference implementation provided by Intel doesn't support Skylake Y. To be able to create coreboot for this system anyhow we addressed the differences between the provided Skylake H and the Skylake Y. The main differences between the two are in the stripped down PCH which results in differences in GPIO mappings differences in PCIe and USB lanes etc. To make the HDMI audio support working we also needed to add the verb tables for that.

We had to create a modified version of the FSP to support the Skylake Y combined with a bios area that requires more than one MTRR for caching.

Secure SSD
Ported the libpayload ahci driver for use in coreboot so this can be used to enable the SSD in the resume from S3 path. We ported this to the extend required to serve our purpose

Coreboot implementation

Secure Controller Communication
Of course we needed to be able to communicate with the Secure Controller in coreboot we needed this in ramstage and in SMM. The Secure Controller is connected to the PC subsystem using a serial port. We use one of the serial ports contained in the PCH for this. As we needed to implement the communication protocol in TianoCore so we had this code implemented we create a small wrapper that allows us to take that code and use it in coreboot so there was no need to implement this twice and we can synchronize easier between the two code bases.

Synchronizing system state
Coreboot will check the Secure Controller state on various places e.g. before unlocking the SSD to make sure the correct actions are taken. Besides that it will update the Secure Controller to make sure this is aware of the state the system is in. It does this before entering SO, S3, S4 or S5.

Preventing possible leakage issues
As the Secure Controller is always active and the PC system is not this required additional attention to prevent leakage issues when the system is in S3 or S4/S5 configuration of the SoC pins is crucial and all details count.

Coreboot implementation

Related to very dense design
Tuning the power supply parameters to make sure the system doesn't exceed the power budget as it is not possible to create a system that allows all power supplies to draw full power

Tuning of the Turbo mechanism to create good performance while still maintaining reasonable power average consumption and limiting the peak power

Qubes
Qubes installation was causing problems as Qubes assumes that a system is running SeaBIOS when the BIOS vendor in SMBIOS reports "coreboot" we fixed this by reporting something else.

We needed to update the BDS handling in TianoCore to allow booting of the install media created by Qubes. The media contains both a CD and a Hard disk (MBR) partition and both appear to contain a valid bootloader. Unfortunately the grub contained in the CD partition doesn't chainload XEN successfully. We changed the order here so the system will first check the HD partition.

Ubuntu
We noticed that Ubuntu doesn't set the "weekday" to a valid value, this caused our rtc mechanism to reject the updated time when running Ubuntu.

Operating systems

[ORWL] comes with either Windows 10, UBUNTU or Qubes OS pre-installed

Windows and Ubuntu can be installed from a standard installation medium without problems

As indicated we have solved some items to make the Qubes OS 3.2 installation working besides that Qubes decided not to sign the EFI binaries they only sign the medium and the packages. So this will not work with secure boot enabled as is.

Q&A

Contact us

Gerard Duynisveld
Gduynisveld at Eltan dot com

Wim Vervoorn
Wvervoorn at Eltan dot com

WWW.Eltan.Com