Trustworthy Platform Module

An attempt to create open-source firmware for TPM

FOSDEM 2023

Maciej Pijanowski
Maciej Pijanowski
Engineering Manager

- 7 years in 3mdeb
- Open-source contributor
- Interested in:
  - build systems (e.g. Yocto)
  - embedded, OSS, OSF
  - firmware/OS security

- @macpijan
- maciej.pijanowski@3mdeb.com
- linkedin.com/in/maciej-pijanowski-9868ab120
Who we are?

- Poland-based company, over 7 years in the market
- Open-source firmware, Embedded Linux
- coreboot licensed service providers since 2016 and leadership participants
- UEFI Adopters since 2018
- Yocto Participants and Embedded Linux experts since 2019
- Official consultants for Linux Foundation fwupd/LVFS project since 2020
• What is TwPM project?
• Why did it start?
• TPM modules pinouts
• How to start such a project?
• Expected challenges
• Roadmap
• Current state
• Q&A
TwPM project aims to increase the trustworthiness of the TPM module (hence the TwPM), by providing the open-source firmware implementation for the TPM device, compliant* to the TCG PC Client Specification.

TPM modules enable measured boot and support verified boot, Dynamic Root of Trust for Measurement, and other security features.

The project is funded through the NGI Assure Fund, a fund established by NLnet foundation.

https://nlnet.nl/project/TwPM
Traditional TPMs are dedicated microcontrollers with proprietary firmware
  - can't be audited
  - bugs can't be fixed if TPM vendor doesn't care
  - capabilities of TPM are defined by the vendor and can't be modified by user (e.g. to include newer hash algorithms)

Different interfaces
  - LPC (older PCs, still commonly used)
  - SPI (new PCs, mobile, IoT)
  - I2C (mobile, IoT)

Each mainboard vendor has different pinout for module
  - some look the same and mechanically can be installed to incompatible boards, but their electrical connections are different
  - doing so may **physically damage your mainboard**
### Examples of TPM pinouts

**MSI - B75MA-E33 (14-1)**

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Definition</th>
<th>Pin No.</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LPC Clock</td>
<td>2</td>
<td>3V Standby Pwr</td>
</tr>
<tr>
<td>3</td>
<td>LPC Reset</td>
<td>4</td>
<td>3.3V Power</td>
</tr>
<tr>
<td>5</td>
<td>LPC address &amp; data pin 0</td>
<td>6</td>
<td>Serial IRQ</td>
</tr>
<tr>
<td>7</td>
<td>LPC address &amp; data pin 1</td>
<td>8</td>
<td>5V Power</td>
</tr>
<tr>
<td>9</td>
<td>LPC address &amp; data pin 2</td>
<td>10</td>
<td>No Pin</td>
</tr>
<tr>
<td>11</td>
<td>LPC address &amp; data pin 3</td>
<td>12</td>
<td>Ground</td>
</tr>
<tr>
<td>13</td>
<td>LPC Frame</td>
<td>14</td>
<td>Ground</td>
</tr>
</tbody>
</table>

**Asus - MAXIMUS IX FORMULA (14-1)**

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Definition</th>
<th>Pin No.</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>F_LAD0</td>
<td>2</td>
<td>+3V</td>
</tr>
<tr>
<td>3</td>
<td>F_LAD1</td>
<td>4</td>
<td>+3V</td>
</tr>
<tr>
<td>5</td>
<td>F_LAD2</td>
<td>6</td>
<td>C_PCIECLK_TPM</td>
</tr>
<tr>
<td>7</td>
<td>F_LAD3</td>
<td>8</td>
<td>GND</td>
</tr>
<tr>
<td>9</td>
<td>F_FRAME#</td>
<td>10</td>
<td>No Pin</td>
</tr>
<tr>
<td>11</td>
<td>F_SERIRQ</td>
<td>12</td>
<td>S_PCIEIRST#_TBD</td>
</tr>
<tr>
<td>13</td>
<td>F_CLKRUN</td>
<td>14</td>
<td>+3VSB</td>
</tr>
</tbody>
</table>
**Examples of TPM pinouts**

### GIGABYTE Z590 AORUS MASTER (12-1)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Definition</th>
<th>Pin No.</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Data output</td>
<td>2</td>
<td>Power 3.3V</td>
</tr>
<tr>
<td>3</td>
<td>No Pin</td>
<td>4</td>
<td>NC</td>
</tr>
<tr>
<td>5</td>
<td>Data Input</td>
<td>6</td>
<td>CLK</td>
</tr>
<tr>
<td>7</td>
<td>Chip Select</td>
<td>8</td>
<td>GND</td>
</tr>
<tr>
<td>9</td>
<td>IRQ</td>
<td>10</td>
<td>NC</td>
</tr>
<tr>
<td>11</td>
<td>NC</td>
<td>12</td>
<td>RST</td>
</tr>
</tbody>
</table>

### MSI - Z590 PRO WIFI (12-1)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Definition</th>
<th>Pin No.</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SPI Power</td>
<td>2</td>
<td>SPI Chip Select</td>
</tr>
<tr>
<td>3</td>
<td>Master In Slave Out (SPI Data)</td>
<td>4</td>
<td>Slave Out Slave In (SPI Data)</td>
</tr>
<tr>
<td>5</td>
<td>Reserved</td>
<td>6</td>
<td>SPI Clock</td>
</tr>
<tr>
<td>7</td>
<td>Ground</td>
<td>8</td>
<td>SPI Reset</td>
</tr>
<tr>
<td>9</td>
<td>Reserved</td>
<td>10</td>
<td>No Pin</td>
</tr>
<tr>
<td>11</td>
<td>Reserved</td>
<td>12</td>
<td>Interrupt request</td>
</tr>
</tbody>
</table>
Examples of TPM pinouts

... and many more!
At first we wanted to choose most commonly used connector and use it with custom board
  - variety is bigger than anticipated
  - connector would have to support different interfaces (LPC, SPI)
We may aim to tackle the hardware problem in the future
  - firmware is enough of a challenge for a start

Examples of TPM pinouts

Image source: https://xkcd.com/927/
• There are **a few** open-source stacks for processing TPM commands
  • ms-tpm-20-ref
    ◦ [https://github.com/microsoft/ms-tpm-20-ref](https://github.com/microsoft/ms-tpm-20-ref)
    ◦ implementation from Microsoft
    ◦ simulator for Windows / Linux / MacOS
    ◦ some others
      ◦ fTPM Trusted Application for ARM Trust Zone
      ◦ samples for STM32 **Nucleo L476RG / L4A6RG**
  • ibmswtpm2
    ◦ [https://sourceforge.net/projects/ibmswtpm2/](https://sourceforge.net/projects/ibmswtpm2/)
    ◦ implementation from IBM
    ◦ simulator
Code for Nucleo samples was contributed 4 years ago
- It was created using the Atollic TRUEStudio for STM32
  - such software no longer exists
  - STM32CubeIDE has replaced it
- Code under directory was contributed at some point in the past
  - it is not maintained - it may or may not work
  - [https://github.com/microsoft/ms-tpm-20-ref/issues/62](https://github.com/microsoft/ms-tpm-20-ref/issues/62)
We have converted the project into STM32CubeIDE
We were able to build it after some modifications
There is some VCOM application for Windows
  - https://github.com/microsoft/ms-tpm-20-ref/tree/main/Samples/Nucleo-TPM/VCOM
    - it was used for testing this sample code
The STM32 code can accept TPM command via USB CDC
  - it can process it
  - it can return response
There is some custom protocol involved there
  - no interoperability with existing tools, such as tpm2-tools
  - no interoperability with existing TPM interfaces (e.g. SPI, LPC, I2C)
The STM32 was low on resources when running this code
• In the meantime, the global shortage of chips happened
  ○ even if we wanted to use STM32L4, they were not available
  ○ the other chips, were also at low-availability
• It was difficult to assess the precise hardware requirements at this point
• The application was using HAL, so switching hardware requires rewrite
• It would be nice to use some OS to switch between boards more easily
• We chose Zephyr as OS for TwPM
  ○ all-rounded RTOS with decent portability between smaller devices
Some TPM registers must return valid state without delay
  - may be hard or impossible to reply on time if it has to pass through interface FIFO and possibly kernel/userspace boundary
  - FPGA would help with returning register values on time, but we're experimenting with other, cheaper options

FPGA will be required for LPC protocol - LPC isn't supported by most MCUs

Reference implementation [1] doesn't implement NV RAM in a secure way
  - only TPM emulator is officially supported, it doesn't have physical flash so no protections implemented
  - wear leveling also has to be considered

Full compliance with TPM specification may be impossible
  - strict initialization time and power consumption requirements
  - no vendor ID assigned

1: https://github.com/microsoft/ms-tpm-20-ref
1. Public site with documentation:

https://twpm.dasharo.com/

**Roadmap**

TwPM project is funded by the NLnet Foundation via the NGI ASSURE.

3mdeb has proposed to implement the following tasks under the grant agreement No 957073.

1. Public site with documentation

All of the documentation produced during this project should be publicly available to users and developers.

Milestones:

- create repositories for the project
- prepare server and domain name
- describe project’s purpose and phases
- create placeholders for description of deviations from TPM specification, software stack and documentation changelog that will be updated after each of the remaining phases

Table of contents

1. Public site with documentation
2. Gather hardware requirements and choose target board
3. Implement LPC protocol in FPGA
4. Implement basic TPM registers in FPGA
5. Implement TPM command parsing and communication between FPGA and MCU
6. Base tests
7. Implement SPI TPM protocol
8. Explore the usage of using simpler hardware platform
9. Flash driver for TPM stack
10. Unique identification and randomness source
11. Manufacturing process
12. Customizable configuration
2. Gather hardware requirements and choose target board
2. Gather hardware requirements and choose target board
3. Implement LPC protocol in FPGA
   - currently in progress
   - targeted for PC only

4. Implement basic TPM registers in FPGA
   - some registers must be readable without delay
   - implementing TPM locality state machine in FPGA significantly simplifies interface between FPGA and MCU

5. Implement TPM command parsing and communication between FPGA and MCU
   - completing this step will produce first semi-usable version with limited capabilities
   - applications requiring persistent storage like sealing data to PCR values or persistent key creation won't be possible yet
Roadmap and current state
6. Base tests
   - test suites mostly documented: [https://github.com/Dasharo/docs/pull/447](https://github.com/Dasharo/docs/pull/447)
   - automation in progress
   - tests results will be added to documentation

7. Implement SPI TPM protocol
   - repetition of steps from previous slide for another interface
   - it is likely, that again FPGA might be required to meet timing requirements

8. Explore the usage of using simpler hardware platform
   - it may or may not be possible to use board without FPGA (usually cheaper)
   - potential benefits make exploration worthwhile
   - already happens in parallel to other tasks
9. Flash driver for TPM stack

- nonvolatile storage for user- and vendor-defined data
- will open the way for additional use cases
- more test suites
- protections are required for compliance with specification, but at this point we put it in nice-to-have category

10. Unique identification and randomness source

- implementation may depend on chosen hardware, hence left for later
- uniqueness required for primary seeds, used to generate primary keys
- primary seed is required to have at least twice the number of bits as the security strength of any symmetric or asymmetric algorithm implemented on the TPM
- TPM should have at least one internal source of entropy
- FPGA can be used if everything else fails
11. Manufacturing process

- each TPM has unique Endorsement Key (EK)
- vendor issues certificate for EK that should be committed to NVRAM
- this step will describe the process in detail and try to automate it

12. Customizable configuration

- prepare easy to use build system integrating whole stack
- build-time configuration including:
  - interface (SPI or LPC)
  - hash algorithms supported by TPM
  - amount of NVRAM
  - whether to include RNG entropy source from FPGA or not
- goal: making transition between different boards easier
Work currently in progress:

- [https://github.com/3mdeb/verilog-lpc-module](https://github.com/3mdeb/verilog-lpc-module)
  - LPC module
  - TPM registers probably will also be implemented in this repo
- [https://github.com/3mdeb/zephyr](https://github.com/3mdeb/zephyr)
  - exploration and abusing of SPI drivers takes place here
- [https://github.com/Dasharo/twpm-docs](https://github.com/Dasharo/twpm-docs)
  - source for [https://twpm.dasharo.com](https://twpm.dasharo.com)
  - will be progressively filled with results of each step

Interested about further development? Want to participate in the project? Join TwPM channel in Dasharo Matrix space:
Interested about further development? Want to participate in the project? Join TwPM channel in Dasharo Matrix space:

[https://matrix.to/#/##twpm:matrix.org](https://matrix.to/#/##twpm:matrix.org)
Want to join our team and work with open-source firmware on a daily basis?

- use contact links from next slide
- approach me directly

We need you
• **Join Dasharo Matrix space:** https://matrix.to/#/#dasharo:matrix.org

• 📧 **contact@3mdeb.com**

• 📡 **facebook.com/3mdeb**

• 🎈 **@3mdeb_com**

• https://fosstodon.org/@3mdeb

• 🟢 **linkedin.com/company/3mdeb**

• [https://3mdeb.com](https://3mdeb.com)

• **Book a call**

• **Sign up for the newsletter**
Q&A