System Management
Interface Forum

Ericsson's config file format and loading method for PMBus devices

Björn Olsson

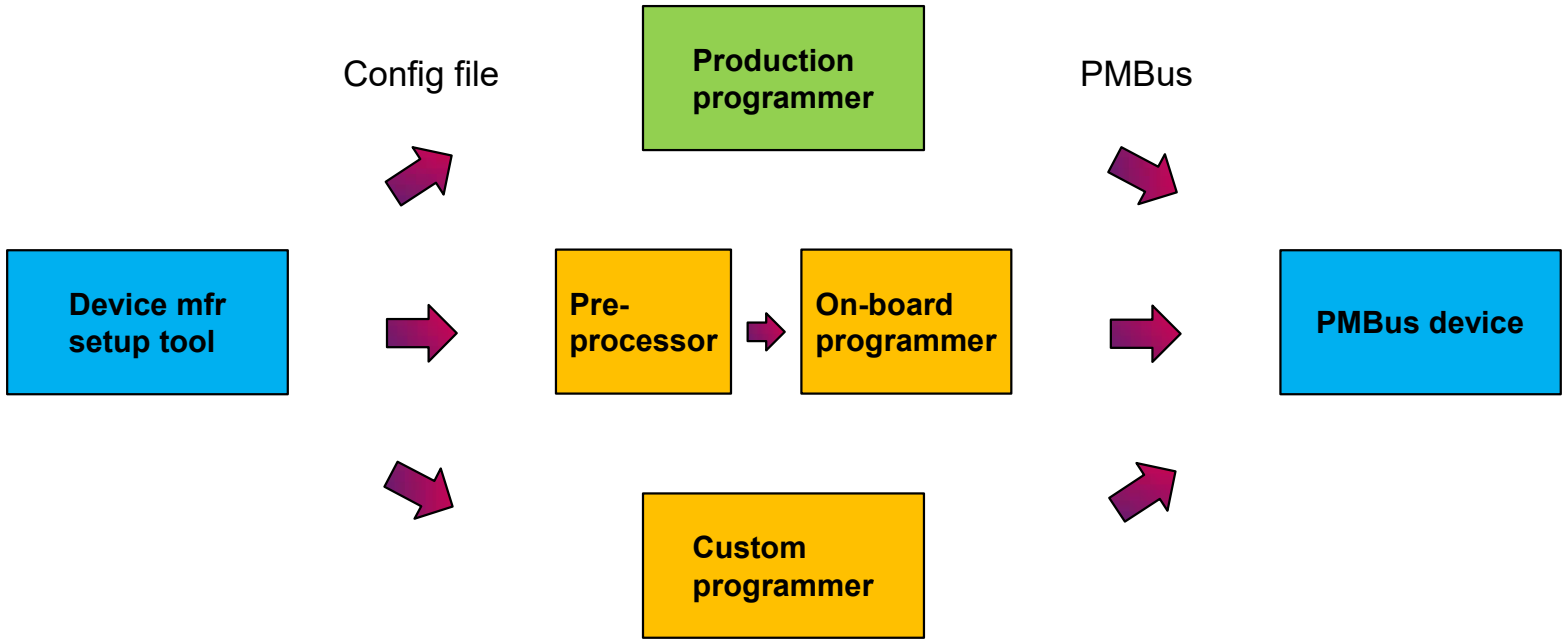
Ericsson

APEC 2017

PMBus device setup

- Device configuration.
  - Set of parameters that determine device behavior.
- Parameter setup by HW.
  - Default values.
  - Pin-programmed (e.g. by R or C).
- Parameter setup by bus programming
  - Stored in Non-volatile memory.
  - Configured directly by bus (volatile).
- Can be combined
  - Precedence order as per PMBus standard.

Bus programming use cases



- Device manufacturer
- Device user
- Standard tool (3rd party)

Problem

System Management Interface Forum

- Bus programming is desired for flexibility, BUT…
  - Requires extra step in production.
- And…
  - Programming procedure is not standardized.
  - Programming data format is not standardized.

Result is a new production procedure for every different device! (more or less)

PMBus® Power Management. Defined. 1.3

System Management SMBus

SBS Smart Battery System Implementers Forum

Complications

System Management Interface Forum

- Non-volatile store using manufacturer specific commands.
  - Instead of using the standard STORE-commands.
- Extra procedural steps, e.g.:
  - Unlocking areas.
  - Switching between write areas.
  - Switching on the non-volatile memory HW.
  - Reading data, modifying it and writing it back.

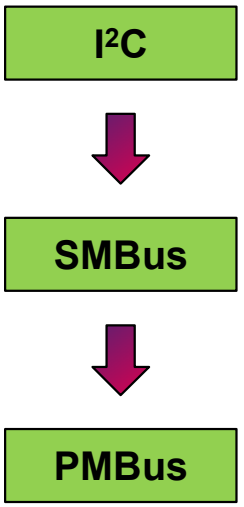
PMBus® Power Management. Defined. 1.3 System Management SMBus Smart Battery System Implementers Forum

Other challenges

- Busy indication
  - Time for storing to Non-volatile memory can exceed bus timeout.
  - “Busy”-bit only states that device was too busy to respond properly.
  - No standard way of indicating STORE-operation in progress.
- Fault indication
  - No specific indicator that a STORE-operation failed.
  - No specific indicator that the configuration is corrupt.
- Verification
  - Read back is complicated if the device truncates the data.

Bus evolution

- I^2C bus transfer is just a sequence of bytes, starting with an Address byte.
- SMBus builds on I^2C, but specifies "SMBus transactions".
  - One byte designated as "Command".
  - Specified allowed byte lengths
  - Read transactions are composite.
  - Adds Packet Error Check capability (CRC-8)
- PMBus uses SMBus transactions, but specifies content.
  - Application profiles adds interoperability.



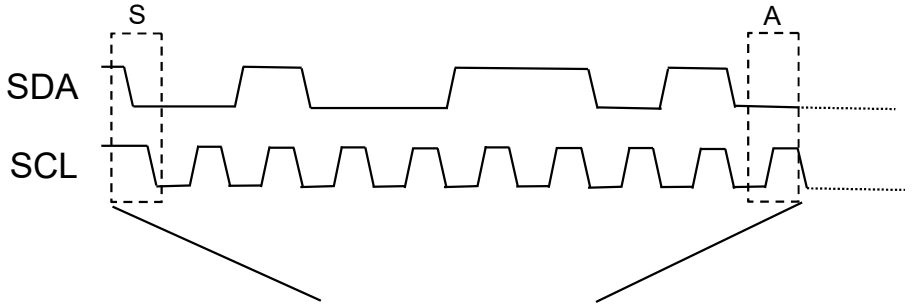
System Management Interface Forum

PMBus Power Management. Defined. 1.3

System Management SMBus

SBS Smart Battery System Implementers Forum

I²C bus transfer

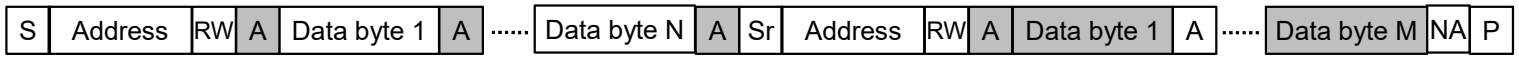


Write (to slave):

Payload (not specified)

Read (from slave):

Combined format:

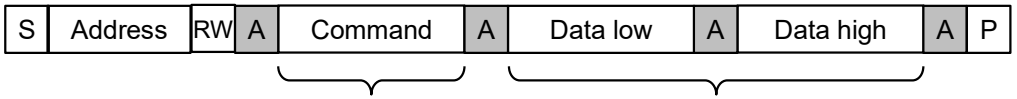


Note: Shaded indicates slave transmitting, S=Start, P=Stop, A=Acknowledge, NA=Not acknowledge

SMBus transactions

Example, word transactions:

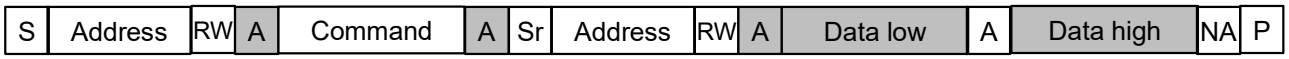
SMBus *Write word*:



Command (not specified)

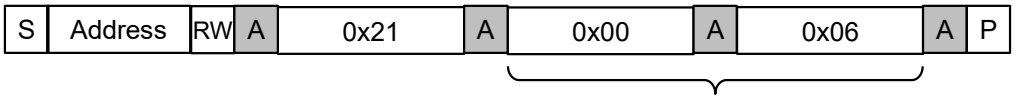
Command argument (word size)

SMBus *Read word*:



PMBus uses SMBus-transactions, but define Command and data format.

Example, PMBus output voltage adjustment (VOUT_COMMAND):



0x0600 = **1.5V** (LIN16-format, exp = -10)

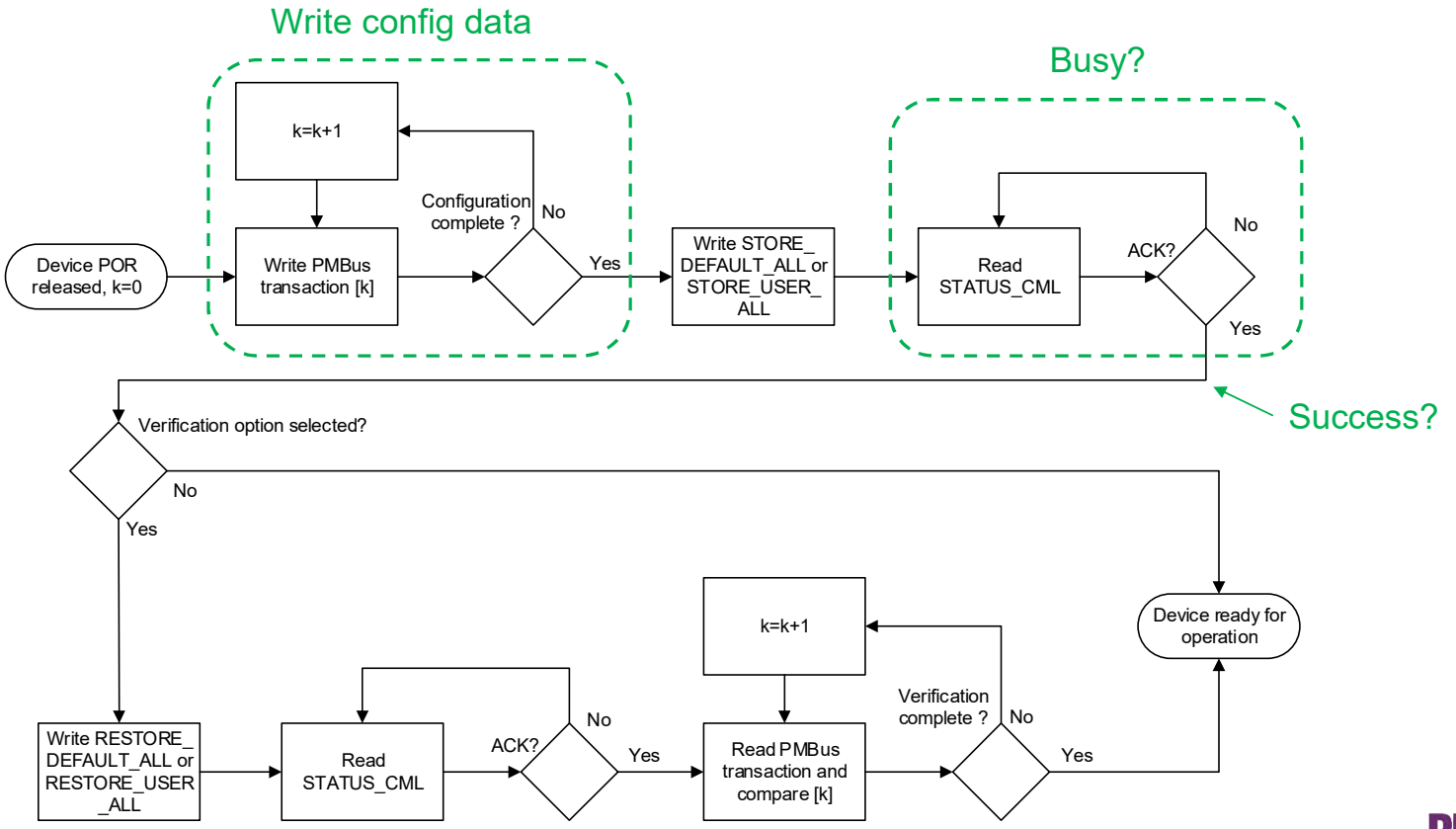
Inferring a procedure

- Lowest common denominator:
  ➡ PMBus uses SMBus transactions on the bus.

- Standard commands: STORE_USER_ALL
  STORE_DEFAULT_ALL

- Status command STATUS_CML has memory fault bit.
  - STATUS_CML NACKs if STORE-operation is busy (not supported in all devices).
  - Will then reflect if STORE-operation succeeded or failed.

Example procedure flow



Example file format

System Management Interface Forum

- Is a list of SMBus transactions in the order they are to be transferred plus control commands
  - ”Command”-byte + transaction type + data bytes
- Control commands:
  - ”SET” : sets address for subsequent commands
  - ”DELAY” : delays execution by specified number of ms
- Command options:
  - ”expect” : expected data when reading
  - ”expect_mask” : binary mask for the expected data.

PMBus® Power Management. Defined. 1.3

System Management SMBus

SBS Smart Battery System Implementers Forum

File format example

```
SET address=24
02 WRITE_BYTE   17              # ON_OFF_CONFIG
21 WRITE_WORD   CD10            # VOUT_COMMAND  1.05 V
33 WRITE_WORD   5802            # FREQUENCY_SWITCH  600 kHz
DELAY 5
40 WRITE_WORD   5213            # VOUT_OV_FAULT_LIMIT  1.2075 V
44 WRITE_WORD   480E            # VOUT_UV_FAULT_LIMIT   0.89258 V
46 WRITE_WORD   00DA            # IOUT_OC_FAULT_LIMIT  16 A
60 WRITE_WORD   C0D3            # TON_DELAY  15 ms
61 WRITE_WORD   00C3            # TON_RISE  3 ms
15 SEND_BYTE                    # STORE_USER_ALL
```

Configuration

```
SET address=24
02 READ_BYTE expect=17 expect_mask=1F    # ON_OFF_CONFIG
20 READ_BYTE expect=14                   # VOUT_MODE
21 READ_WORD expect=CD10                 # VOUT_COMMAND   1.05 V
33 READ_WORD expect=5802                 # FREQUENCY_SWITCH   600 kHz
40 READ_WORD expect=5213                 # VOUT_OV_FAULT_LIMIT  1.2075 V
44 READ_WORD expect=480E                 # VOUT_UV_FAULT_LIMIT  0.89258 V
46 READ_WORD expect=00DA                 # IOUT_OC_FAULT_LIMIT  16 A
60 READ_WORD expect=C0D3                 # TON_DELAY   15 ms
61 READ_WORD expect=00C3                 # TON_RISE   3 ms
```

Verification

Advantages

- Flexibility
  - Based on transactions eliminates need ”know” what data does.
  - Not limited to PMBus, but also SMBus devices.
- Simplicity
  - Allows programming to be executed on resource limited HW.
  - Limits risk in the transfer/programming stage.
  - Complexity is handled by the generating tool (e.g. setup GUI)
- Extensible
  - Subset of larger set functions.
  - Extension will increase complexity. What is really necessary?

Limitations

- Requires a separate file for verification
  - Currently no defined delimiter between configuration and verification parts.
- Only a single data type
  - Verbosity by comments only.
  - Comments only help interpretation, not modification.
- No support for modifying existing data, e.g.
  - Adding to existing value (or any other arithmetic operation).
  - Setting/resetting a single bit (or few bits).
- No support for configuration flow control e.g.
  - *"if – then - else"*

PMBus configuration file format standardization

System Management Interface Forum

- Work is ongoing in the PMBus Specification Workgroup Committee.
- Focus is on the configuration file format.
  - Based on bus transactions.
  - XML-based.
  - Procedure is not currently in the scope.
  - Will support all existing devices.

PMBus® Power Management. Defined. 1.3

System Management SMBus

SBS Smart Battery System Implementers Forum

System Management Interface Forum

Thank you!

Questions?

Smart Battery System Implementers Forum