Report on changes in NVMEM use in PTB between SVI FF R2 and R3

# Purpose of redesign

1. Reduce the sizes of NVMEM objects. That would
   1. Allow inline (i.e. not deferred) writes of smaller objects (currently, up to 200 bytes)
   2. Reduce the average amount of NVMEM accesses and thus
      1. CPU load
      2. Power consumption (though probably negligible)
2. Avoid pointless duplication of items owned by and stored in APP. That would
   1. Allow inline (i.e. not deferred) writes of smaller objects (currently, up to 200 bytes)
   2. Reduce the average amount of NVMEM accesses and thus
      1. CPU load
      2. Power consumption (though probably negligible)
   3. Remove the risk of divergence of FFP and APP data
3. Pave the way to translating NVMEM image on activation of updated firmware, just as R2 does in APP
4. Guarantee non-destruction in any pattern of resets/power cycles

# Implementation issues

Below are annotated parameters of Static and NV attributes which are gone from T\_FBIF\_PTB\_STATIC and T\_FBIF\_PTB\_NV types, respectively.

The parameters that remain there are stored, like parameters of other blocks, using the original Softing eep API. However, eep implementation is now a simple shim layer over the standard MN NVRAM. This minimized the amount of work since the remaining objects are relatively small.

The parameters annotated with “Now implemented directly in MN-style NVRAM via eep shim” still use the original eep API. Their shims are created automatically. The purpose of this is

1. To minimize code tear-up where no application-layer changes are needed, and
2. Keep the automation of ST\_REV

## Special Cases

The following data are stored in log files:

1. TB Static and NV data to save/restore “factory configuration”, just as in APP
2. Alert log

NOTE: “factory defaults” currently have a **bug**: only parameters kept in TB static or NV types are saved/restored. To fix this, need to port APP “factory defaults”, but the question is, should we limit ourselves to TB or apply to all blocks

# Static

## SCALE final\_value\_range;

Constant; doesn’t need storage

## \_POSITION\_LIMITS position\_limits;

Stored in APP; no need to cache (not used in FFP other than to read/write)

## \_FINAL\_VALUE\_CUTOFF\_HI final\_value\_cutoff\_hi;

Stored in APP; no need to cache (not used in FFP other than to read/write)

## \_FINAL\_VALUE\_CUTOFF\_LO final\_value\_cutoff\_lo;

Stored in APP; no need to cache (not used in FFP other than to read/write)

## \_CONTROL\_SET custom\_control\_set;

No need to store scratchpad parameter

## --

## \_DEVIATION\_ALERT deviation\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_POSITION\_ALERT position\_hihi\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_POSITION\_ALERT position\_hi\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_POSITION\_ALERT position\_lo\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_POSITION\_ALERT position\_lolo\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_NEAR\_CLOSED\_ALERT near\_closed\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_SYSTEM\_ALERT setpoint\_timeout\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## --

## \_CHAR\_TYPE char\_selection;

Stored in APP; no need to cache (not used in FFP other than to read/write)

For speed, however, cached in FFP RAM. Only if magic number doesn’t match, fetch from APP. This has been so since R1 and there is no need to store in FFP

## --

## INT16 custom\_char\_points[38];

Now implemented directly in MN-style NVRAM

However, it is a **bug**: no need to save a scratchpad. Cf. [custom\_control\_set](#__CONTROL_SET_custom_control_set;).

Another **bug** is writing the scratchpad succeeds but *something* is rejected for incorrect mode for request (TB in AUTO). Scratchpad shouldn’t care.

## --

## \_SYSTEM\_ALERT supply\_pressure\_hi\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_SYSTEM\_ALERT supply\_pressure\_lo\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_SYSTEM\_ALERT supply\_pressure\_lolo\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_SYSTEM\_ALERT temperature\_hi\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_SYSTEM\_ALERT temperature\_lo\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_IP\_DRIVE\_CURRENT\_ALERT ip\_drive\_current\_hi\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_IP\_DRIVE\_CURRENT\_ALERT ip\_drive\_current\_lo\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## --

## USIGN8 ui\_language;

Stored in APP; no need to cache (not used in FFP other than to read/write)

## FLOAT open\_stop\_adjustment;

Stored in APP; no need to cache (not used in FFP other than to read/write)

## --

## \_ALERT\_ACTION alert\_action;

Now implemented directly in MN-style NVRAM via eep shim

## --

## \_WORKING\_TIME\_ALERT working\_time\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_ACTUATOR\_3 actuator\_3;

Now implemented directly in MN-style NVRAM via eep shim

## \_ACCESSORY accessory;

Now implemented directly in MN-style NVRAM via eep shim

--

# NV

## \_TRAVEL\_CALIBRATION travel\_calibration;

Now implemented directly in MN-style NVRAM via eep shim

## \_TRAVEL\_ALERT travel\_accumulation\_a\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_TRAVEL\_ALERT travel\_accumulation\_b\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_TREND travel\_accumulation\_trend;

Now implemented directly in MN-style NVRAM outside of eep model

## \_CYCLE\_COUNTER\_ALERT cycle\_counter\_a\_alert;

Now implemented directly in MN-style NVRAM via eep shim

## \_CYCLE\_COUNTER\_ALERT cycle\_counter\_b\_alert;

## \_TREND cycle\_counter\_trend;

Now implemented directly in MN-style NVRAM outside of eep model

## \_ERR\_TREND position\_error\_trend;

Now implemented directly in MN-style NVRAM outside of eep model

## \_POS\_HISTOGRAM position\_histogram;

Now implemented directly in MN-style NVRAM outside of eep model

## \_POS\_ERR\_HISTOGRAM position\_error\_histogram;

Now implemented directly in MN-style NVRAM outside of eep model

## DISCRETE\_S final\_value\_d;

No need to store this recomputed parameter (similar to final\_value)

## DISCRETE\_S final\_value\_dint;

No need to store this recomputed parameter (similar to final\_value)

## \_WORKING\_TIMES working\_times;

Now implemented directly in MN-style NVRAM via eep shim

## \_ACTUATOR\_1 actuator\_1;

Now implemented directly in MN-style NVRAM via eep shim

## \_ACTUATOR\_2 actuator\_2;

Now implemented directly in MN-style NVRAM via eep shim

## \_ACTUATOR\_INFO actuator\_info;

Now implemented directly in MN-style NVRAM via eep shim

## \_VALVE\_IDENTIFICATION valve\_identification;

Now implemented directly in MN-style NVRAM via eep shim

## \_VALVE\_SERVICE valve\_service;

Now implemented directly in MN-style NVRAM via eep shim

## \_VALVE\_BODY\_1 valve\_body\_1;

Now implemented directly in MN-style NVRAM via eep shim

## \_VALVE\_BODY\_2 valve\_body\_2;

Now implemented directly in MN-style NVRAM via eep shim

## \_VALVE\_BODY\_3 valve\_body\_3;

Now implemented directly in MN-style NVRAM via eep shim

## \_VALVE\_INFO valve\_info;

Now implemented directly in MN-style NVRAM via eep shim

## \_BOOSTER booster;

Now implemented directly in MN-style NVRAM via eep shim

## \_POSITION\_EXTREMES position\_extremes;

Now implemented directly in MN-style NVRAM via eep shim

## \_PRESSURE\_EXTREMES pressure\_extremes;

Now implemented directly in MN-style NVRAM via eep shim

## \_TEMPERATURE\_EXTREMES temperature\_extremes;

Now implemented directly in MN-style NVRAM via eep shim

## \_IP\_CURRENT\_EXTREMES ip\_current\_extremes;

Now implemented directly in MN-style NVRAM via eep shim

## USIGN32 advanced[14];

Now implemented directly in MN-style NVRAM outside of eep model