```
/**
 *
 * @copyright &copy; 2010 - 2020, Fraunhofer-Gesellschaft zur Foerderung der
 *  angewandten Forschung e.V. All rights reserved.
 *
 * BSD 3-Clause License
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 * 1.  Redistributions of source code must retain the above copyright notice,
 *     this list of conditions and the following disclaimer.
 * 2.  Redistributions in binary form must reproduce the above copyright
 *     notice, this list of conditions and the following disclaimer in the
 *     documentation and/or other materials provided with the distribution.
 * 3.  Neither the name of the copyright holder nor the names of its
 *     contributors may be used to endorse or promote products derived from
 *     this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE
 * LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
 * CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
 * SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
 * INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
 * CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
 * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
 * POSSIBILITY OF SUCH DAMAGE.
 *
 * We kindly request you to use one or more of the following phrases to refer
 * to foxBMS in your hardware, software, documentation or advertising
 * materials:
 *
 * &Prime;This product uses parts of foxBMS&reg;&Prime;
 *
 * &Prime;This product includes parts of foxBMS&reg;&Prime;
 *
 * &Prime;This product is derived from foxBMS&reg;&Prime;
 *
 */

/**
 * @file    database_cfg.c
 * @author  foxBMS Team
 * @date    18.08.2015 (date of creation)
 * @ingroup ENGINE_CONF
 * @prefix  DATA
 *
 * @brief   Database configuration
 *
 * Configuration of database module
 *
```

```
 */

/*================== Includes ============================================*/
#include "database_cfg.h"

/*================== Macros and Definitions ==============================*/

/*================== Static Constant and Variable Definitions ============*/
/**
 * data block: cell voltage
 */
static DATA_BLOCK_CELLVOLTAGE_s data_block_cellvoltage;

/**
 * data block: cell temperature
 */
static DATA_BLOCK_CELLTEMPERATURE_s data_block_celltemperature;

/**
 * data block: sox
 */
static DATA_BLOCK_SOX_s data_block_sox;

/**
 * data block: sof
 */
static DATA_BLOCK_SOF_s data_block_sof;

/**
 * data block: balancing control
 */
static DATA_BLOCK_BALANCING_CONTROL_s data_block_control_balancing;

/**
 * data block: balancing feedback
 */
static DATA_BLOCK_BALANCING_FEEDBACK_s data_block_feedback_balancing;

/**
 * data block: current measurement
 */
static DATA_BLOCK_CURRENT_SENSOR_s data_block_curr_sensor;

/**
 * data block: ADC
 */
static DATA_BLOCK_HW_INFO_s data_block_hwinfo;

/**
 * data block: can state request
 */
static DATA_BLOCK_STATEREQUEST_s data_block_staterequest;
```

```
105
106 /**
107  * data block: LTC minimum and maximum values
108  */
109 static DATA_BLOCK_MINMAX_s data_block_minmax;
110
111 /**
112  * data block: isometer measurement
113  */
114 static DATA_BLOCK_ISOMETER_s data_block_isometer;
115
116 /**
117  * data block: error flags
118  */
119 static DATA_BLOCK_ERRORSTATE_s data_block_errors;
120
121 /**
122  * data block: maximum safety limit violations
123  */
124 static DATA_BLOCK_MSL_FLAG_s data_block_MSL;
125
126 /**
127  * data block: recommended safety limit violations
128  */
129 static DATA_BLOCK_RSL_FLAG_s data_block_RSL;
130
131 /**
132  * data block: maximum operating limit violations
133  */
134 static DATA_BLOCK_MOL_FLAG_s data_block_MOL;
135
136 /**
137  * data block: moving mean current and power
138  */
139 static DATA_BLOCK_MOVING_AVERAGE_s data_block_mov_average;
140
141 /**
142  * data block: contactor feedback
143  */
144 static DATA_BLOCK_CONTFEEDBACK_s data_block_contfeedback;
145
146 /**
147  * data block: interlock feedback
148  */
149 static DATA_BLOCK_ILCKFEEDBACK_s data_block_ilckfeedback;
150
151 /**
152  * data block: slave control
153  */
154 static DATA_BLOCK_SLAVE_CONTROL_s data_block_slave_control;
155
156 /**
```

```
157   * data block: system state
158   */
159  static DATA_BLOCK_SYSTEMSTATE_s data_block_systemstate;
160
161  /**
162   * data block: open wire check
163   */
164  static DATA_BLOCK_OPENWIRE_s data_block_open_wire;
165
166  /**
167   * data block: LTC diagnosis values
168   */
169  static DATA_BLOCK_LTC_DEVICE_PARAMETER_s data_block_ltc_diagnosis;
170
171  /**
172   * data block: LTC ADC accuracy verification
173   */
174  static DATA_BLOCK_LTC_ADC_ACCURACY_s data_block_ltc_adc_accuracy;
175
176  /**
177   * data block: LTC ADC accuracy verification
178   */
179  static DATA_BLOCK_ALLGPIOVOLTAGE_s data_block_ltc_allgpiovoltages;
180
181  /**
182   * data block: SOH of contactors
183   */
184  static DATA_BLOCK_CONT_SOH_s data_block_contactor_soh;
185
186  /**
187   * @brief channel configuration of database (data blocks)
188   *
189   * all data block managed by database are listed here (address,size,consistency type)
190   *
191   */
192  static DATA_BASE_HEADER_s  data_base_header[] = {
193      {
194          (void*)(&data_block_cellvoltage),
195          sizeof(DATA_BLOCK_CELLVOLTAGE_s)
196      },
197      {
198          (void*)(&data_block_celltemperature),
199          sizeof(DATA_BLOCK_CELLTEMPERATURE_s)
200      },
201      {
202          (void*)(&data_block_sox),
203          sizeof(DATA_BLOCK_SOX_s)
204      },
205      {
206          (void*)(&data_block_control_balancing),
207          sizeof(DATA_BLOCK_BALANCING_CONTROL_s)
208      },
```

data_base_header[] defines the pair of (dataPointer, size). With this info, we can access all the contents in the database.

```
209         {
210             (void*)(&data_block_feedback_balancing),
211             sizeof(DATA_BLOCK_BALANCING_FEEDBACK_s)
212         },
213         {
214             (void*)(&data_block_curr_sensor),
215             sizeof(DATA_BLOCK_CURRENT_SENSOR_s)
216         },
217         {
218             (void*)(&data_block_hwinfo),
219             sizeof(DATA_BLOCK_HW_INFO_s)
220         },
221         {
222             (void*)(&data_block_staterequest),
223             sizeof(DATA_BLOCK_STATEREQUEST_s)
224         },
225         {
226             (void*)(&data_block_minmax),
227             sizeof(DATA_BLOCK_MINMAX_s)
228         },
229         {
230             (void*)(&data_block_isometer),
231             sizeof(DATA_BLOCK_ISOMETER_s)
232         },
233         {
234             (void*)(&data_block_slave_control),
235             sizeof(DATA_BLOCK_SLAVE_CONTROL_s)
236         },
237         {
238             (void*)(&data_block_open_wire),
239             sizeof(DATA_BLOCK_OPENWIRE_s)
240         },
241         {
242             (void*)(&data_block_ltc_diagnosis),
243             sizeof(DATA_BLOCK_LTC_DEVICE_PARAMETER_s)
244         },
245         {
246             (void*)(&data_block_ltc_adc_accuracy),
247             sizeof(DATA_BLOCK_LTC_ADC_ACCURACY_s)
248         },
249         {
250             (void*)(&data_block_errors),
251             sizeof(DATA_BLOCK_ERRORSTATE_s)
252         },
253         {
254             (void*)(&data_block_MSL),
255             sizeof(DATA_BLOCK_MSL_FLAG_s)
256         },
257         {
258             (void*)(&data_block_RSL),
259             sizeof(DATA_BLOCK_RSL_FLAG_s)
260         },
```

```
261         {
262             (void*)(&data_block_MOL),
263             sizeof(DATA_BLOCK_MOL_FLAG_s)
264         },
265         {
266             (void*)(&data_block_mov_average),
267             sizeof(DATA_BLOCK_MOVING_AVERAGE_s)
268         },
269         {
270             (void*)(&data_block_contfeedback),
271             sizeof(DATA_BLOCK_CONTFEEDBACK_s)
272         },
273         {
274             (void*)(&data_block_ilckfeedback),
275             sizeof(DATA_BLOCK_ILCKFEEDBACK_s)
276         },
277         {
278             (void*)(&data_block_systemstate),
279             sizeof(DATA_BLOCK_SYSTEMSTATE_s)
280         },
281         {
282             (void*)(&data_block_sof),
283             sizeof(DATA_BLOCK_SOF_s)
284         },
285         {
286             (void*)(&data_block_ltc_allgpiovoltages),
287             sizeof(DATA_BLOCK_ALLGPIOVOLTAGE_s)
288         },
289         {
290             (void*)(&data_block_contactor_soh),
291             sizeof(DATA_BLOCK_CONT_SOH_s)
292         },
293     };
294
295
296     /*================== Extern Constant and Variable Definitions ===============*/
297
298     /**
299      * @brief device configuration of database
300      *
301      * all attributes of device configuration are listed here (pointer to channel list, number of channels)
302      */
303     const DATA_BASE_HEADER_DEV_s data_base_dev = {
304         .nr_of_blockheader  = sizeof(data_base_header)/sizeof(DATA_BASE_HEADER_s),    /* number of blocks (and block
        headers) */
305         .blockheaderptr     = &data_base_header[0],
306     };
307
308     /*================== Static Function Prototypes ============================*/
309
310     /*================== Static Function Implementations =======================*/
311
```

data_base_dev contains two values:
* Total number of data pairs in data_base_header
* The pointer to data_base_header.

With data_base_dev, we can access the database anywhere. Hence, this variable exposed in the header file.

```
312  /*================== Extern Function Implementations ========================*/
313
```