Coding convention

1. Common Rules

1.1 General

- No tab character, 1 tab = 4 spaces
- No trailing space
- The maximum width of a code line should not exceed 250 characters
- Avoid global variables or else justify their usage
- Avoid extern variables or else justify their usage
- No unconditional jumps:
    - break is used only in switch statement
    - continue is avoided
    - goto is avoided
    - return is used only one time at end of function
- No recursions
- Restricted size and complexity of function
- Restricted size number of function parameters
- Local (file–scope) objects (variables or functions) that do not need external linkage shall be declared as static
- Function parameter should be checked (apply MOCK 1 and MCP)

1.2 Variables Naming Convention and rules

1.2.1 Global Variable:

g_[VariableName]: camel case for VariableName.

1.2.2 Static Variable:

s_[VariableName]: camel case for variableName.

1.2.3 Local Variable:

Local variables used inside function should be defined at the beginning of function (after assert() block), unless variable is only used inside #wrapper.

Exception: By using local variables declared at the C code blocks the compiler can optimize the stack usage and/or register allocation. To be analyzed case by case. If special optimizations are requested, this rule can be ignored with remark in comment.

One local variable get its own line but not add several local variables into one line like

```
uint32_t a, b, c;  /* Wrong format. */
```

Local variable must be initialized.

1.3 Comments

Always use "/**/" for code comments.

Examples:

```
int i;

for (;;)

{

    i++; /* Increase i. */

}
```

1.4 Hard Coding Numbers

All hard coding unsigned numbers should be postfixed with 'U'. MISRA 2012 Rule 7.2: A 'u' or 'U' suffix shall be applied to all integer constants that are represented in an unsigned type

Examples:

```
Hex: 0x1234ABCDU

Dec: 1234U
```

1.5 Preprocessor

- Comments should be appended after the #endif:

```
    #if I_AM_A_MACRO


    #endif /* I_AM_A_MACRO */
```

- Add a blank line at the end of file.
- Header files guard macro

```
#ifndef _HEADER_FILENAME_

#define _HEADER_FILENAME_


#endif /* _HEADER_FILENAME_ */
```

2 General comment style

Add macros, enumeration types, structure types, inside:

```
/*******************************************************************************
 * Definitions
 ******************************************************************************/
```

Add global variables definition inside:

```
/*******************************************************************************
 * Variables
 ******************************************************************************/
```

Add static API prototype in C file inside:

```
/*******************************************************************************
 * Prototypes
 ******************************************************************************/
```

Add function implementation inside:

```
/*******************************************************************************
 * Code
 ******************************************************************************/
```

Add public API in header files inside:

```
/*******************************************************************************
 * API
 ******************************************************************************/
```

- For header file, the sequence is:
  - Definition
  - API
- For C file, the sequence is:

- Definitions
- Prototypes
- Variables
- Codes