

**Coding Standards**

**EVSE Automation**

|  |  |
| --- | --- |
|  |  |

**Confidentiality Notice**

Copyright 2017 eInfochips. - All rights reserved

This document is authored by eInfochips and is eInfochips intellectual property, including the copyrights in all countries in the world. This document is provided under a license to use only with all other rights, including ownership rights, being retained by eInfochips. This file may not be distributed, copied, or reproduced in any manner, electronic or otherwise, without the express written consent of eInfochips.

Contents

[1 Document Details 4](#_Toc484520521)

[1.1 Document History 4](#_Toc484520522)

[1.2 Definition, Acronyms and Abbreviations 4](#_Toc484520523)

[1.3 References 4](#_Toc484520524)

[2 Introduction 5](#_Toc484520525)

[2.1 Purpose of the Document 5](#_Toc484520526)

[2.2 Scope of the Document 5](#_Toc484520527)

[3 Guidance 6](#_Toc484520528)

[3.1 Naming Convention 6](#_Toc484520529)

[3.1.1 Variables 6](#_Toc484520530)

[3.1.2 Functions 6](#_Toc484520531)

[3.1.3 Data Types 7](#_Toc484520532)

[3.1.4 Macros 7](#_Toc484520533)

[3.1.5 Structure 7](#_Toc484520534)

[3.2 File Naming and Organization 7](#_Toc484520535)

[3.2.1 Header Files 7](#_Toc484520536)

[3.3 Formatting and Indentation 8](#_Toc484520537)

[3.4 Comments and Documentation 8](#_Toc484520538)

[3.5 Example source file and header file 8](#_Toc484520539)

**TABLES**

[Table 1: Document History 4](#_Toc484520540)

[Table 2: Definition, Acronyms and Abbreviations 4](#_Toc484520541)

[Table 3: References 4](#_Toc484520542)

[Table 4: Variable Prefix 6](#_Toc484520543)

[Table 5: Data Types 7](#_Toc484520544)

# Document Details

## Document History

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Version** | **Author** | | **Reviewer** | | **Approver** | | **Description Of Changes** |
| **Name** | **Date** | **Name** | **Review Comment ID** | **Name** | **Date** |
| 0.1 | eInfochips | 06 June 2017 | eInfochips |  |  |  | Created initial draft version |
|  |  |  |  |  |  |  |  |

Table 1: Document History

## Definition, Acronyms and Abbreviations

|  |  |
| --- | --- |
| Definition/Acronym/Abbreviation | Description |
| EVSE | Electric Vehicle Supply Equipment |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Table 2: Definition, Acronyms and Abbreviations

## References

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Document | Version | Remarks |
| 1. |  |  | - |

Table 3: References

# Introduction

## Purpose of the Document

The purpose of this document is to define the coding guidelines for various programming components so that standard and uniform way of coding is followed by all the project team members in the project. This document will serve as a single reference document for all coding done.

## Scope of the Document

The scope of this guideline is all software code written in C or C++ language for this project.

# Guidance

## Naming Convention

### Variables

Variable name will be as per below guidelines

1. First lower case characters (Maximum 3 character) will display data type of variable.
2. Next characters will display functionality of variable. Its first character will also be capital to distinguish it from module name. There is no limit of functionality name.

|  |  |
| --- | --- |
| Data Type | Functionality |

*Data Type* will be as per below table

|  |  |
| --- | --- |
| **Data types** | **Short form** |
| char | CPU\_CHAR |
| unsigned char | CPU\_BOOLEAN or boolean |
| unsigned char | CPU\_INT08U or ubyte |
| unsigned int | CPU\_INT32U |
| signed int | CPU\_INT32S |
| unsigned short | CPU\_INT16U or uword |
| signed short | CPU\_INT16S |
| signed long long | CPU\_INT64S |
| unsigned long long | CPU\_INT64U |

Table 4: Variable Prefix

Variable name example: CPU\_BOOLEAN result

Data type: CPU\_BOOLEAN

Functionality: result

### Functions

**Function name:**

Function name will be as per below guidelines

1. First two characters will display module name, for which it will be used.
2. Remaining characters will display functionality of function. It has no length limit and first character will be of capital letter.

|  |  |
| --- | --- |
| Module Name | Functionality |

**Function declaration:**

Function declaration will include arguments with data type and name.

Example: Boolean MH\_Init\_Modbus\_Access\_Task ( void );

**Function Header:**

Header of function is as per given in section “[Example source file and header file](#_Example_source_file)”

### Data Types

We will use in our developed code, below mentioned type defined data types. This will be declared in separate type definition header file.

|  |  |
| --- | --- |
| **Data types** | **Short form** |
| char | CPU\_CHAR |
| unsigned char | CPU\_BOOLEAN or Boolean |
| unsigned char | CPU\_INT08U |
| unsigned int | CPU\_INT32U or uint |
| signed int | CPU\_INT32S |
| unsigned short | CPU\_INT16U or ushort |
| signed short | CPU\_INT16S |
| signed long long | CPU\_INT64S |
| unsigned long long | CPU\_INT64U |
| unsigned long | ulong |

Table 5: Data Types

Example: ubyte load\_cnt

### Macros

Pre-processor macro names will be in capital letters.

Example:

#define MH\_ILLEGAL\_FUNCTION 1

### Structure

Name of structure will start with underscore and all the letters will be in lower case. Name of the structure variable will be in capital letters. Refer the below example.

Example:

**typedef** **struct** \_msg\_access\_info\_t

{

uword buf\_size; // size of message buffer (in bytes)

uword msg\_size; // size of message (in bytes)

ubyte \*msg\_pbt; // pointer to start of message buffer

ML\_ACCESS\_ERR\_T error; // returned error codes

} **MH\_MSG\_ACCESS\_INFO\_T**;

## File Naming and Organization

Each module has its own source file and its name will be based on that module.

For example

Source file :- MT\_NVM\_Task.c

Header File :- MT\_NVM\_Task.h

Header is be as per given in section “[Example source file and header file](#_Example_source_file)”

### Header Files

Each module has its own header file and its name will be same as its source file.

For example

Source file :- MT\_MBUShi.c

Header File :- MT\_MBUShi.h

Header is be as per given in section “[Example source file and header file](#_Example_source_file)”

## Formatting and Indentation

To make the reading of code easy (thus enhancing its maintainability), proper indentation will be used.

* All code statements which are linked to the next higher-level code statement will be right indented using one tab character with respect to the higher level statement.
* All code statements that are independent of each other in a code block, the number of indents used in all of them will be same.
* Tab will contain 4 characters.
* For easy readability each line will contain maximum 120 characters.
* Block of statements must be separated by blank lines to separate tasks
* Variables will be listed, one per line, with an inline comment describing its purpose

Example is given in section “[Example source file and header file](#_Example_source_file)”

## Comments and Documentation

Each function or procedure must have a description. The Description in the function header will be good enough to give an idea about the function, thus minimizing the commenting required in the body of the function. However, wherever a complex logic is coded in a code block inside a function, it will be commented.

Project will follow Doxygen documentation convention. Thus Doxygen tool can run on the code and HTML documentation can be generated.

## Example source file and header file



