**Buzz Software Design**

***SteerTurnIllum***



|  |  |  |
| --- | --- | --- |
| Author: |  | Nicolae-Bogdan Bacrău |
| Version: |  | 1.0 |

Index

1 Glossary 3

2 Introduction 4

2.1 Purpose of the Document 4

2.2 Overview 4

3 Design Requirements 5

4 Information about this Document 7

4.1 Copyright 7

4.2 Version Index 7

# Glossary

This section contains a glossary of all the important terms and acronyms used inside the document.

|  |  |
| --- | --- |
| **Term / Acronym** | **Description** |
| AUTOSAR | AUTomotive Open System ARchitecture |
| VFB | Virtual Functional Bus |
| SWC | Software Component |
| RTE | Runtime Environment |
| BSW | Basic Software |
| OS | Operating System |
| S/R | Sender / Receiver |
| C/S | Client / Server |
| ECU | Electronic Control Unit |
| uC | Microcontroller |
| ADC | Analog Digital Converter |
| DIO | Digital Input / Output |
| PWM | Pulse Width Modulation |

Table 1 - Glossary.

# Introduction

## Purpose of the Document

The purpose of the document is to define the software design of the ***Buzz*** SWC for the ***SteerTurnIllum*** embedded academy project.

## Overview

The ***Buzz*** SWC implements a simple mute or buzzing control of buzzers.

# Design Requirements

1. The Buzz SWC shall adhere to the structure illustrated in the composite structure diagram from **Figure 1**.

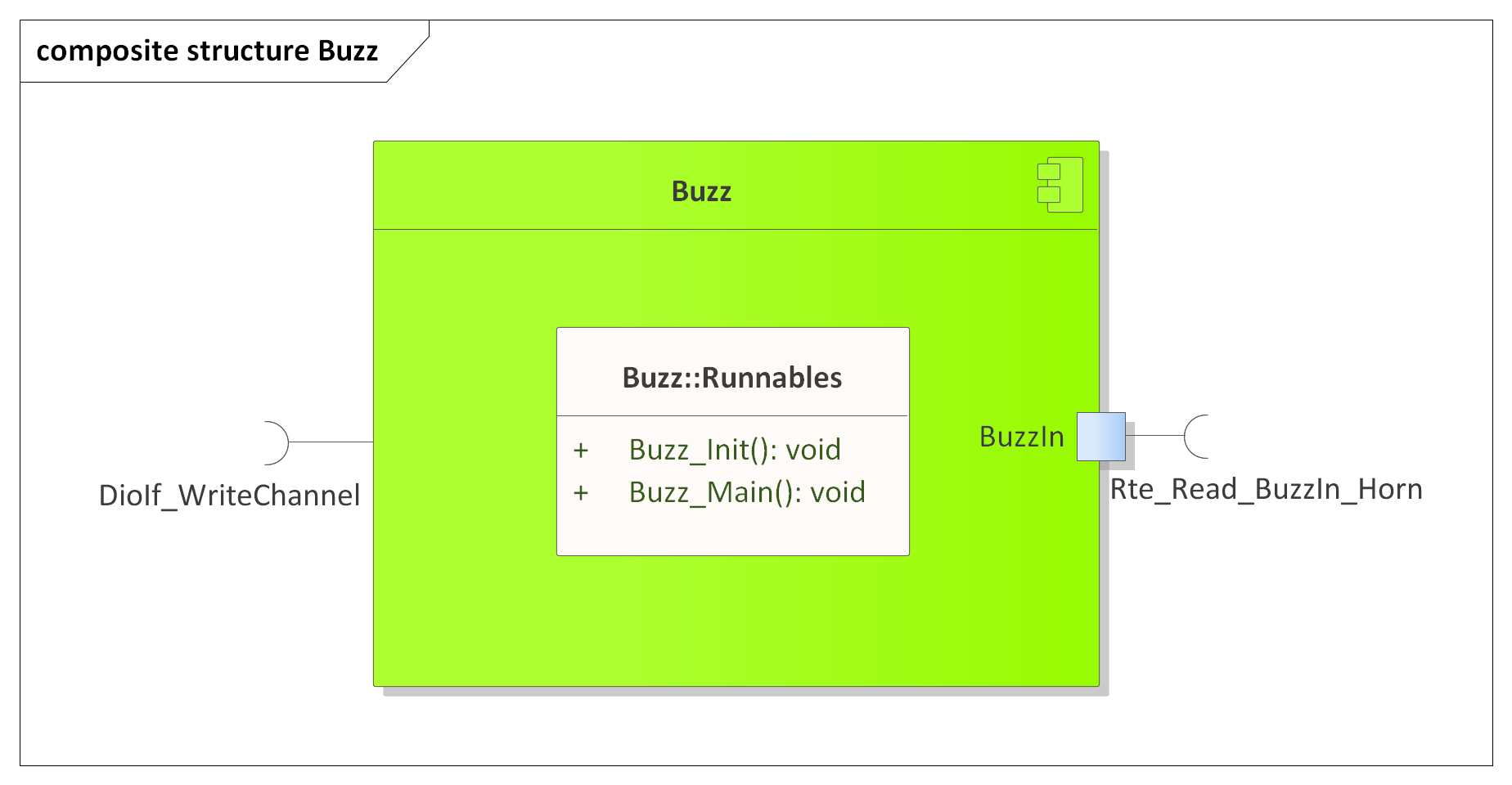


Figure 1 - Buzz composite structure diagram.

1. The Buzz SWC shall adhere to the SWC file structure template from 4\_Engineering\1\_Software\2\_Development\1\_Sources\8\_Templates\Swc\Code.
2. The Buzz SWC shall contain the following configuration parameters:

* In *Buzz\_Cfg.h*:
  + *BUZZ\_NUMBER\_OF\_INSTANCES*: macro defining the number of buzzer instances to be processed in the main function.
  + *BUZZ\_\*\_INSTANCE*: zero based macros defining unique IDs of all the buzzer instances, needed for the interaction between the core and RTE.
* In *Buzz\_Cfg.c*:
  + *Buzz\_gkat\_Config*[BUZZ\_NUMBER\_OF\_INSTANCES]: internal global constant array defining the VCC and IO DioIf channels for all buzzer instances.

1. The Buzz SWC shall implement the *void Buzz\_Init(void)* runnable for initializing all the internal static and global variables and setting the buzzer in the mute state through *DioIf\_WriteChannel()*.
2. The Buzz SWC shall implement the *void Buzz\_Main(void)* runnable for implementing the processing of all the buzzer instances as follows:
   * Reads the buzzer input control data through *Rte\_Read\_BuzzIn\_\*()*.
   * Sets the buzzer state according to the read input control data: mute if RTE\_BUZZ\_OFF, buzzing if RTE\_BUZZ\_ON, through *DioIf\_WriteChannel()*. Any other input control value shall be ignored.
3. The Buzz SWC shall include *DioIf.h* and directly use the *DioIf\_WriteChannel()* functionfor writing the buzzer DioIf channels.

# Information about this Document

## Copyright

All rights, including translation rights, are reserved. Under no circumstances shall any fragment of this document be reproduced without written authorization from NTT DATA Romania S.A, including copying, photographing or replicating through other methods.

Copyright (c) NTT DATA Romania S.A.

## Version Index

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Chapter** | **Modification description** |
| 1.0 | 10.07.2021 | Nicolae-Bogdan Bacrău | All | Created. |

Table 2 - Version Index.