

Bmbed *High-Level-Protocol*

Condition Document

Version 1.0



Contents

1	Introduction.....	2
1.1	Scope of this Document	2
1.2	Intended Audience	2
1.3	Terms and Abbreviations.....	2
2	Network and Communication	3
2.1	Overview.....	3
2.2	Protocol Architecture	4
3	Protocol Dictionary	5
3.1	Definition	5
3.2	Nodes	5
3.3	Commands.....	6
3.3.1	General.....	6
3.3.2	Bootloader	6
3.3.3	Application	6
4	Manufacturer Information	9
5	Version History	10



1 Introduction

1.1 Scope of this Document

This is the definition of the **Bmbed** High-Level-Protocol and is used as a reference for all development and test work within the **Bmbed** Framework.

1.2 Intended Audience

Customer technical product management, customer technical responsible, development engineers.

1.3 Terms and Abbreviations



2 Network and Communication

2.1 Overview

The **Bmbed** network connects different embedded boards and/or different Windows® personal computer as nodes together. It allows to communicate among each other through the **Bmbed High-Level-Protocol**.

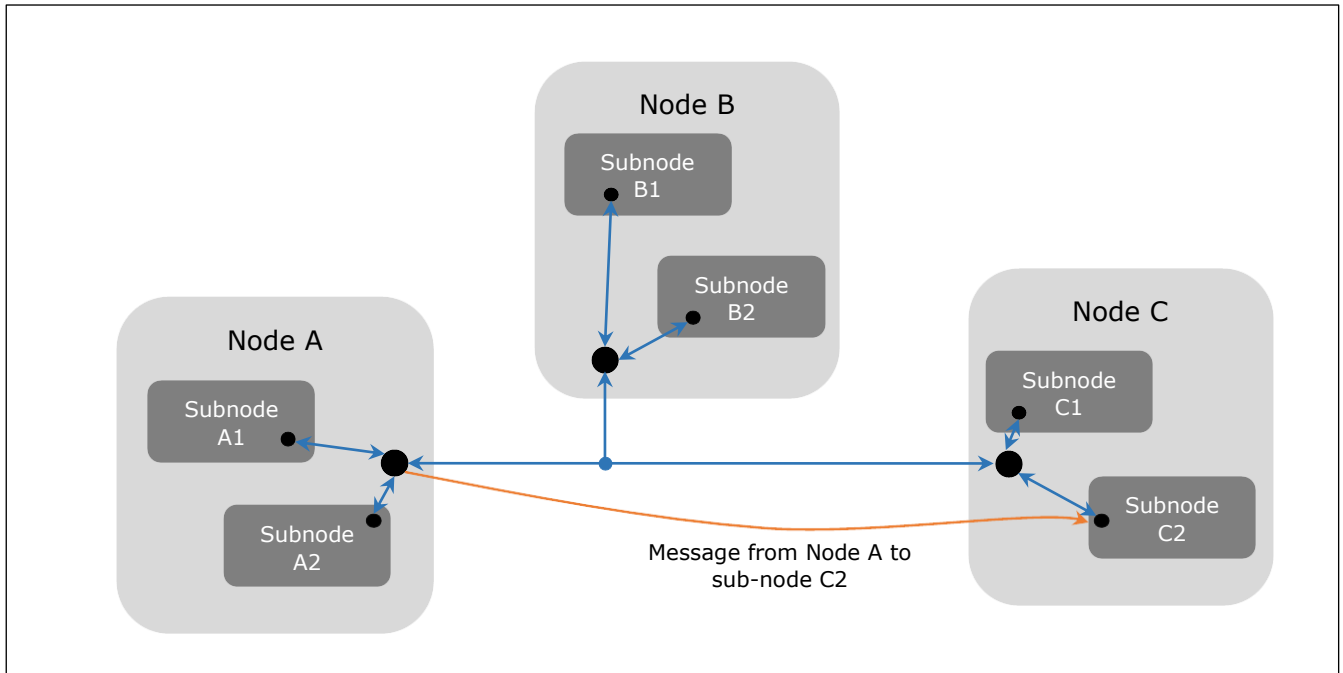


Figure 1 **Bmbed** Network Principle

Every network node represents a physical computer unit e.g. PC, Microcontroller, Embedded Board. Within a node there can be any sub-nodes.

Every node and sub-node contains a visible unique 16-bit address for the entire network.

Every communication occurs as peer-to-peer connection. This means that every node and sub-node sends and receives messages to/from any other node or sub-node.

Example:

Node A sends a message to sub-node C2 for switch on the Node C board LED.

- Receiver C2
- Sender A
- Command LED on

The protocol implementation is responsible for communication features like blinking frequency or acknowledge responds.

Bmbed network provides the framework with a few standard message implementations. The user is free to expand its own protocol and messages.



2.2 Protocol Architecture

The Bmbed high level protocol definition is as simple as possible to make it easy to use and to expand. This protocol is valid for the entire network. Use it also for communication within a microcontroller between two sub-nodes.

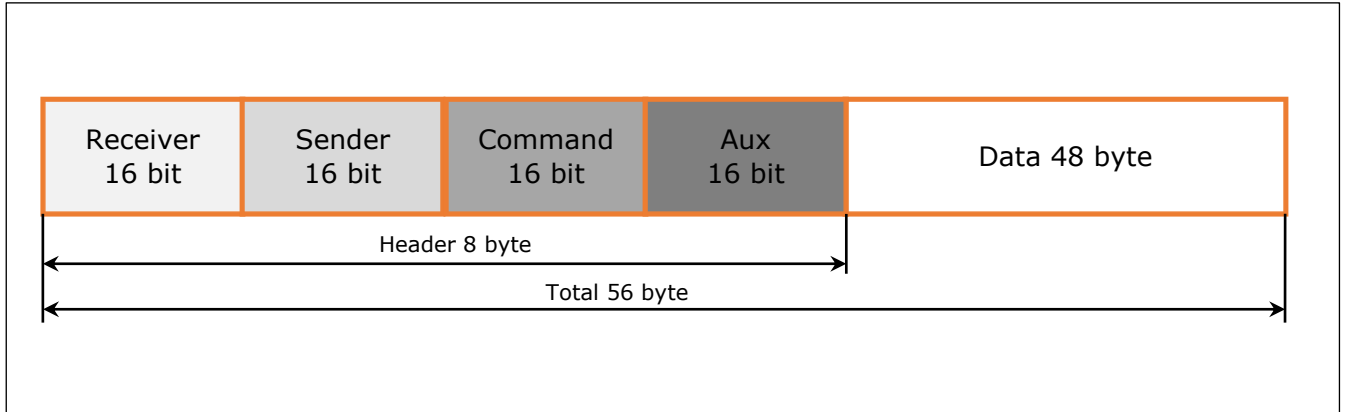


Figure 2 High Level Protocol Architecture

There is no length field in the protocol. This means that always 56 bytes are transferred whether used or not. Receiver, sender and command are mandatory with valid data at least.

The protocol will locally expand for lower levels e.g. CRC (cyclic redundancy check).

- UART
- SPI
- Wireless IEEE 802.15.4
- CAN
- MQTT

Look for expanded protocol details in the according interface documentation.



3 Protocol Dictionary

3.1 Definition

The **Bmbed** protocol dictionary must be valid for the entire **Bmbed** network. Different **Bmbed** systems can be connected. For that reason this is a standalone document without any system boundaries.

3.2 Nodes

Name	Code	Description
NET_NODE_F28069_BASE	0x0010	TI TMS320F28069 Piccolo controller on TI LAUNCHPAD-F28069 evaluation board
NET_NODE_F28069_XXXX	0x0011...0x001F	Internal F28069 sub-nodes
NET_NODE_F28027_BASE	0x0020	TI TMS320F28027 Piccolo controller on TI LAUNCHPAD-F28027 evaluation board
NET_NODE_F28027_XXXX	0x0021...0x002F	Internal F28027 sub-nodes
NET_NODE_K64F_BASE	0x0030	NXP K64F Cortex-M4 on FRDM-K64F evaluation board
NET_NODE_K64F_XXXX	0x0031...0x003F	Internal K64F sub-nodes
NET_NODE_KL02_BASE	0x0040	NXP KL02 Cortex-M0 on FRDM-KL02Z evaluation board
NET_NODE_KL02_XXXX	0x0041...0x004F	Internal KL02 sub-nodes
NET_NODE_CC2650_BASE_A	0x0050	TI CC2650 on TI LAUNCHPAD-CC2650 evaluation board
NET_NODE_CC2650_XXXX_A	0x0051...0x005F	Internal CC2650 sub-nodes
NET_NODE_CC2650_BASE_B	0x0060	TI CC2650 on TI LAUNCHPAD-CC2650 evaluation board
NET_NODE_CC2650_XXXX_B	0x0061...0x006F	Internal CC2650 sub-nodes
NET_NODE_CC2650_BASE_C	0x0070	TI CC2650 on TI LAUNCHPAD-CC2650 evaluation board
NET_NODE_CC2650_XXXX_C	0x0071...0x007F	Internal CC2650 sub-nodes
NET_NODE_STM32F4_BASE	0x0080	ST STM32F413 on TI NUCLEO-F413 evaluation board
NET_NODE_STM32F4_XXXX	0x0081...0x008F	Internal CC2650 sub-nodes
NET_NODE_STM32F0_BASE	0x0090	ST STM32F042 on TI NUCLEO-F042 evaluation board
NET_NODE_STM32F0_XXXX	0x0091...0x009F	Internal CC2650 sub-nodes
PC_BMBED_BASE_1000	0x1000	Windows® PC
PC_BMBED_XXXX_1000	0x1001...0x101F	Internal PC sub-nodes
PC_BMBED_BASE_2000	0x2000	Windows® PC
PC_BMBED_XXXX_2000	0x2001...0x201F	Internal PC sub-nodes
PC_BMBED_BASE_3000	0x3000	Windows® PC
PC_BMBED_XXXX_3000	0x3001...0x301F	Internal PC sub-nodes
PC_BMBED_BASE_4000	0x4000	Windows® PC
PC_BMBED_XXXX_4000	0x4001...0x401F	Internal PC sub-nodes



3.3 Commands

3.3.1 General

Name	Code	Data	Description
NET_EVENT_ALARM	0x0001	s32 value char[44] text	Alarm message
NET_EVENT_ERROR	0x0002	s32 value char[44] text	Error message
NET_EVENT_WARNING	0x0003	s32 value char[44] text	Warning message
NET_EVENT_INFORMATION	0x0004	s32 value char[44] text	Information message
NET_CMD_RESET_SET	0x0010	-	Perform reset on acc. Microcontroller
NET_CMD_RESET_ACK	0x0011	-	Acknowledge back
NET_CMD_FIRMWARE_REQ	0x0200	-	Firmware data request
NET_CMD_FIRMWARE_RES	0x0201	char[12] name char[12] version char[12] compDate char[12] compTime	Firmware respond message with name, version, compiler date, compiler time

3.3.2 Bootloader

Name	Code	Data	Description
NET_CMD_START_UPDATE_SET	0x1000	u32 size u16 magic u16 crc	Start command for an application upload. The application flash memory will be erased. Application size and magic number will be checked.
NET_CMD_START_UPDATE_ACK	0x1001	-	The acknowledge signals that the bootloader is ready to get the new application binary data.
NET_CMD_START_UPDATE_NACK	0x1002	-	Something went wrong with the flash erase or with the size or with the magic number.
NET_CMD_BINARY_DATA_SET	0x1003	u8[] bin data	Binary data row from the according application Intel© hex file.
NET_CMD_BINARY_DATA_ACK	0x1004	-	The data are programmed successfully.
NET_CMD_BINARY_DATA_NACK	0x1005	-	Something went wrong with the programming.
NET_CMD_END_UPDATE_SET	0x1006	-	After programming the application binary data, the end is marked with some security flags.
NET_CMD_END_UPDATE_ACK	0x1007	-	New application is ready and will be started directly.
NET_CMD_END_UPDATE_NACK	0x1008	-	The new application is now running. Something went wrong with the flag setting. Bootloader is still running.

3.3.3 Application

Name	Code	Data	Description
NET_CMD_START_UPDATE_SET	0x1000	u16 magic	Start command for an application upload. The current application checks the magic number and erases the security flags and execute a reset. After that the bootloader is running.
NET_CMD_START_UPDATE_ACK	0x1001	-	Successful update start.
NET_CMD_START_UPDATE_NACK	0x1002	-	Application is still running. Something went wrong.
NET_CMD_HARDWARE_REQ	0x2000	-	Hardware data request
NET_CMD_HARDWARE_RES	0x2001	char[12] HW name char[12] HW version char[12] HW date char[12] HW time	Hardware data response



NET_CMD_HARDWARE_SET	0x2002	char[12] HW name char[12] HW version char[12] HW date char[12] HW time	New hardware data setting. Will stored on the FLASH.
NET_CMD_HARDWARE_ACK	0x2003	-	Successful hardware data setting
NET_CMD_HARDWARE_NACK	0x2004	-	Something went wrong
NET_CMD_COMPANY_REQ	0x2010	-	Company data request
NET_CMD_COMPANY_RES	0x2011	char[48] text	Company data response
NET_CMD_COMPANY_SET	0x2012	char[48] text	New company data setting. Will stored on the FLASH.
NET_CMD_COMPANY_ACK	0x2013	-	Successful company data setting
NET_CMD_COMPANY_NACK	0x2014	-	Something went wrong
NET_CMD_DEVICE_REQ	0x2020	-	Device data request
NET_CMD_DEVICE_RES	0x2021	char[36] device name char[12] serie number	Device data response
NET_CMD_DEVICE_SET	0x2022	char[36] device name char[12] serie number	New device data setting. Will stored on the FLASH.
NET_CMD_DEVICE_ACK	0x2023	-	Successful device data setting
NET_CMD_DEVICE_NACK	0x2024	-	Something went wrong
NET_CMD_PRODUCTION_REQ	0x2030	-	Production data request
NET_CMD_PRODUCTION_RES	0x2031	char[12] plant name char[12] prod date char[12] prod time	Production data response
NET_CMD_PRODUCTION_SET	0x2032	char[12] plant name char[12] prod date char[12] prod time	New production data setting. Will stored on the FLASH.
NET_CMD_PRODUCTION_ACK	0x2033	-	Successful production data setting
NET_CMD_PRODUCTION_NACK	0x2034	-	Something went wrong
NET_CMD_PARAMETER_REQ	0x2050	-	Parameter data request
NET_CMD_PARAMETER_RES	0x2051	Integer: u16 number u16 format s32 min s32 max s32 measure char[10] unit char[10] result Float: u16 number u16 format f32 min f32 max f32 measure char[10] unit char[10] result	Parameter data response. Number: 1 2 3 4 Format 0: 32 bit integer Format 1: 32 bit float
NET_CMD_PARAMETER_SET	0x2052	Integer: u16 number u16 format s32 min s32 max s32 measure char[10] unit char[10] result Float: u16 number u16 format f32 min f32 max f32 measure char[10] unit char[10] result	New parameter data setting. Number: 1 2 3 4 Format 0: 32 bit integer Format 1: 32 bit float
NET_CMD_PARAMETER_ACK	0x2053	-	Successful parameter data setting
NET_CMD_PARAMETER_NACK	0x2054	-	Something went wrong
NET_CMD_LED_SET	0x2060	u32 tick1 u32 tick2	LED blink frequency setting. Depend on the board there is an LED and a timer tick.



		u32 tick3	
NET_CMD_TEMPERATURE_REQ	0x2070	-	MCU temperature request
NET_CMD_TEMPERATURE_RES	0x2071	f32 instant value f32 average value f32 minimum value f32 maximum value	MCU temperature response The average value is calculated with a moving average filter.
NET_CMD_DEBUG_REQ	0x2080	u16 number	Debug data request. Number 0..debug length (board depend)
NET_CMD_DEBUG_RES	0x2081	u16 number s16[] xVal s16[] yVal	Debug data respond. Number 0..debug length (board depend) xVal array: x length (board depend) yVal array: y length (board depend)



4 Manufacturer Information

Customer support and technical support for **Bmbed**



Eicher Engineering
Software Development
Frigadenstrasse 23
CH-8739 Rieden SG
Switzerland

www.bmbed.ch
bmbed@eichereng.ch

Bmbed High-Level-Protocol Developer Manual Version 1.0

© 2016-2019 by Eicher Engineering

The **Bmbed High-Level-Protocol** described in this manual is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from Eicher Engineering.

Disclaimer of Warranty

THE SOFTWARE AND DOCUMENTATION HAS PROVIDED 'AS IS.' TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, EICHER ENGINEERING DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE. ANY LIABILITY OF EICHER ENGINEERING WILL BE LIMITED EXCLUSIVELY TO REFUND OF THE PURCHASE PRICE.



5 Version History

Version	Type	Date	Author	Comments
1.0	released	2019-04-17	ME	Initial writing