Getting started with the AEK-MOT-TK200G1 evaluation board for car opening/closing systems

Introduction

The AEK-MOT-TK200G1 evaluation board has been developed to drive the opening/closing systems of the car power lift gates, ensuring high levels of safety and reliability.

The board offers the possibility of driving three different motors. Two motors raise and lower the car tailgate, while the third motor locks the trunk.

The board also allows driving two different high input capacity loads, that is, two strings of LEDs. One string lights up the interior of the trunk when it is open, while the other one lights up the car license plate.

The AEK-MOT-TK200G1 evaluation board can be used as a small ECU that meets the typical requirements of a car lift gate. The board hosts:

- a Chorus 1M ASIL-B microcontroller (SPC582B60E1), which communicates with the multimotor driver (L99DZ200G) through the SPI;
- an L99DZ200G device that controls all the loads connected to the board;
- a CAN connector that allows a domain controller to interact remotely with the AEK-MOT-TK200G1.

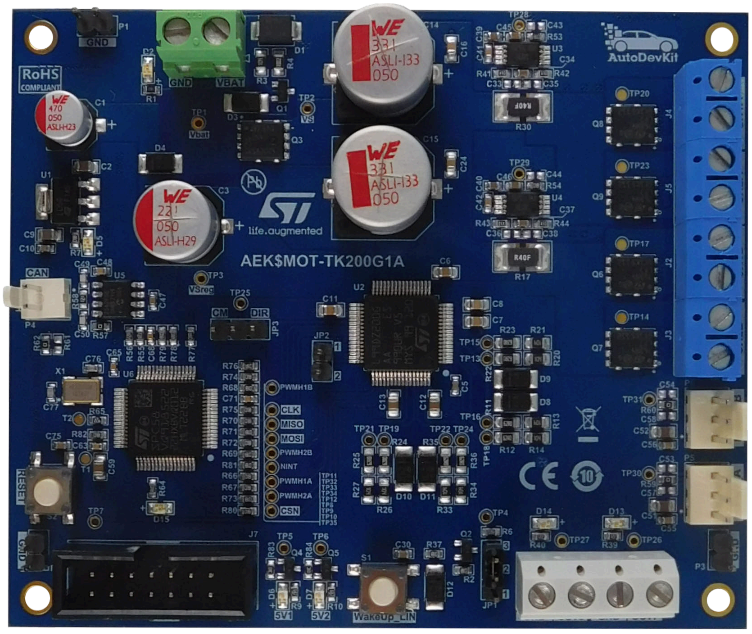
Moreover, to support DC motor positioning and increase actuation safety, a circuitry for the current sensing of the L99DZ200G H-bridges outputs has been added. On the board, this circuit is coupled by two connectors dedicated to Hall sensor feedback coming from the motors. These two features can be combined or used alternatively.

To achieve a higher level of safety, a service key mechanism is provided. This mechanism consists of a continuous communication interaction between the MCU and the L99DZ200G chip. This safety feature is concurrent with all the other communications between the MCU and the L99DZ200G.

For a proper management of the L99DZ200G, the service key mechanism must be satisfied within a configurable temporal window. When the service key fails, the L99DZ200G switches off all the outputs and enters the fail-safe mode. The service key temporal window can be configured in real-time, too.

Warning: *The AEK-MOT-TK200G1 evaluation board has not to be used in a vehicle as it is designed for R&D laboratory use only.*

Figure 1. AEK-MOT-TK200G1 evaluation board

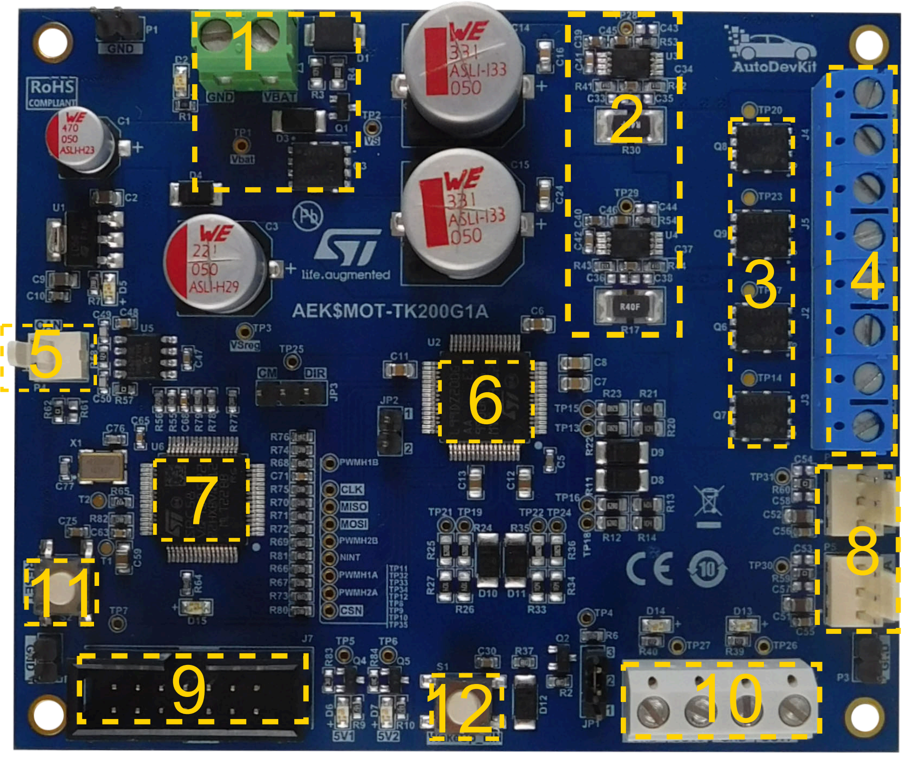


1 Hardware overview

1.1 Board main components

1. Power supply connector
2. Current sensing network based on the TSC103
3. STL76DN4LF7AG MOSFET for H-bridge. The MOSFET package contains a high-side and a low-side
4. Connector for a DC motor
5. CAN connector
6. L99DZ200G driver
7. SPC582B60E1 microcontroller
8. Connector for eventual Hall sensors
9. JTAG connector for MCU programming
10. Connector for the two high-side outputs
11. Reset button
12. WakeUp_LIN button

Figure 2. **AEK-MOT-TK200G1 evaluation board: main components**



1.1.1 SPC582B60E1

The AEK-MOT-TK200G1 evaluation board hosts a Chorus 1M SPC582B60E1 microcontroller that belongs to the SPC58 Chorus family.

The MCU is in charge of controlling the L99DZ200G driver.

The main MCU features are:

- AEC-Q100 qualified
- High performance e200z2 single core:
  - 32-bit Power Architecture technology CPU
  - Core frequency up to 80 MHz
- 1088 KB (1024 KB code flash memory + 64 KB data flash) on-chip flash memory: supports reading during program and erase operations, and multiple blocks allow performing the EEPROM emulation
- 96 KB on-chip general-purpose SRAM
- Comprehensive new generation ASIL-B safety concept:
  - ASIL-B of ISO 26262
  - FCCU for collection and reaction to failure notifications
  - Memory error management unit (MEMU) for collection and reporting of error events in the memories
- One enhanced 12-bit SAR analog-to-digital converter unit:
  - up to 27 channels (two channels for the power lift gate application to monitor the linear actuator position)
  - enhanced diagnostic features (such as current sensing current monitoring)
- Seven CAN interfaces
- Four serial peripheral interface (DSPI) modules (a DSPI is used for the communication between the MCU and the L99DZ200G chip).

Note: For further information, refer to RM0403 or to the SPC582Bx datasheet.

1.1.2 L99DZ200G

The L99DZ200G chip belongs to the STMicroelectronics "Door-Zone" family. It consists of a range of system ICs specifically designed to integrate in a single package all the main components and functions required to manage advanced automotive door applications.

The L99DZ200G is a multifunctional actuator driver. It is programmed by a microcontroller. Its main features include four half-bridges, seven high-side actuators, and two H-bridge drivers. Thanks to the H-bridge drivers (configurable in single or dual mode), the L99DZ200G is able to manage the spindle motors used to raise and lower the tailgate as well as the trunk lock. The L99DZ200G can also manage other typical loads located in the power trunk (for example, buzzer, LED, and bulb supplies).

The device standby state reduces the battery power consumption.

The L99DZ200G features available in the AEK-MOT-TK200G1 are:

- Two H-bridge drivers for the spindle motors and lock motor. The on-board connector has eight outputs to simplify the connection of the three motors. The outputs are doubled (that is, eight in total) as there are two outputs for each half bridge of the two H-bridges. This duplication facilitates the connection of the third motor in the middle of the two H-bridges
- Two high-side drivers for the LED modules
- One 5 V voltage regulator for the microcontroller supply
- One 5 V voltage tracker for the peripheral supply
- All the actuator outputs come with the following protection and supervisor features:
  - Current monitoring (high-side only)
  - Open-load and overcurrent
  - Thermal warning and shutdown
- Configurable window watchdog
- A/D conversion of supply voltages and internal temperature sensors

Some of the L99DZ200G features are not implemented in the AEK-MOT-TK200G1:

- On the board, we disabled the MCU programming via the LIN and CAN transceiver by connecting the LIN pin to Vsreg (12 V), while we left all the other pins related to CAN and LIN floating. To exit from a standby condition, the L99DZ200G state machine requires at least one of the two interfaces to see a 12 V to 0 V transition. We used the LIN one for this purpose. We also connected a wake-up button to the LIN pin to wake up the device from the standby condition.
- All the other half-bridges and high-side outputs are not connected and left floating.

Note: The AEK-MOT-TK200G1 hosts an MCU that is always active as it is not powered by the L99DZ200G. Therefore, the L99DZ200G standby status does not impact the MCU.

Note: For further information on the L99DZ200G, see the related datasheet.

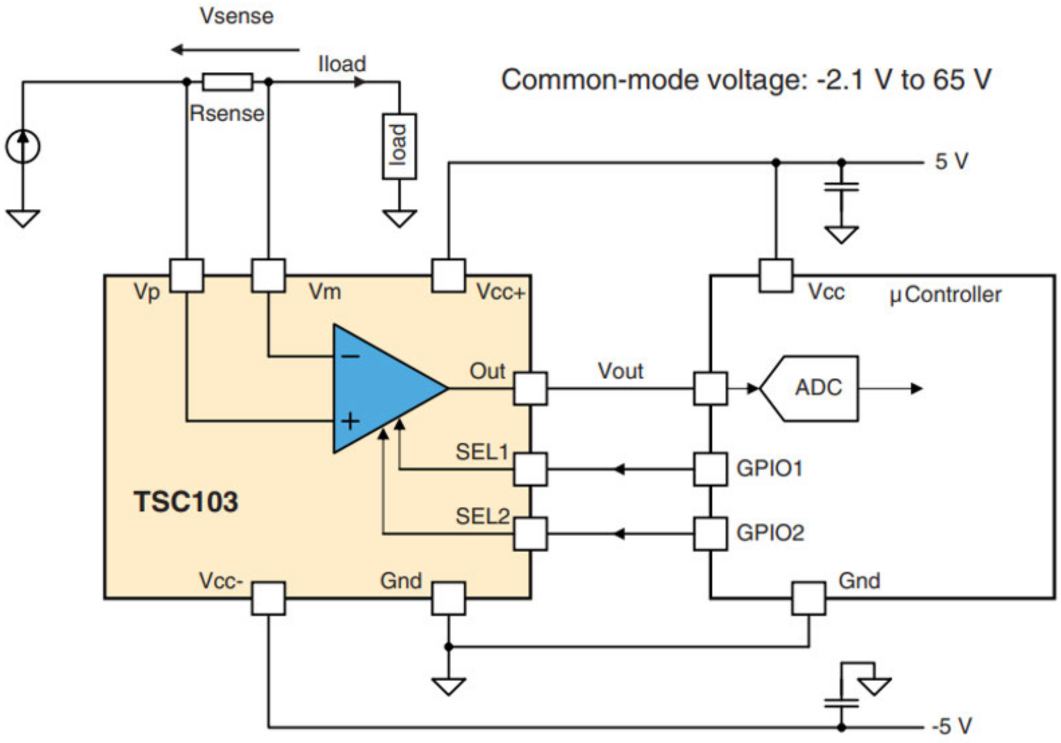
1.1.3 Current sensing monitoring network

The AEK-MOT-TK200G1 also features a current sensing network. This network senses the current that the DC motors absorb when connected to the AEK-MOT-TK200G1 H-bridges.

This motor current information can be used for rough position control and obstacle detection during the motor actuation.

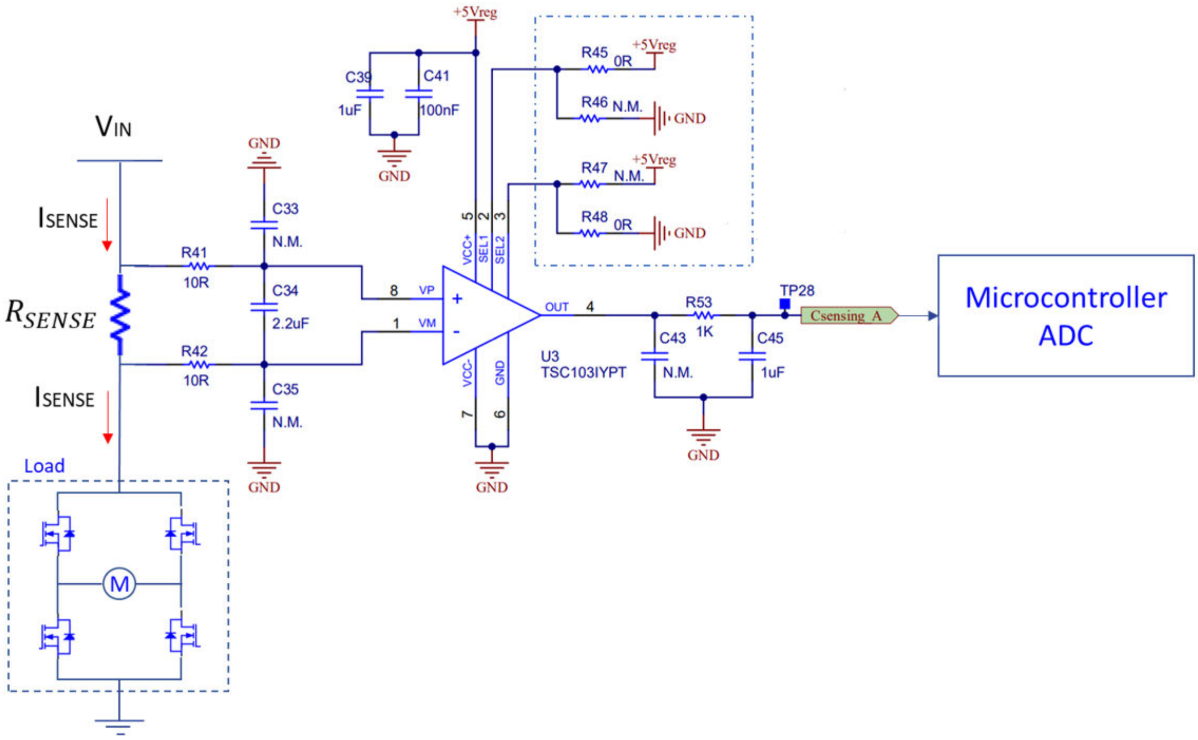
This current sensing network is based on the high-side current sensing topology, where the sensing resistor is located between the power supply and the load.

Figure 3. TSC103 current sense amplifier



The following figure shows the diagram of the current sensing network implemented, based on the TSC103IYPT current sense amplifier.

Figure 4. **TSC103 current sense amplifier**



This current amplifier gives the possibility of selecting four different gain levels:

- 25
- 50
- 75
- 100

The current sensing network on the board has been developed for the L12-50-100-12-P Actuonix DC motor.

Table 1. **Specifications of the L12-50-100-12-P Actuonix DC motor**

Characteristic	Value
Maximum input voltage	12 V
Stall current	250 mA
Back drive force (static)	22 N
Closed length (hole to hole)	102 mm
Maximum speed (no load)	13 mm/s
Maximum force	42 N

The computed Rsense soldered on the board is 0.4 Ohms.

Note: Section 6 describes how to change the sensing network according to the chosen motor characteristics.

1.1.4 CAN connector and potentiometer connectors

The AEK-MOT-TK200G1 additional features are:

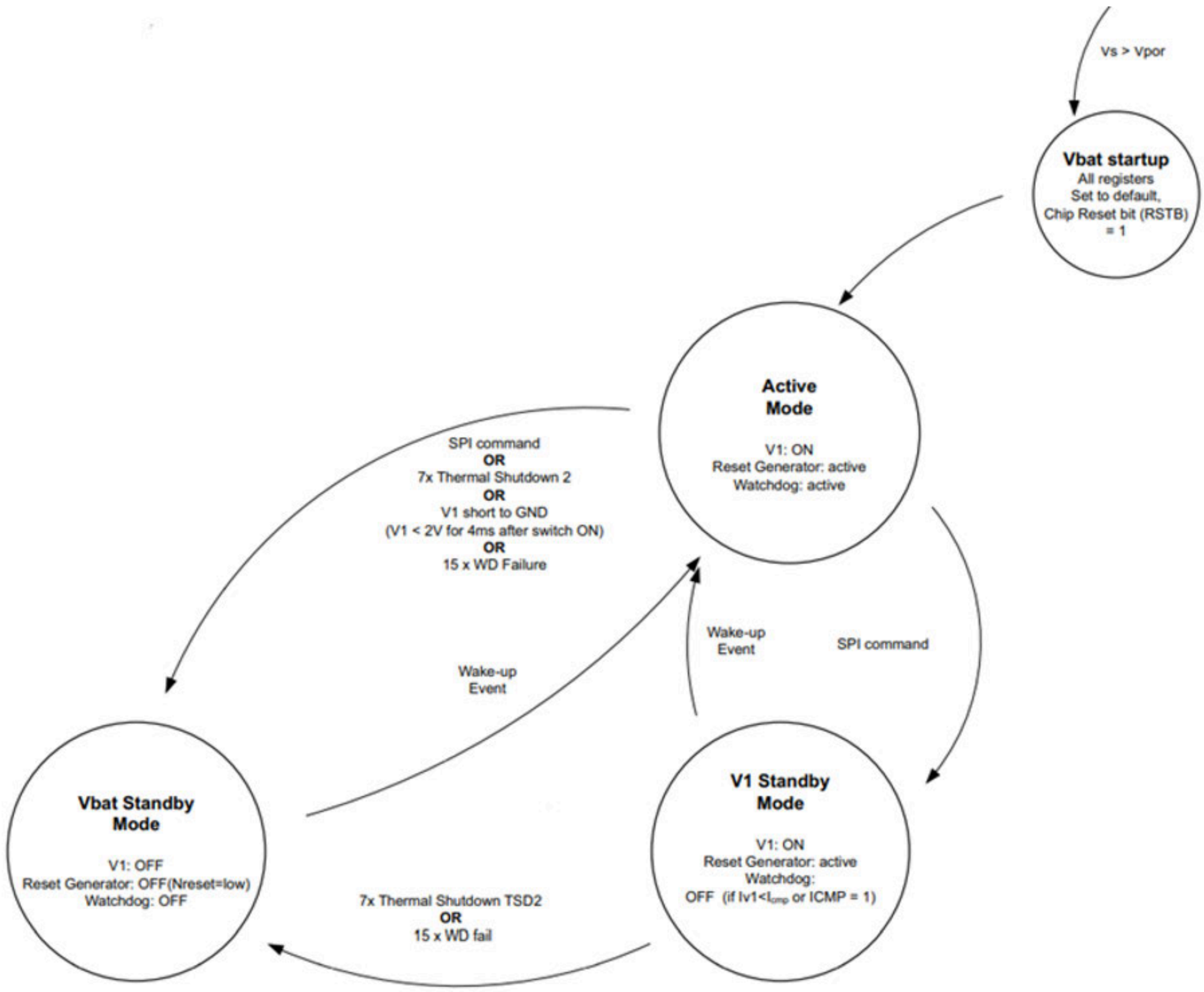
- A CAN connector for an external domain controller to drive the board via CAN messages, that is, to manage the opening/closing of the trunk. The CAN network has been compensated with a 120 Ohm resistor as per CAN bus specification

- The connectors for motor Hall sensor feedback. The sensors achieve an accurate motor positioning and increase the actuation reliability

1.1.5 L99DZ200G state machine

As we are not using all the features of the L99DZ200G, the finite state machine (FSM) of the chip is simplified as shown below.

Figure 5. **Simplified L99DZ200G FSM**



The main states are:

- **Vbat_startup**: the L99DZ200G enters this state when VS > VPOR. All the registers are set to the default value. After about 0.1 milliseconds, the L99DZ200G enters the active mode.
- **Active mode**: to keep the device in the active state, the MCU activates a watchdog that monitors the communication between the microcontroller and the chip. In this state, all the outputs are active, including the H-bridge driver.
- **V1_Standby**: the transition from the active mode to V1_Standby mode is controlled through an SPI message or it is a consequence of loosing the watchdog signal.
- **VBAT_Standby**: the L99DZ200G enters this state in case of:
  - multiple watchdog failures
  - multiple thermal shutdowns
  - V1 regulator failures
  - an explicit SPI command

- To exit from the VBAT_Standby state and return to the active one, press the wake-up button on the LIN pin or drive the MCU pin no. 58 (PIN_WAKEUP) from low to high. You can perform this second option only if:
  - a jumper on JP1 connects pins two and three
  - the wake-up pin is configured as an input pin in the L99DZ200G
  - OUT15 is on and it is not driven through an internally generated PWM signal

Note: There is a software debug mode that simplifies the debugging procedure. In this mode, the watchdog requirement is turned off. For further details on how to enter the debug mode, refer to Section 3.5 How to execute SW debug for the AEK-MOT-TK200G1.

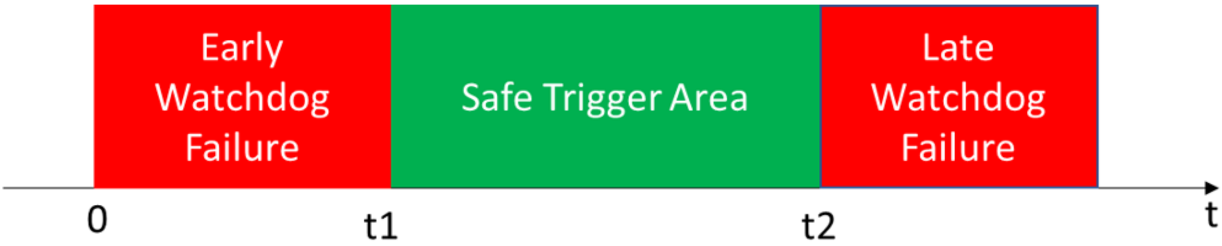
1.1.6 Watchdog scheme

The device state machine provides a state transition from the active mode to the standby mode when the continuity of communication between the device and the microcontroller is lost. This continuity of communication must be guaranteed by writing SPI messages into a special register that toggles a specific bit (bit 0 of the control register CR1 or Config Reg).

After the power-on or the standby mode, the watchdog has to start within a maximum timeout (Long Open Window TLW). The time window gives the microcontroller the time to run its own setup before starting the watchdog. From this moment, the microcontroller has to serve the watchdog within a safe triggering time range. The trigger time window is configurable by SPI, both at startup and runtime.

The watchdog failure happens if the watchdog trigger occurs before the t1, in the “early write" window, or after t2, in the "late write" window. In case of watchdog failures, a reset signal is sent to the MCU.

Figure 6. Watchdog timing 1



Note: For further details, refer to the L99DZ200G documentation on www.st.com.

2 Software overview

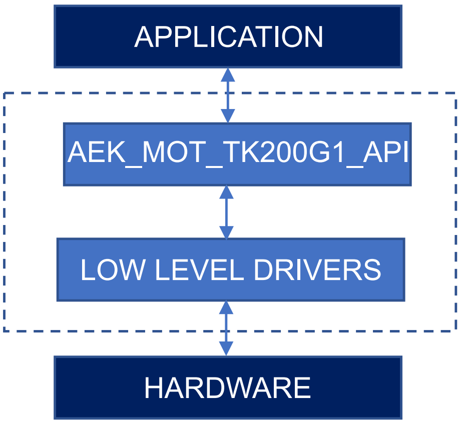
The AEK-MOT-TK200G1 software structure enhances reuse and simplifies maintenance. It also reduces the prototyping time.

Thus, we have implemented a layered architecture that embeds the following blocks:

- Low-level drivers
- AEK_MOT_TK200G1_API

The library is written in embedded C code.

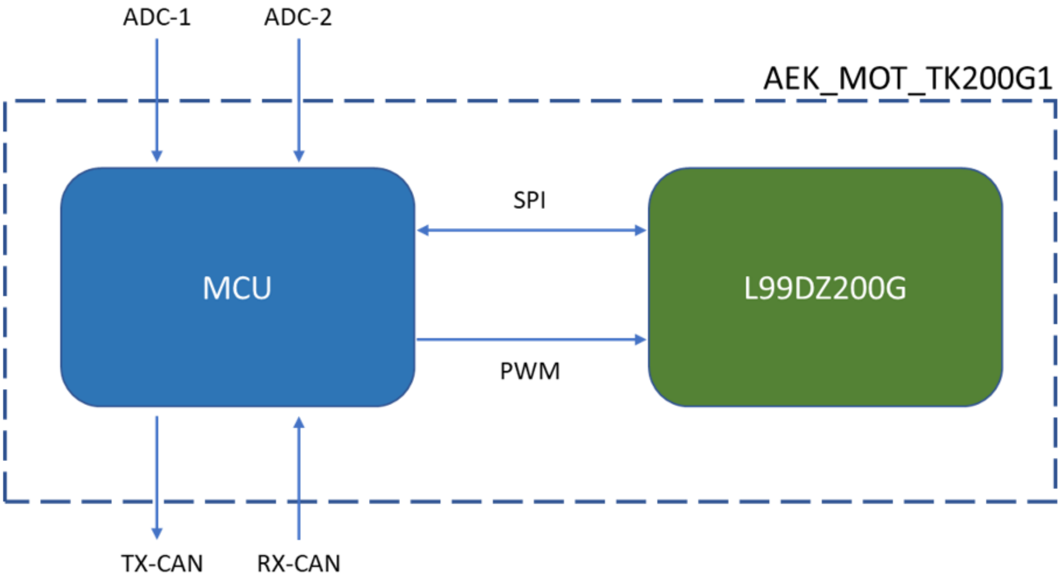
Figure 7. Software architecture



2.1 Low-level drivers

The low-level drivers interface with the AEK-MOT-TK200G1 board. They support all the MCU peripherals (CAN, SPI, PWM, and GPIO).

Figure 8. Low-level drivers



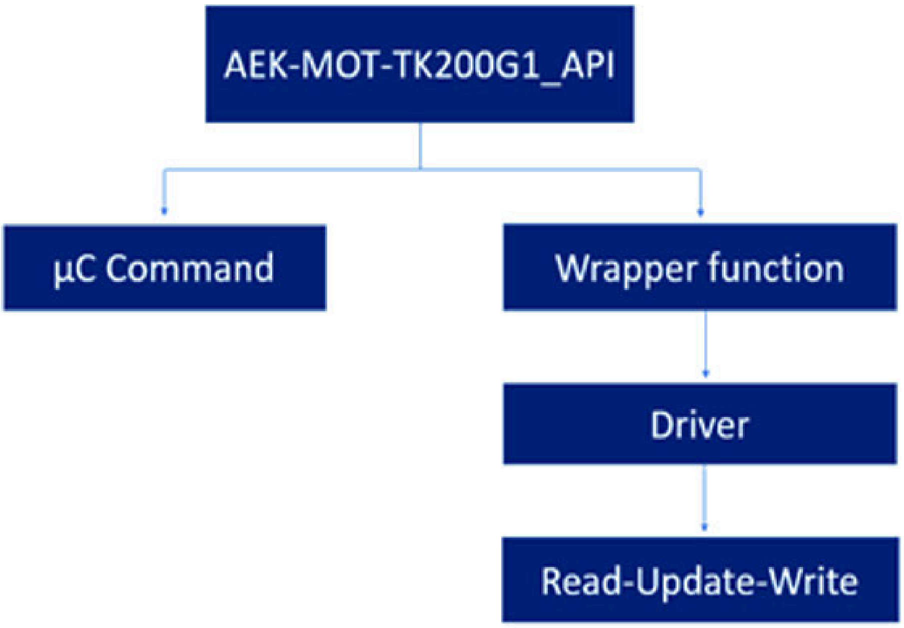
The AEK_MOT_TK200G1_API software is based on the following peripherals:

- SPI: to implement the bidirectional communication between the MCU and the L99DZ200G
- Programmable interrupt timer (PIT): to trigger the watchdog
- PWM: to generate motor-driving signals
- CAN: to manage the messages received or transmitted by other ECUs
- ADC: to convert the signal coming from the Hall sensors

2.2 AEK_MOT_TK200G1_API

The purpose of the API implemented is to expose all the functions of the AEK-MOT-TK200G1 board. The figure below shows the API functional blocks.

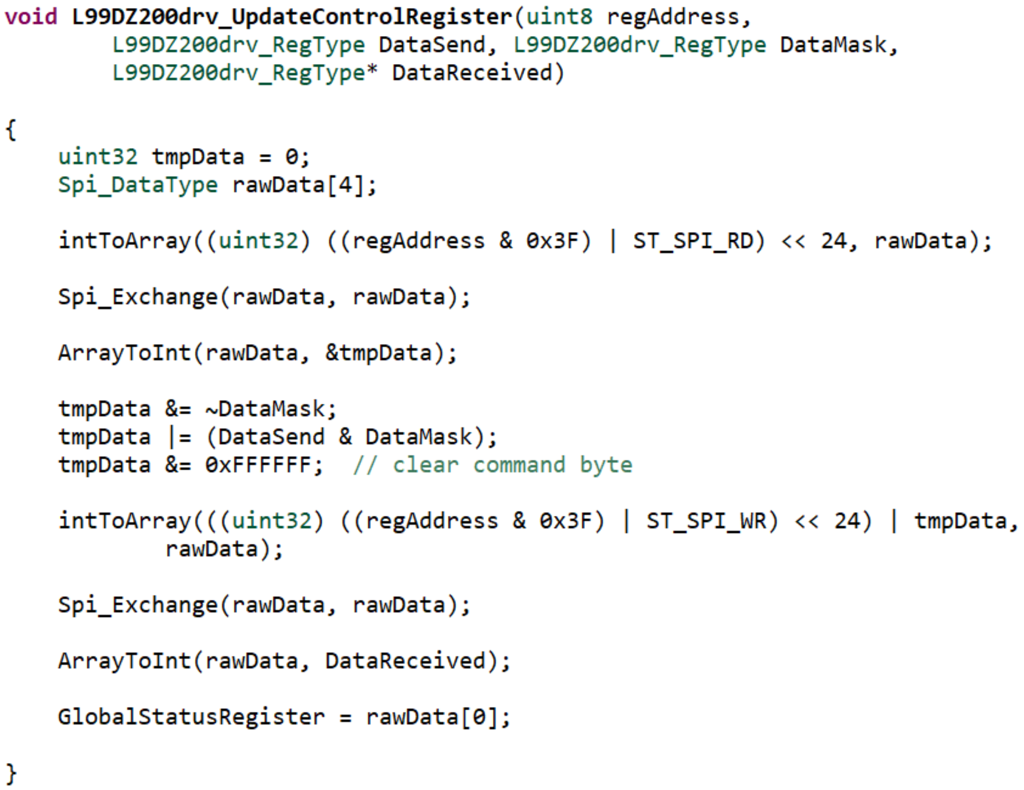
Figure 9. API architecture



The main blocks are:

- The µC command block that wraps the low-level drivers for the MCU configuration
- The "Read, Update, Write", which are low-level functions responsible for updating the L99DZ200G registers. The input parameters are the register address, the data to write/update, the register mask to apply, and the variable address to contain the data read. The return is the global status register value, which represents the status and possible faults of the L99DZ200G device. The most common errors are:
  - SPI error: related to SPI communication
  - Fail-safe: error related to internal fault like watchdog failure

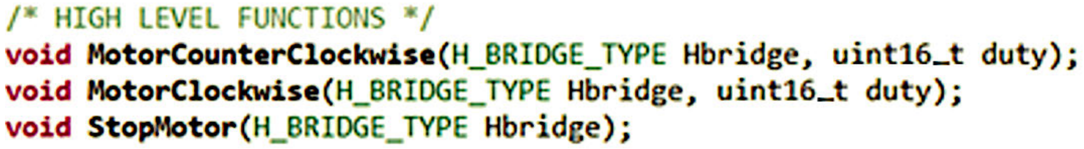
Figure 10. Update register function



```
void L99DZ200drv_UpdateControlRegister(uint8 regAddress,
        L99DZ200drv_RegType DataSend, L99DZ200drv_RegType DataMask,
        L99DZ200drv_RegType* DataReceived)

{
    uint32 tmpData = 0;
    Spi_DataType rawData[4];

    intToArray((uint32) ((regAddress & 0x3F) | ST_SPI_RD) << 24, rawData);

    Spi_Exchange(rawData, rawData);

    ArrayToInt(rawData, &tmpData);

    tmpData &= ~DataMask;
    tmpData |= (DataSend & DataMask);
    tmpData &= 0xFFFFFF;  // clear command byte

    intToArray(((uint32) ((regAddress & 0x3F) | ST_SPI_WR) << 24) | tmpData,
            rawData);

    Spi_Exchange(rawData, rawData);

    ArrayToInt(rawData, DataReceived);

    GlobalStatusRegister = rawData[0];

}
```

- The "driver functions" cover all L99DZ200G features and significantly simplify the usage of the device by masking the information of the register details.
- The wrapper functions expose at the AEK-MOT-TK200G1 level features (for example, motor driving)

Figure 11. Wrapper functions



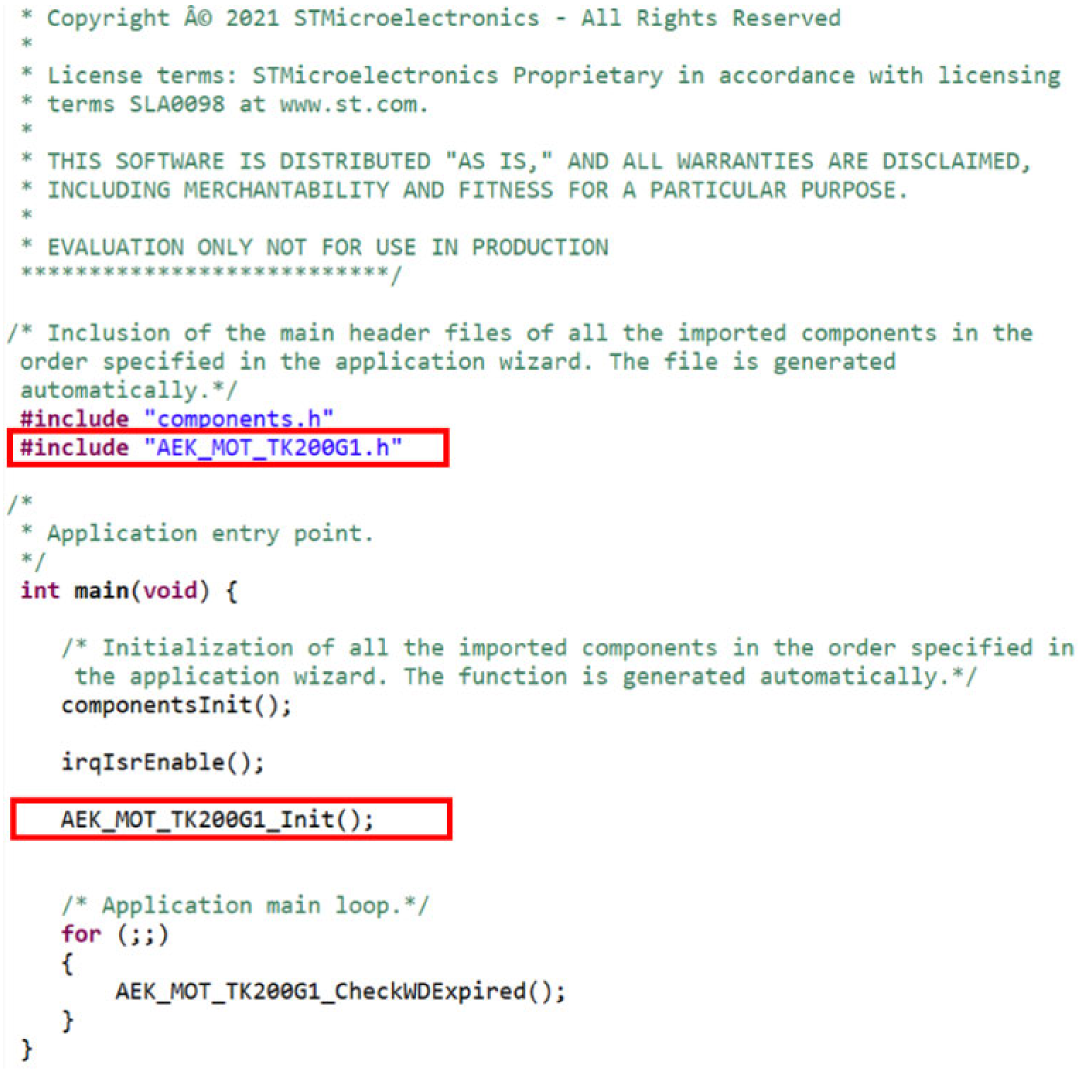
```
/* HIGH LEVEL FUNCTIONS */
void MotorCounterClockwise(H_BRIDGE_TYPE Hbridge, uint16_t duty);
void MotorClockwise(H_BRIDGE_TYPE Hbridge, uint16_t duty);
void StopMotor(H_BRIDGE_TYPE Hbridge);
```

Important: *`BOARD_STATUS_TYPE` maps the current state of the board at runtime, while `BOARD_CONFIG_TYPE` stores the board configuration in the flash memory for a future use.*

To include the above APIs in an application code, include the AEK_MOT_TK200G1.h header file in the main.c file.

To initialize the library, include the `Sel_mot_tk200g1_init()` function in your code.

Figure 12. Sel_mot_tk200g1_init() function



```
 * Copyright Â© 2021 STMicroelectronics - All Rights Reserved
 *
 * License terms: STMicroelectronics Proprietary in accordance with licensing
 * terms SLA0098 at www.st.com.
 *
 * THIS SOFTWARE IS DISTRIBUTED "AS IS," AND ALL WARRANTIES ARE DISCLAIMED,
 * INCLUDING MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
 *
 * EVALUATION ONLY NOT FOR USE IN PRODUCTION
 ***************************/

/* Inclusion of the main header files of all the imported components in the
 order specified in the application wizard. The file is generated
 automatically.*/
 #include "components.h"
 #include "AEK_MOT_TK200G1.h"

/*
 * Application entry point.
 */
 int main(void) {

    /* Initialization of all the imported components in the order specified in
     the application wizard. The function is generated automatically.*/
    componentsInit();

    irqIsrEnable();

    AEK_MOT_TK200G1_Init();


    /* Application main loop.*/
    for (;;)
    {
        AEK_MOT_TK200G1_CheckWDExpired();
    }
 }
```

2.3 Safety mechanism: watchdog trigger

The AEK-MOT-TK200G1 features a safety mechanism. The software portion of this safety mechanism includes a service key (that is, a watchdog) implemented with a programmable interrupt timer (PIT).

When the PIT expires, a global state variable is updated through the interrupt associated callback. The value of this global variable is evaluated within each implemented API function. When this value confirms that the trigger time has expired, the bit toggling SPI message is sent. This global state variable mechanism avoids the "missing trigger" fault. It prevents overwriting the L99DZ200G CR1 register that would compromise the watchdog.

The `AEK_MOT_TK200G1_CheckWDExpired()` function is used for watchdog expiry check and it is included in all the library APIs.

Moreover, in the user application code, a proper watchdog triggering has to be maintained. Therefore, the `AEK_MOT_TK200G1_CheckWDExpired()` function has to be invoked in all the code portions that potentially require an execution time comparable to the triggering time window (for example, a loop).

3 AEK-MOT-TK200G1 in AutoDevKit

The driver for the AEK-MOT-TK200G1 board is part of the AutoDevKit ecosystem.

An AutoDevKit component for the AEK-MOT-TK200G1 board has not been created, as the board hosts an MCU, and it is similar to a small ECU.

In AutoDevKit, we have included some AEK-MOT-TK200G1 evaluation demos.

The developed L99DZ200G driver is included in all the demos. They represent a very good starting point for user's development.

3.1 AutoDevKit ecosystem

The application development employing the AEK-MOT-TK200G1 takes full advantage of the AutoDevKit ecosystem, whose basic components are:

- SPC5-STUDIO integrated development environment (IDE)
- AutoDevKit software library (STSW-AUTODEVKIT)
- PLS UDE programmer and debugger

3.1.1 SPC5-STUDIO

SPC5-STUDIO is an integrated development environment (IDE) based on Eclipse designed to assist the development of embedded applications based on SPC5 Power Architecture 32-bit microcontrollers.

The package includes an application wizard to initiate projects with all the relevant components and key elements required to generate the final application source code. It also contains straightforward software examples for each MCU peripheral.

SPC5-STUDIO also features:

- the possibility of integrating other software products from the standard Eclipse marketplace
- free license GCC GNU C Compiler component
- support for industry-standard compilers
- support for multi-core microcontrollers
- PinMap editor to facilitate MCU pin configuration

Download the SPC5-UDESTK software to run and debug applications created with SPC5-STUDIO.

3.1.2 STSW-AUTODEVKIT

The STSW-AUTODEVKIT plug-in for Eclipse extends SPC5-STUDIO for automotive and transportation applications.

STSW-AUTODEVKIT features:

- integrated hardware and software components, component compatibility checking, and MCU and peripheral configuration tools
- the possibility of creating new system solutions from existing ones by adding or removing compatible function boards
- new code can be generated immediately for any compatible MCU
- high-level application APIs to control each functional component

The GUI helps configure interfaces, including SPI, and can automatically manage all relevant pin allocation and deallocation operations.

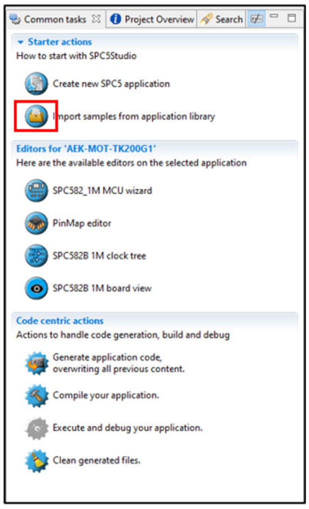
For more information, refer to UM2623 (in particular, Section 6 and Section 7) or watch the video tutorials.

Note: *AutoDevKit does not have a dedicated component for the AEK-MOT-TK200G1 board, but includes some demos containing the board drivers.*

3.2 How to download demos from SPC5Studio and AutoDevKit

After downloading and installing SPC5-STUDIO and AutoDevKit, you can import the application samples related to the AEK-MOT-TK200G1 evaluation board, as per the procedure below.

Step 1. From the [**Common task**] panel, click on the [**Import sample from application library**] icon.



Common tasks | Project Overview | Search

Starter actions

How to start with SPC5Studio

Create new SPC5 application

Import samples from application library

Editors for 'AEK-MOT-TK200G1'

Here are the available editors on the selected application

SPC582_1M MCU wizard

PinMap editor

SPC582B 1M clock tree

SPC582B 1M board view

Code centric actions

Actions to handle code generation, build and debug

Generate application code, overwriting all previous content.

Compile your application.

Execute and debug your application.

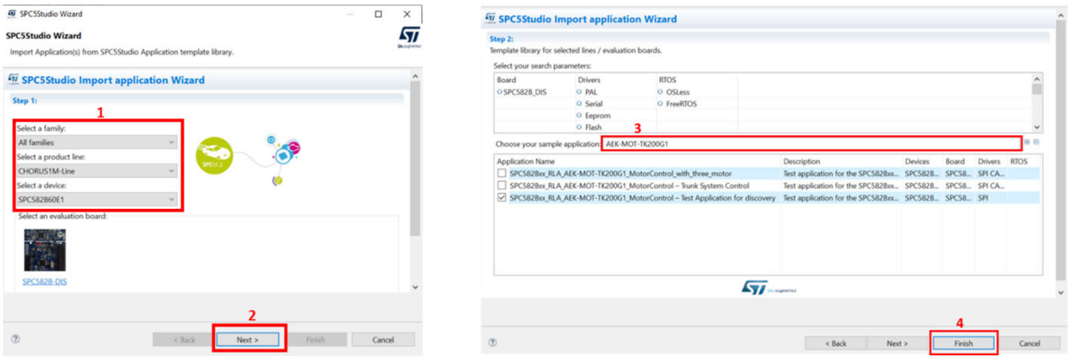
Clean generated files.

Figure 13. **Import sample from application library**

Step 2. In the Visual Studio Wizard, from the drop-down menu, select the family, the product, and the device. Then, click on the next button.

Step 3. Type "AEK-MOT-TK200G1" in [**Choose your sample application**] text-box.

Step 4. Tick the demo to import and click on the finish button.



SPC5Studio Wizard

SPC5Studio Wizard

Import Application(s) from SPC5Studio Application template library.

SPC5Studio Import application Wizard

Step 1:

1

Select a family: All families

Select a product line: CHORUS1M-Line

Select a device: SPC582B60E1

Select an evaluation board:

SPC582B-DIS

2

< Back | Next > | Finish | Cancel

SPC5Studio Import application Wizard

Step 2:

Template library for selected lines / evaluation boards.

Select your search parameters:

Board	Drivers	RTOS
○ SPC582B_DIS	○ PAL	○ OSLess
	○ Serial	○ FreeRTOS
	○ Eeprom	
	○ Flash	

3

Choose your sample application: AEK-MOT-TK200G1

Application Name	Description	Devices	Board	Drivers	RTOS
□ SPC582Bxx_RLA_AEK-MOT-TK200G1_MotorControl_with_three_motor	Test application for the SPC582Bxx...	SPC582B...	SPC58...	SPI CA...	
□ SPC582Bxx_RLA_AEK-MOT-TK200G1_MotorControl – Trunk System Control	Test application for the SPC582Bxx...	SPC582B...	SPC58...	SPI CA...	
☑ SPC582Bxx_RLA_AEK-MOT-TK200G1_MotorControl – Test Application for discovery	Test application for the SPC582Bxx...	SPC582B...	SPC58...	SPI	

4

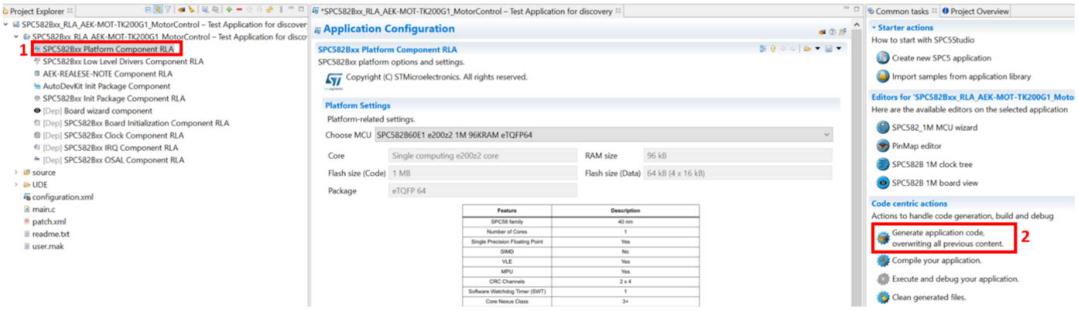
< Back | Next > | Finish | Cancel

Figure 14. **MCU and demo selection**

3.3 How to locate the AEK-MOT-TK200G1 SW library in the demo

Step 1. After importing a demo into SPC5-STUDIO, run the code generation.

Step 2. Select the platform component and click on the `Generate application Code` icon.



Project Explorer

- SPC582Bxx_RLA_AEK-MOT-TK200G1_MotorControl – Test Application for discover
  - SPC582Bxx_RLA_AEK-MOT-TK200G1_MotorControl – Test Application for disco
    - SPC582Bxx Platform Component RLA
    - SPC582Bxx Low Level Drivers Component RLA
    - AEK-REALESE-NOTE Component RLA
    - AutoDevKit Init Package Component
    - SPC582Bxx Init Package Component RLA
    - [Dep] Board wizard component
    - [Dep] SPC582Bxx Board Initialization Component RLA
    - [Dep] SPC582Bxx Clock Component RLA
    - [Dep] SPC582Bxx IRQ Component RLA
    - [Dep] SPC582Bxx OSAL Component RLA
  - source
  - UDE
  - configuration.xml
  - main.c
  - patch.xml
  - readme.txt
  - user.mak

*SPC582Bxx_RLA_AEK-MOT-TK200G1_MotorControl – Test Application for discovery

Application Configuration

SPC582Bxx Platform Component RLA

SPC582Bxx platform options and settings.

Copyright (C) STMicroelectronics. All rights reserved.

Platform Settings

Platform-related settings.

Choose MCU: SPC582B60E1 e200z2 1M 96KRAM eTQFP64

Core: Single computing e200z2 core

RAM size: 96 kB

Flash size (Code): 1 MB

Flash size (Data): 64 kB (4 x 16 kB)

Package: eTQFP 64

Feature	Description
SPC58 family	40 nm
Number of Cores	1
Single Precision Floating Point	Yes
SIMD	No
VLE	Yes
MPU	Yes
CRC Channels	2 x 4
Software Watchdog Timer (SWT)	1
Core Nexus Class	3+

Common tasks | Project Overview

Starter actions

How to start with SPC5Studio

- Create new SPC5 application
- Import samples from application library

Editors for 'SPC582Bxx_RLA_AEK-MOT-TK200G1_Moto

Here are the available editors on the selected application

- SPC582_1M MCU wizard
- PinMap editor
- SPC582B 1M clock tree
- SPC582B 1M board view

Code centric actions

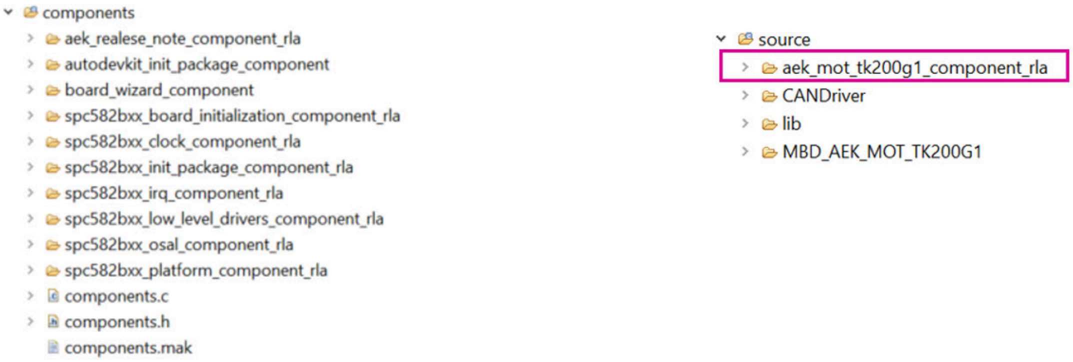
Actions to handle code generation, build and debug

- Generate application code, overwriting all previous content.
- Compile your application.
- Execute and debug your application.
- Clean generated files.

Figure 15. **Code generation**

The code generation produces a new folder in the demo project called 'component'. This folder holds the libraries of the SPC5-STUDIO and AutoDevKit components associated with the project, that is, the low-level driver library, the interrupt request queue library, and the board initialization library. The AEK-MOT-TK200G1 driver is in the aek_mot_tk200g1_component_rla subfolder under the source folder.

Figure 16. **Driver folders**



- components
  - aek_realese_note_component_rla
  - autodevkit_init_package_component
  - board_wizard_component
  - spc582bxx_board_initialization_component_rla
  - spc582bxx_clock_component_rla
  - spc582bxx_init_package_component_rla
  - spc582bxx_irq_component_rla
  - spc582bxx_low_level_drivers_component_rla
  - spc582bxx_osal_component_rla
  - spc582bxx_platform_component_rla
  - components.c
  - components.h
  - components.mak

- source
  - aek_mot_tk200g1_component_rla
  - CANDriver
  - lib
  - MBD_AEK_MOT_TK200G1

Step 3. To create from scratch a project that requires the AEK-MOT-TK200G1, copy the aek_mot_tk200g1_component_rla folder with all its content from a demo project and paste it under the source folder of the new project.

3.4 How to configure the low-level drivers

The configuration of the low-level drivers is mandatory. This configuration is simplified by the [**Configuration Application**] tab in SPC5-STUDIO (refer to UM2623, paragraph 7.4). The low-level drivers to be configured for the AEK-MOT-TK200G1 board are:

- SPI: to implement the bidirectional communication between the MCU and the L99DZ200G
- Programmable interrupt timer (PIT): to trigger the watchdog
- PWM: to generate the motor driving signal
- CAN: to manage the messages received or transmitted by other ECUs
- ADC: to convert the signal coming from the Hall sensors

To enable the low-level drivers, follow the procedure below.

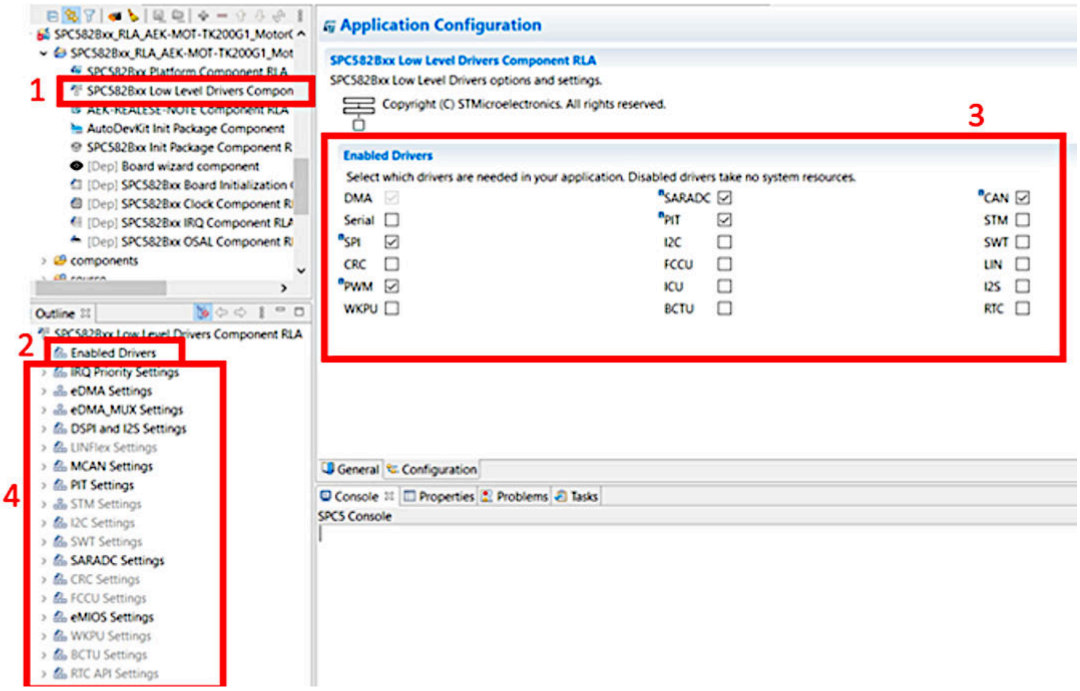
Step 1. Select [**Low level driver Component**].

Step 2. Select [**Enabled Drivers**] from the outline tab.

Step 3. Selects the low-level drivers to enable.

In the [**Outline**] tab, the enabled drivers become selectable.

Figure 17. Enabling low-level drivers



SPC582Bxx_RLA_AEK-MOT-TK200G1_Motor(
SPC582Bxx_RLA_AEK-MOT-TK200G1_Mot
SPC582Bxx Platform Component RLA
SPC582Bxx Low Level Drivers Compon
AEK-REALESE-NOTE Component RLA
AutoDevKit Init Package Component
SPC582Bxx Init Package Component R
[Dep] Board wizard component
[Dep] SPC582Bxx Board Initialization (
[Dep] SPC582Bxx Clock Component R
[Dep] SPC582Bxx IRQ Component RLA
[Dep] SPC582Bxx OSAL Component R
components

Outline

SPC582Bxx Low Level Drivers Component RLA
- Enabled Drivers
- IRQ Priority Settings
- eDMA Settings
- eDMA_MUX Settings
- DSPI and I2S Settings
- LINFlex Settings
- MCAN Settings
- PIT Settings
- STM Settings
- I2C Settings
- SWT Settings
- SARADC Settings
- CRC Settings
- FCCU Settings
- eMIOS Settings
- WKPU Settings
- BCTU Settings
- RTC API Settings

Application Configuration

SPC582Bxx Low Level Drivers Component RLA

SPC582Bxx Low Level Drivers options and settings.

Copyright (C) STMicroelectronics. All rights reserved.

Enabled Drivers

Select which drivers are needed in your application. Disabled drivers take no system resources.

DMA	☑	SARADC	☑	CAN	☑
Serial	□	PIT	☑	STM	□
SPI	☑	I2C	□	SWT	□
CRC	□	FCCU	□	LIN	□
PWM	☑	ICU	□	I2S	□
WKPU	□	BCTU	□	RTC	□

Step 4. To configure each of the enabled driver, select and double-click the driver in the [**Outline**] tab to open the corresponding configuration dialog.

Since the microcontroller pins are already wired on the board, the possible changes related to the configurations of the low-level drivers concern only some parameters, like the baud rate in the SPI, the frequency of the PIT, and the names of the callback functions. Some of the key configurations are described in the following paragraphs.

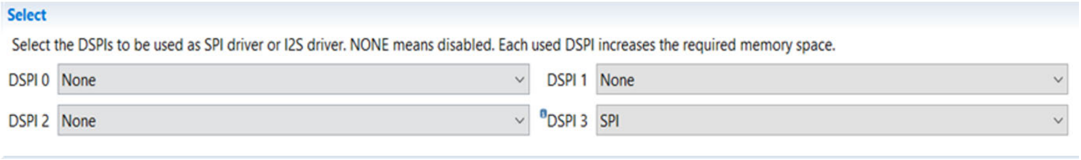
3.4.1 SPI configuration

To configure the SPI according to the AEK-MOT-TK200G1 hardware requirements, follow the procedure below.

Step 1. Double-click on [**DSPI and I2S Settings**] in the [**Outline**] tab.

Step 2. Choose the DSPI3 (DSPI stands for SPI) in the [**Select**] section.

Figure 18. SPI selection



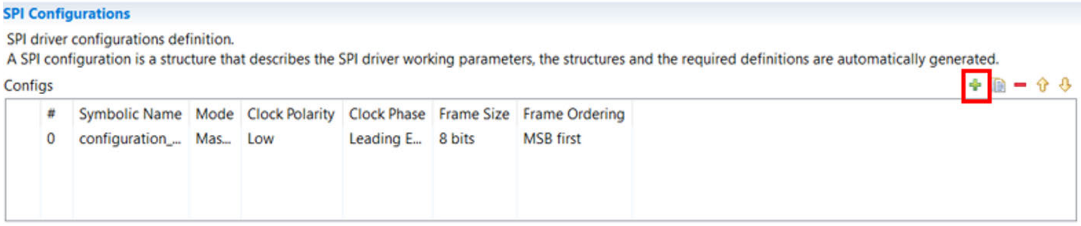
Select

Select the DSPIs to be used as SPI driver or I2S driver. NONE means disabled. Each used DSPI increases the required memory space.

DSPI 0	None	DSPI 1	None
DSPI 2	None	DSPI 3	SPI

Step 3. In the [**SPI configurations**] section, click on the [**+**] button to add a row and configure the SPI port selected in the previous step.

Figure 19. SPI configuration list



SPI Configurations

SPI driver configurations definition.
A SPI configuration is a structure that describes the SPI driver working parameters, the structures and the required definitions are automatically generated.

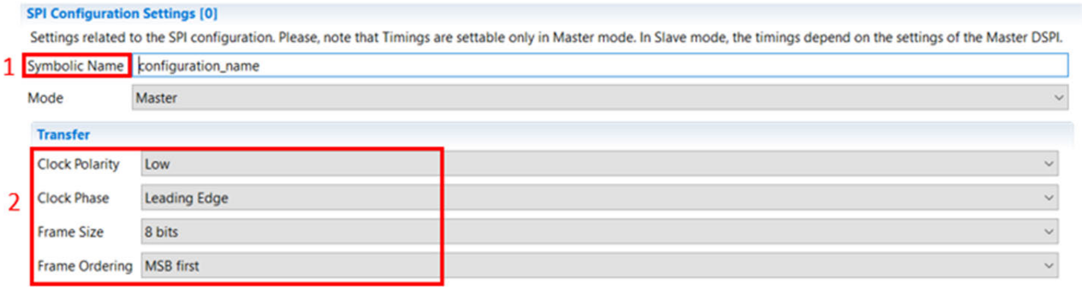
Configs

#	Symbolic Name	Mode	Clock Polarity	Clock Phase	Frame Size	Frame Ordering
0	configuration_...	Mas...	Low	Leading E...	8 bits	MSB first

Step 4. Double-click on the row just added.

Step 5. Add the name that you want to give to the configuration in the [**Symbolic Name**] field.
For example, type "configuration_name”. In addition, to set up the communication with the L99DZ200G device properly, set the parameters in the [**Transfer**] section as shown below.

Figure 20. SPI transfer configuration



SPI Configuration Settings [0]

Settings related to the SPI configuration. Please, note that Timings are settable only in Master mode. In Slave mode, the timings depend on the settings of the Master DSPI.

Symbolic Name: configuration_name

Mode: Master

Transfer

Clock Polarity: Low

Clock Phase: Leading Edge

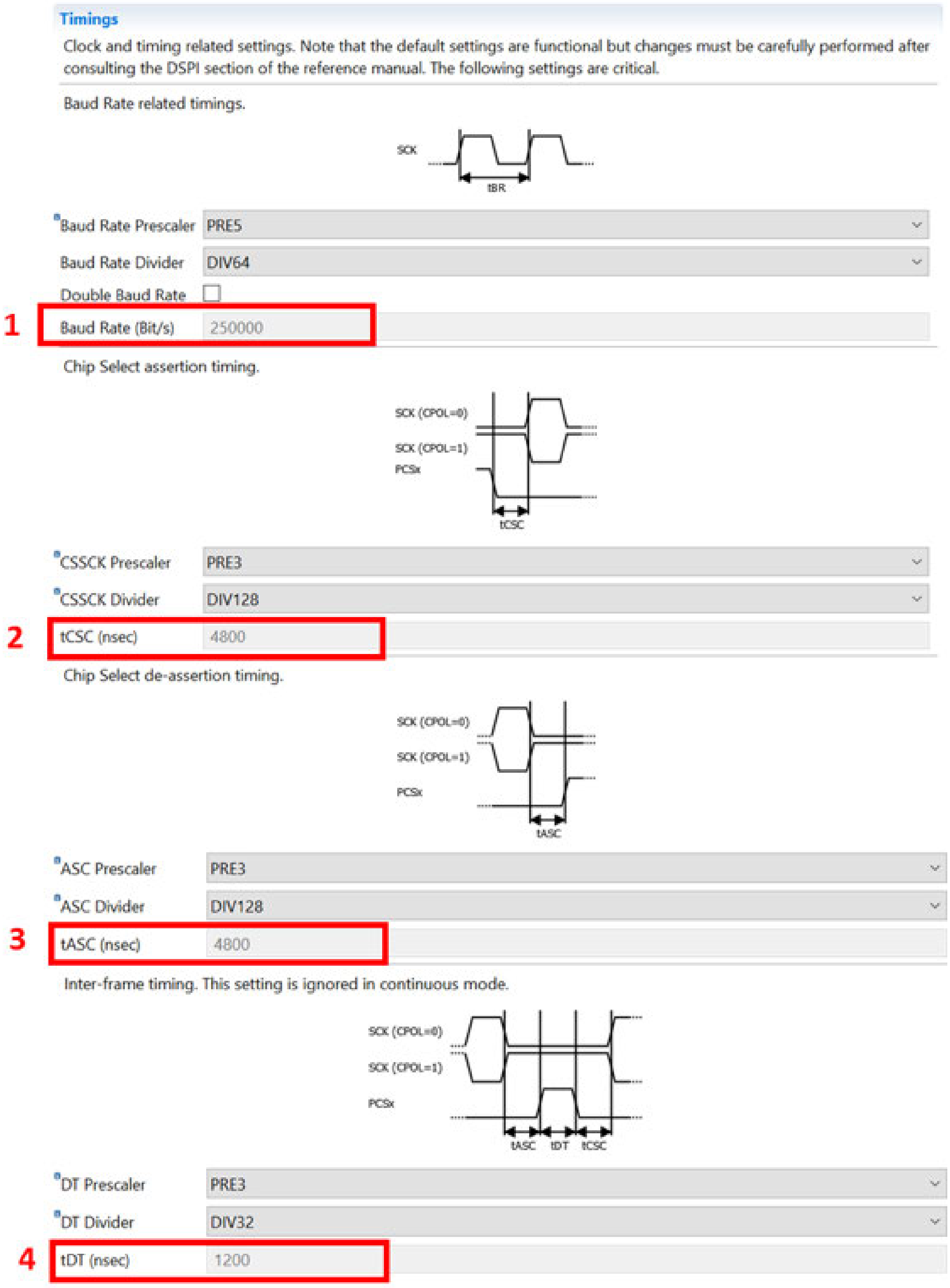
Frame Size: 8 bits

Frame Ordering: MSB first

Step 6. In the [**Timings**] section, configure the baud rate for the L99DZ200G device.

The values used in the demo are:

- baud rate (bit/s): 250000
- tCSC(nsec): 4800
- tASC(nsec): 4800
- tDT(nsec): 1200



Timings

Clock and timing related settings. Note that the default settings are functional but changes must be carefully performed after consulting the DSPI section of the reference manual. The following settings are critical.

Baud Rate related timings.

Baud Rate Prescaler	PRE5
Baud Rate Divider	DIV64
Double Baud Rate	□
Baud Rate (Bit/s)	250000

Chip Select assertion timing.

CSSCK Prescaler	PRE3
CSSCK Divider	DIV128
tCSC (nsec)	4800

Chip Select de-assertion timing.

ASC Prescaler	PRE3
ASC Divider	DIV128
tASC (nsec)	4800

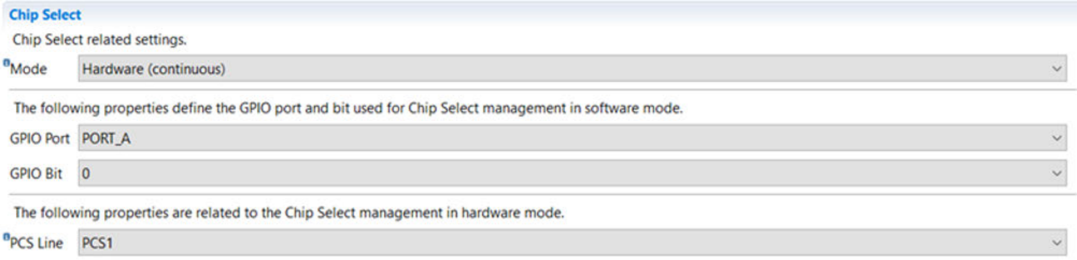
Inter-frame timing. This setting is ignored in continuous mode.

DT Prescaler	PRE3
DT Divider	DIV32
tDT (nsec)	1200

Figure 21. **SPI timing configuration**

Step 7. Fill the [**Chip Select**] section as shown below.

Figure 22. **SPI chip selection**



Chip Select
Chip Select related settings.
Mode: Hardware (continuous)
The following properties define the GPIO port and bit used for Chip Select management in software mode.
GPIO Port: PORT_A
GPIO Bit: 0
The following properties are related to the Chip Select management in hardware mode.
PCS Line: PCS1

3.4.2 Programmable interrupt timer (PIT) configuration

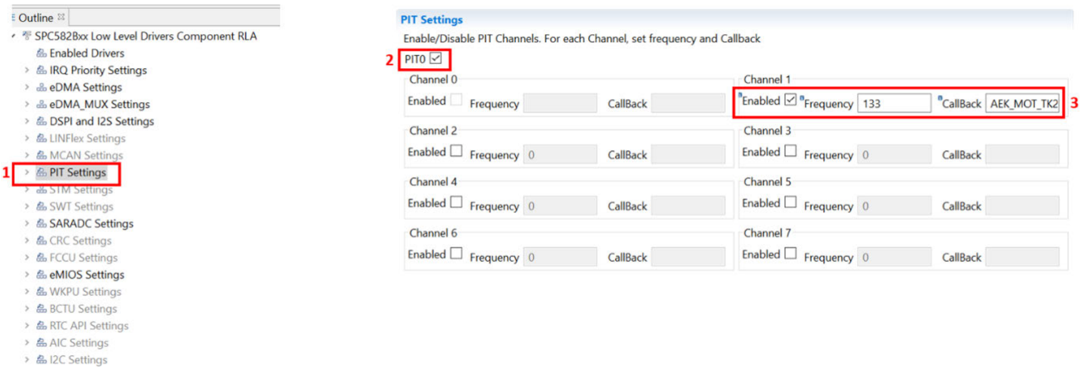
The PIT configuration is important to satisfy the correct triggering of the watchdog.

Step 1. Double-click on [**PIT Settings**] in the [**Outline**] tab.

Step 2. In the [**PIT settings**] section, tick [**PIT0**].

Step 3. In the [**Channel1**] box, enable the channel to insert the frequency according to the requirements of the L99DZ200G device and add the name of the callback function that is invoked as soon as the PIT expires. The name of the callback used in the demo we have developed is `AEK_MOT_TK200G1_TriggerWatchDog`.

Figure 23. **PIT configuration**



PIT Settings
Enable/Disable PIT Channels. For each Channel, set frequency and Callback
PIT0 ☑

Channel	Enabled	Frequency	CallBack
Channel 0	□		
Channel 1	☑	133	AEK_MOT_TK2
Channel 2	□	0	
Channel 3	□	0	
Channel 4	□	0	
Channel 5	□	0	
Channel 6	□	0	
Channel 7	□	0	

3.4.3 PWM configuration

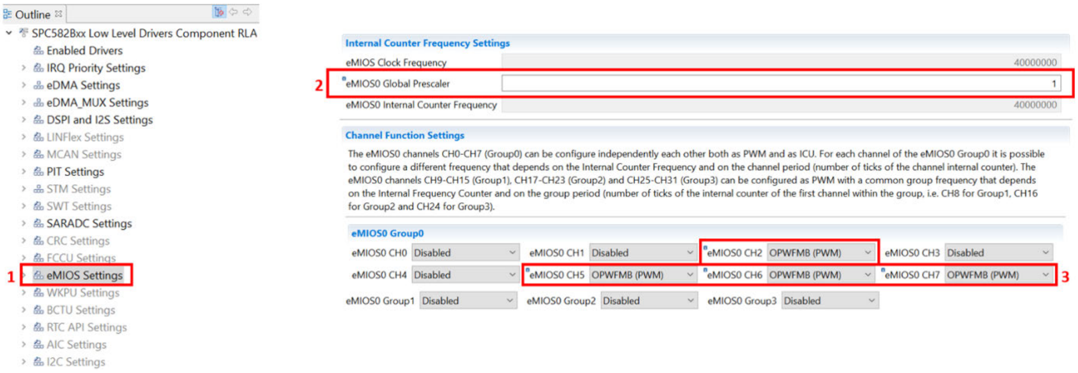
To drive the motors connected to the AEK-MOT-TK200G1 board with PWM signals, configure the eMIOS low-level driver.

Step 1. Double-click on [**eMios Settings**] in the [**Outline**] tab.

Step 2. Set the global prescaler in the [**Internal Counter Frequency Settings**] section.

Step 3. In the [**eMios Group0**], select channels 2, 5, 6, and 7 as PWM.

Figure 24. **PWM configuration (1 of 3)**



Internal Counter Frequency Settings

eMIOS Clock Frequency: 40000000

eMIOS0 Global Prescaler: 1

eMIOS0 Internal Counter Frequency: 40000000

Channel Function Settings

The eMIOS0 channels CH0-CH7 (Group0) can be configure independently each other both as PWM and as ICU. For each channel of the eMIOS0 Group0 it is possible to configure a different frequency that depends on the Internal Counter Frequency and on the channel period (number of ticks of the channel internal counter). The eMIOS0 channels CH9-CH15 (Group1), CH17-CH23 (Group2) and CH25-CH31 (Group3) can be configured as PWM with a common group frequency that depends on the Internal Frequency Counter and on the group period (number of ticks of the internal counter of the first channel within the group, i.e. CH8 for Group1, CH16 for Group2 and CH24 for Group3).

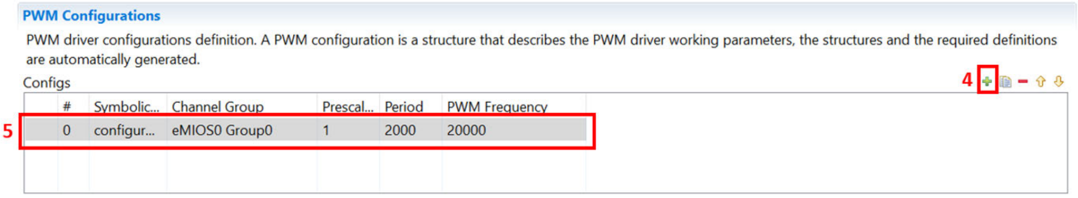
eMIOS0 Group0

eMIOS0 CH0: Disabled
eMIOS0 CH1: Disabled
eMIOS0 CH2: OPWFMB (PWM)
eMIOS0 CH3: Disabled
eMIOS0 CH4: Disabled
eMIOS0 CH5: OPWFMB (PWM)
eMIOS0 CH6: OPWFMB (PWM)
eMIOS0 CH7: OPWFMB (PWM)
eMIOS0 Group1: Disabled
eMIOS0 Group2: Disabled
eMIOS0 Group3: Disabled

Step 4. In the [**PWM Configurations**] section, click on the [**+**] button to add a row where to configure the PWM.

Step 5. Double-click on the row just added.

Figure 25. **PWM configuration (2 of 3)**



PWM Configurations

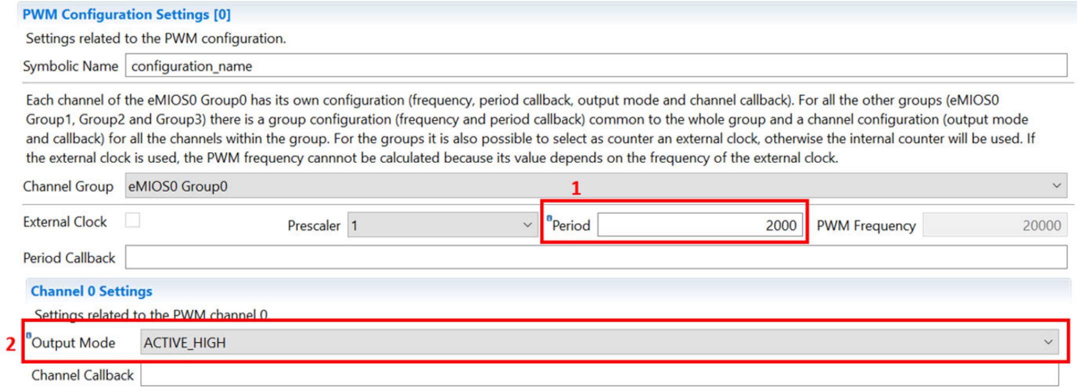
PWM driver configurations definition. A PWM configuration is a structure that describes the PWM driver working parameters, the structures and the required definitions are automatically generated.

Configs

#	Symbolic...	Channel Group	Prescal...	Period	PWM Frequency
0	configur...	eMIOS0 Group0	1	2000	20000

Step 6. In the [**PWM Configuration Settings [0]**] section, set the parameters as shown in the figure below.

Figure 26. **PWM configuration (3 of 3)**



PWM Configuration Settings [0]

Settings related to the PWM configuration.

Symbolic Name: configuration_name

Each channel of the eMIOS0 Group0 has its own configuration (frequency, period callback, output mode and channel callback). For all the other groups (eMIOS0 Group1, Group2 and Group3) there is a group configuration (frequency and period callback) common to the whole group and a channel configuration (output mode and callback) for all the channels within the group. For the groups it is also possible to select as counter an external clock, otherwise the internal counter will be used. If the external clock is used, the PWM frequency cannot be calculated because its value depends on the frequency of the external clock.

Channel Group: eMIOS0 Group0

External Clock: □ Prescaler: 1 Period: 2000 PWM Frequency: 20000

Period Callback:

Channel 0 Settings

Settings related to the PWM channel 0

Output Mode: ACTIVE_HIGH

Channel Callback:

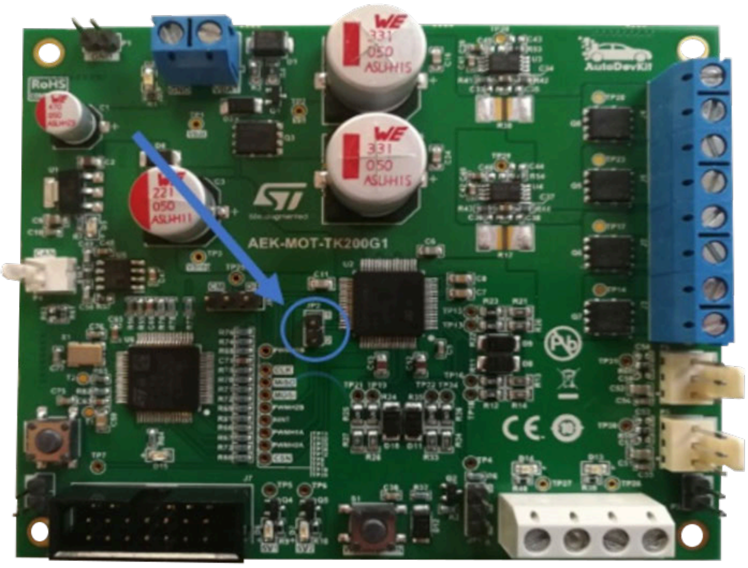
3.5 How to execute SW debug for the AEK-MOT-TK200G1

The software debug mode in the L99DZ200G is used for the micro controller code debugging. In this mode, all the L99DZ200G functionalities are available, except the watchdog requirement. Thus, the watchdog deactivation eases the debug of the microcontroller firmware.

To enter the debug mode, follow the procedure below.

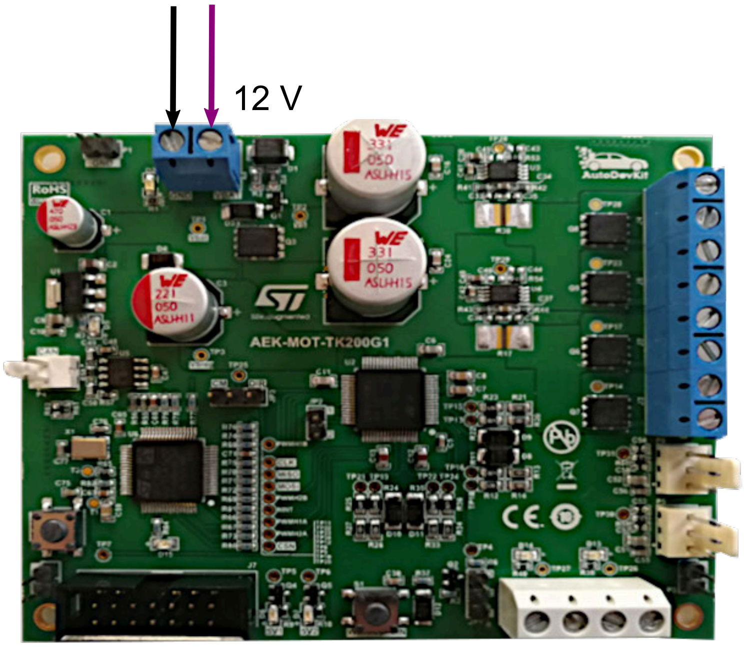
Step 1. Insert a jumper in the JP2 connector.

Figure 27. **Debug input mode: jumper positioning**



Step 2. Connect the power supply to the J1 connector and power the board.

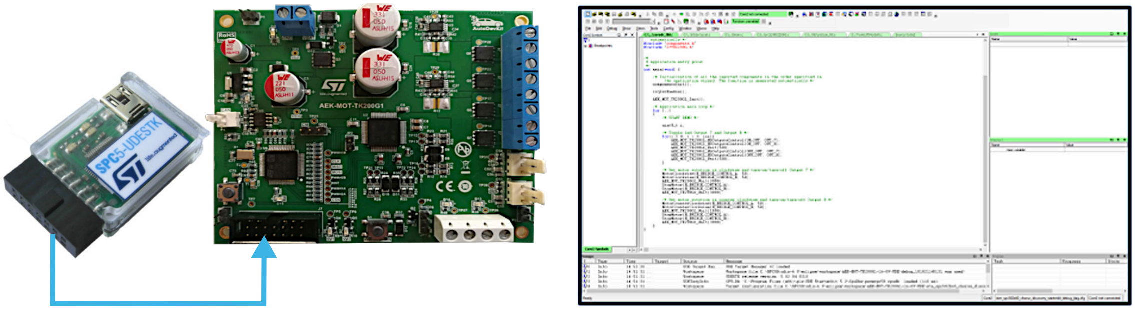
Figure 28. **Debug input mode: powering the board**



Step 3. Remove the jumper added in step 1.

Step 4. Connect the JTAG connector to the J7 connector and download the firmware with UDE PLS.

Figure 29. **Debug input mode: powering the board**



4 Available demos for the AEK-MOT-TK200G1

The following demos with specific features are provided for the AEK-MOT-TK200G1 board:

1. SPC582Bxx_RLA_AEK-MOT-TK200G1_MotorControl – Test Application for discovery
2. SPC582Bxx_RLA_AEK-MOT-TK200G1_MotorControl_via_CAN – Test Application
3. SPC58ECxx_RLA_MainECUFor_AEK-MOT-TK200G1Control – Test Application

4.1 SPC582Bxx_RLA_AEK-MOT-TK200G1_MotorControl – Test Application for discovery

The purpose of this demo is to show how to drive two DC motors and turn on/off two LED strings with the AEK-MOT-TK200G1 board. The code is characterized by an infinite loop where the actuations are repeated. At startup, the LED blinks to indicate that the application is starting. Then the two DC motors connected to the board are turned clockwise and counterclockwise.

The used APIs are:

- `AEK_MOT_TK200G1_Init()`: to initialize the board.
- `AEK_MOT_TK200G1_HSOutputControl()`: to configure led turn-on
- `AEK_MOT_TK200G1_Wait()`: to add a delay
- `MotorClockwise()`: to turn the motors clockwise
- `MotorCounterClockwise()`: to turn the motors counterclockwise
- `StopMotor()`: to stop the DC motors

4.2 SPC58ECxx_RLA_MainECUFor_AEK-MOT-TK200G1Control – Test Application

The purpose of this demo is to show how to drive the AEK-MOT-TK200G1 board through a domain controller (the AEK-MCU-C4MLIT1 board) via CAN messages, simulating a typical automotive system.

At startup, the two LED strings connected to the AEK-MOT-TK200G1 board blink to indicate that the application is starting. Then, the board remains on hold, waiting for the CAN messages to arrive from the domain/control zone. The demo supports the CAN messages that allow stopping and moving clockwise or counterclockwise the two connected DC motors. To run this demo, connect the AEK-MCU-C4MLIT1 board via CAN. Then, download the ”SPC58ECxx_RLA_MainECUFor_AEK-MOT-TK200G1Control – Test Application” demo on the board MCU.

The used APIs are:

- `AEK_MOT_TK200G1_Init()`: to initialize the board
- `AEK_MOT_TK200G1_HSOutputControl()`: to turn on the LED string
- `AEK_MOT_TK200G1_Wait()`: to add a delay
- `MotorClockwise()`: to turn the motors clockwise
- `MotorCounterClockwise()`: to turn the motors counterclockwise
- `StopMotor()`: to stop the DC motors
- `mcanconf_rxreceive()`: to parse the received CAN messages

The CAN commands supported by the demo are defined in the CANCommunication.h file (path: source/CANDriver/CANCommunication.h).

4.3 SPC58ECxx_RLA_MainECUFor_AEK-MOT-TK200G1Control – Test Application

The purpose of this demo is to show how the microcontroller SPC58ECxx hosted on the AEK-MCU-C4MLIT1 board is able to control the AEK-MOT-TK200G1 board via the CAN bus.

Through the SW2 and SW3 on-board buttons, two CAN messages are sent to move the DC motors clockwise and counterclockwise, respectively. In addition, the SW1 button sends a CAN message to stop the motors. Every time a CAN message is sent from the domain controller, an on-board LED blinks to highlight that the message has been correctly sent.

The used APIs are:

- `sendCanMessage()`: to send the message CAN
- `toggle_pad()`: to toggle the LED on the domain controller board

The CAN commands supported by the demo are defined in the CANCommunication.h file (path: source/CANDriver/CANCommunication.h).

5 Available APIs

Table 2. Available APIs for the AEK-MOT-TK200G1

API name	Description
`AEK_MOT_TK200G1_Init()`	Initializes the driver and clears the L99DZ200G status registers.
`AEK_MOT_TK200G1_CheckGlobalStatusByte()`	Checks the validity of the global status byte (GSB)
`AEK_MOT_TK200G1_ReadROMDeviceInfo()`	Reads the ROM registers and returns the information about the device
`AEK_MOT_TK200G1_GlobalStatusByte()`	Return the GSB
`AEK_MOT_TK200G1_GetWDCConfig()`	Gets the watchdog trigger time
`AEK_MOT_TK200G1_SetWDCTime()`	Sets the watchdog trigger time (10 ms, 50 ms, 100 ms, or 200 ms).
`AEK_MOT_TK200G1_SetModeControl()`	Sets the device in standby mode (Vbat standby or V1 standby).
`AEK_MOT_TK200G1_TriggerWatchDog()`	Callback PIT to toggle the watchdog
`AEK_MOT_TK200G1_SetPWMDuty()`	Sets the PWM duty cycle
`AEK_MOT_TK200G1_SetPWMFrequency()`	Sets the PWM frequency
`AEK_MOT_TK200G1_HSOutputsControl()`	Sets the high-side outputs and their configuration
`AEK_MOT_TK200G1_Autorecovery()`	Enables/disables the overcurrent auto-recovery for the outputs
`AEK_MOT_TK200G1_PWMOddDutyCycleSettings()`	Sets the PWM duty cycle for the odd output channels
`AEK_MOT_TK200G1_PWMEvenDutyCycleSettings()`	Sets the PWM duty cycle for the even PWM output channels
`AEK_MOT_TK200G1_EnableHB()`	Enables/disables the H-bridge A or H-bridge B
`AEK_MOT_TK200G1_getDeviceStatusInformations()`	Selects the clusters with dedicated thermal sensors to monitor the output temperature or voltage at the VSREG, VS, or WU pin
`AEK_MOT_TK200G1_V1Reset()`	Sets the voltage regulator V1 reset threshold (VRTH), monitored in active or V1_Standby state
`AEK_MOT_TK200G1_PWMFrequencySettings()`	Sets the PWM frequency for the channel
`AEK_MOT_TK200G1_OCROnTime()`	Sets the time for the overcurrent recovery (overcurrent filter time for high-side).
`AEK_MOT_TK200G1_OCRFrequency()`	Sets the frequency for the overcurrent recovery (recovery frequency for the OC)
`AEK_MOT_TK200G1_TimerConfig()`	Programmable timer interrupt. It sets period and on time.
`AEK_MOT_TK200G1_VoltReg2Control()`	Sets the voltage regulator V2 configuration
`AEK_MOT_TK200G1_ClearAllStatusRegisters()`	Clears all the status registers
`AEK_MOT_TK200G1_WU1Config()`	Sets the wake-up input pin as a wake-up source (disable) or an input voltage measurement (enable)
`AEK_MOT_TK200G1_DIR_CM_Control()`	Configuration register for the current monitor output or DIR input

API name	Description
AEK_MOT_TK200G1_OutCurrMonitoring()	Configuration register for the output current monitoring
AEK_MOT_TK200G1_OutOCThExp()	Enables/disables the thermal expiration feature that protects the device when continuous auto-recovery events are triggered
AEK_MOT_TK200G1_AutomaticDCCOmpensationOdd()	Sets the automatic Vs compensation for the odd outputs (VLed settings)
AEK_MOT_TK200G1_AutomaticDCCOmpensationEven()	Sets the automatic Vs compensation for the even outputs (VLed settings)
AEK_MOT_TK200G1_EnableOddAutomaticDCCompensation()	Enables the automatic Vs compensation for the odd outputs (VLed settings)
AEK_MOT_TK200G1_EnableEvenAutomaticDCCompensation()	Enables the automatic Vs compensation for the even outputs (VLed settings)
AEK_MOT_TK200G1_ClearErrorFlagsHBridgeA()	Clears bit 12-15 of the SR2 status register (drain-source monitoring for H-bridge A)
AEK_MOT_TK200G1_SlewRateCurrent()	Sets the slew rate current for the H-bridge
AEK_MOT_TK200G1_CrossCurrentProtectionTime()	Sets the cross-current protection time for the H-bridge A or H-bridge B
AEK_MOT_TK200G1_TestOLHxandLy()	Test open-load condition between H1 and L2 or H2 and L1 of the H-bridge A or H-bridge B
AEK_MOT_TK200G1_HBridgeOLHighThreshold()	Selects the H-bridge OL high threshold (5/6 Vs)
AEK_MOT_TK200G1_SlowDecayHS()	Sets the slow decay for leg 1 of the H-bridges
AEK_MOT_TK200G1_SlowDecaySingle()	Sets the slow decay single for leg 1 of the H-bridges
AEK_MOT_TK200G1_DSMonitorThreshold()	Drain-source monitoring threshold for the H-bridges
AEK_MOT_TK200G1_DualMotorMode()	Enables/disables the dual motor mode of the H-bridges
AEK_MOT_TK200G1_MotorDirection()	Sets the direction of the H-bridges
AEK_MOT_TK200G1_CheckWDExpired()	Checks whether the watchdog has expired
AEK_MOT_TK200G1_Wait()	Delays for the specified number of milliseconds
AEK_MOT_TK200G1_adcAEK_MOT_TK200G1()	ADC callback function
readADCLeft()	Filters the values coming from the left channel of the ADC
readADCRight()	Filters the values coming from the right channel of the ADC
MotorCounterClockwise()	Turns the motors counterclockwise
MotorClockwise()	Turns the motors clockwise
StopMotor()	Breaks the motor

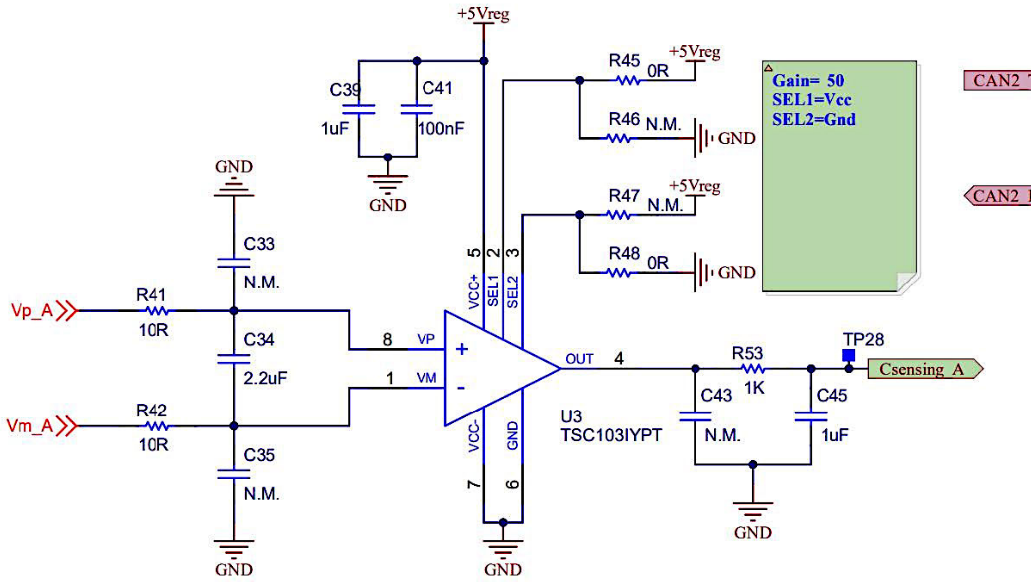
6 How to customize the current sensing network

The current sensing network implemented in the AEK-MOT-TK200G1 is designed to be used with the L12-50-100-12-P DC motor by Actuonix or a compatible one (see Section 1.1.3).

To customize the current detection network for other DC motors, consider:

- the minimum and maximum current value flowing in the Rsense resistor
- the ADC reference voltage (5 V in this case)

Figure 30. ADC reference voltage



With these parameters, you can compute the value of the Rsense resistor and the relative gain of the TSC103IYPT current sense amplifier.

The following steps show how the current sensing network has been dimensioned and how it can be modified according to the user's needs.

Step 1. From the motor datasheet, choose the stall current (250 mA in this case).

Step 2. Measure the motor nominal current with a multimeter (25 mA in this case).

Step 3. Check how the ADC working voltage has been configured in the SPC5-STUDIO. In this case, it is 5 V. This means that the voltage range that the ADC can read is 0 V-5 V.

Step 4. Calculate the maximum power dissipation of the Rsense.

In our case:

$$P_S = 0.025W$$

Step 5. Considering $P_S = R_{SENSE} \cdot (I_{\max})^2$ and knowing that the maximum current required by the DC motor at its maximum torque is $I_{\max} = 0.25A$, compute the Rsense value ($R_{SENSE} = 400mOhm$).

Step 6. Compute the gain of the current amplifier.

The gain value must amplify the Vrsense input signal (the voltage across the Rsense resistor) so that the Vout maximum and minimum output voltages (output voltage of the opamp) is measurable via the ADC.

The following formulas were used to calculate the opamp gain when Imax = 250 mA.

$$V_{RSENSE} = R_{SENSE} \cdot I_{MAX} = 0.1V$$
$$V_{OUT} = GAIN \cdot V_{RSENSE}$$

According to the previous formula, and with the constraint that Vout must not exceed 5 V, the opamp gain is set to 50. This value has been selected between four possible gain values (25, 50, 75, and 100) provided by the TSC103IYPT. The gain selection is implemented with the resistors connected to the SEL1 and SEL2 pins.

$$V_{OUT} = 50 \cdot V_{RSENSE} = 50 \cdot 0.1 = 5V$$

Step 7. Repeating the same approach used above, check that the gain = 50 is a good choice even when the current flowing through the Rsense resistor is the mimimum possible (that is, Imin = 25 mA).

$$V_{RSENSE} = R_{SENSE} \cdot I_{MIN} = 0.01V$$
$$V_{OUT} = GAIN \cdot V_{RSENSE}$$
$$V_{OUT} = 50 \cdot V_{RSENSE} = 50 \cdot 0.01 = 0.5V$$

Note: The layout of the board can accommodate Rsense resistors with footprints compatible with 0603 to 2512 packages.

7 Motor model used during the AEK-MOT-TK200G1 emission tests

The motors used during the tests are 413-0622 RS PRO.

Table 3. **Motor specifications**

Specification	Value
Voltage option	12 VDC
Maximum input voltage	12 V
Current rating	190 mA
Maximum speed	66 rpm/min
Maximum momentary tolerance torque	3.0 kgf-cm
Maximum speed (no load)	82 rpm/min
Reduction ratio	1/100

8 Schematic diagrams

Figure 31. AEK-MOT-TK200G1 circuit schematic (1 of 5)

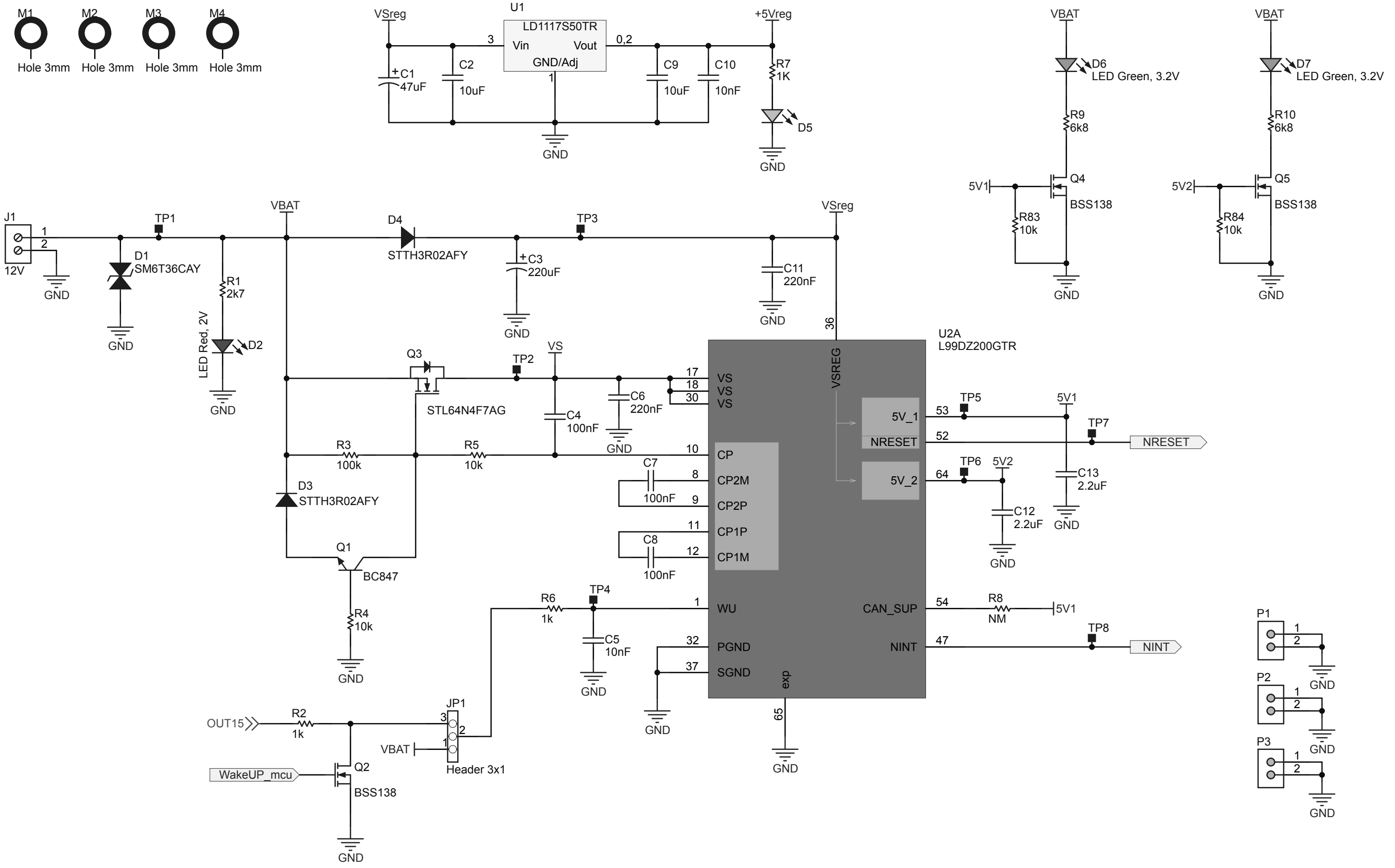


Figure 32. AEK-MOT-TK200G1 circuit schematic (2 of 5)

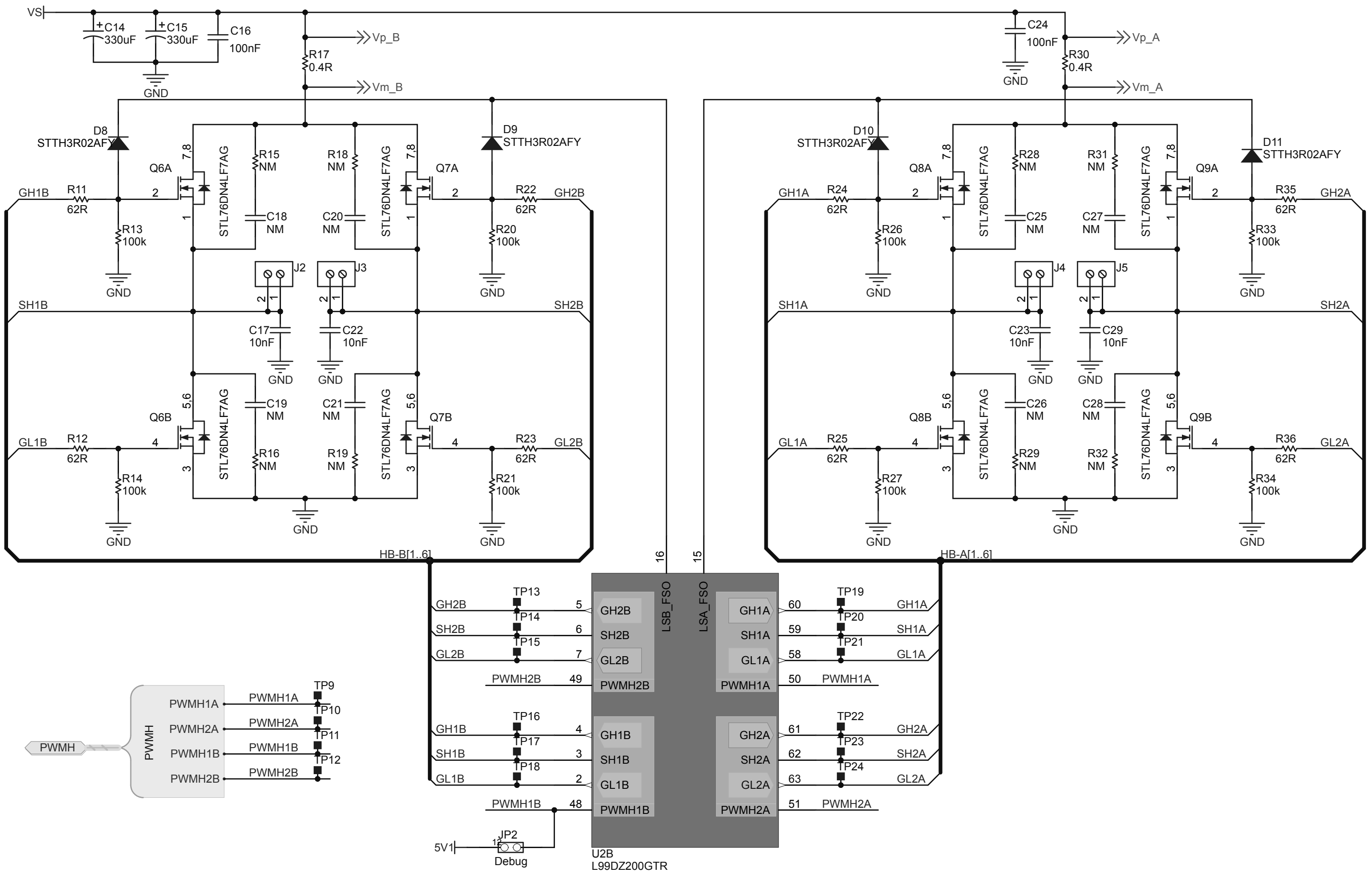


Figure 33. AEK-MOT-TK200G1 circuit schematic (3 of 5)

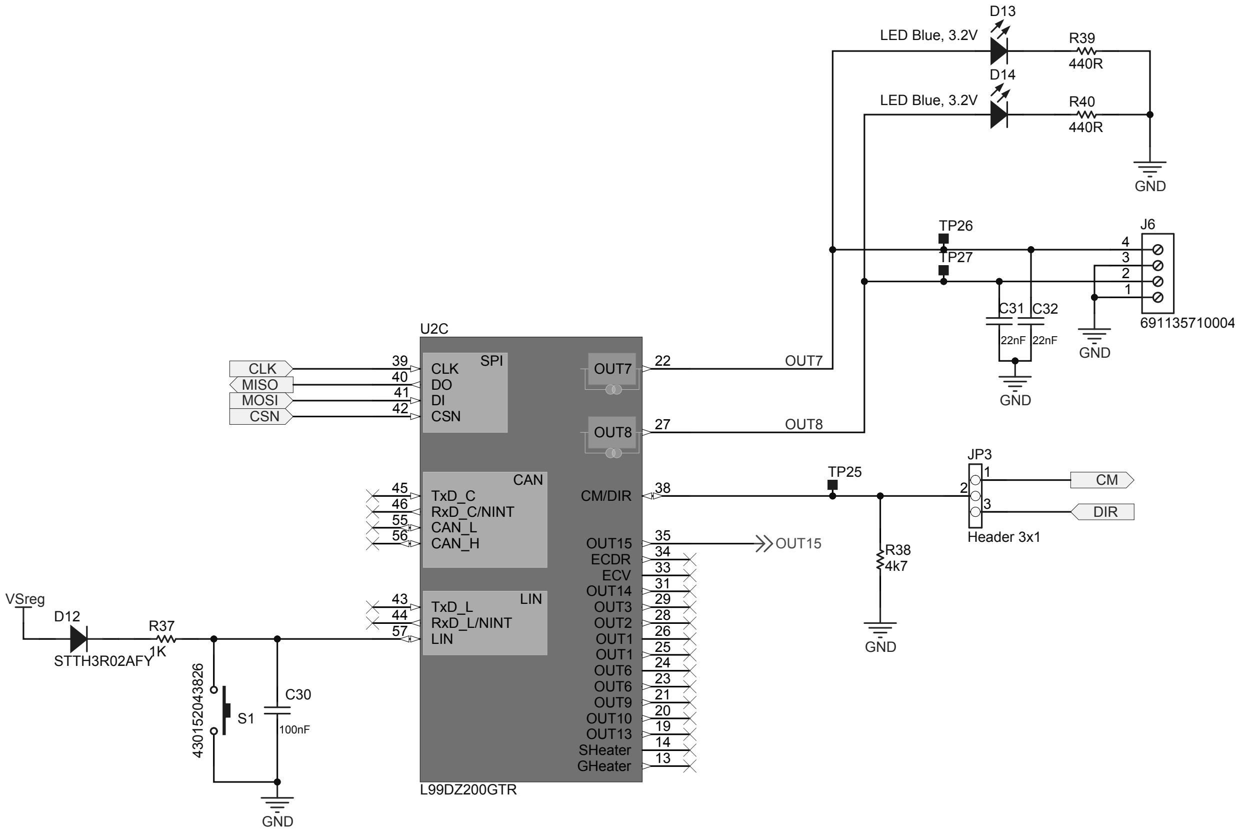
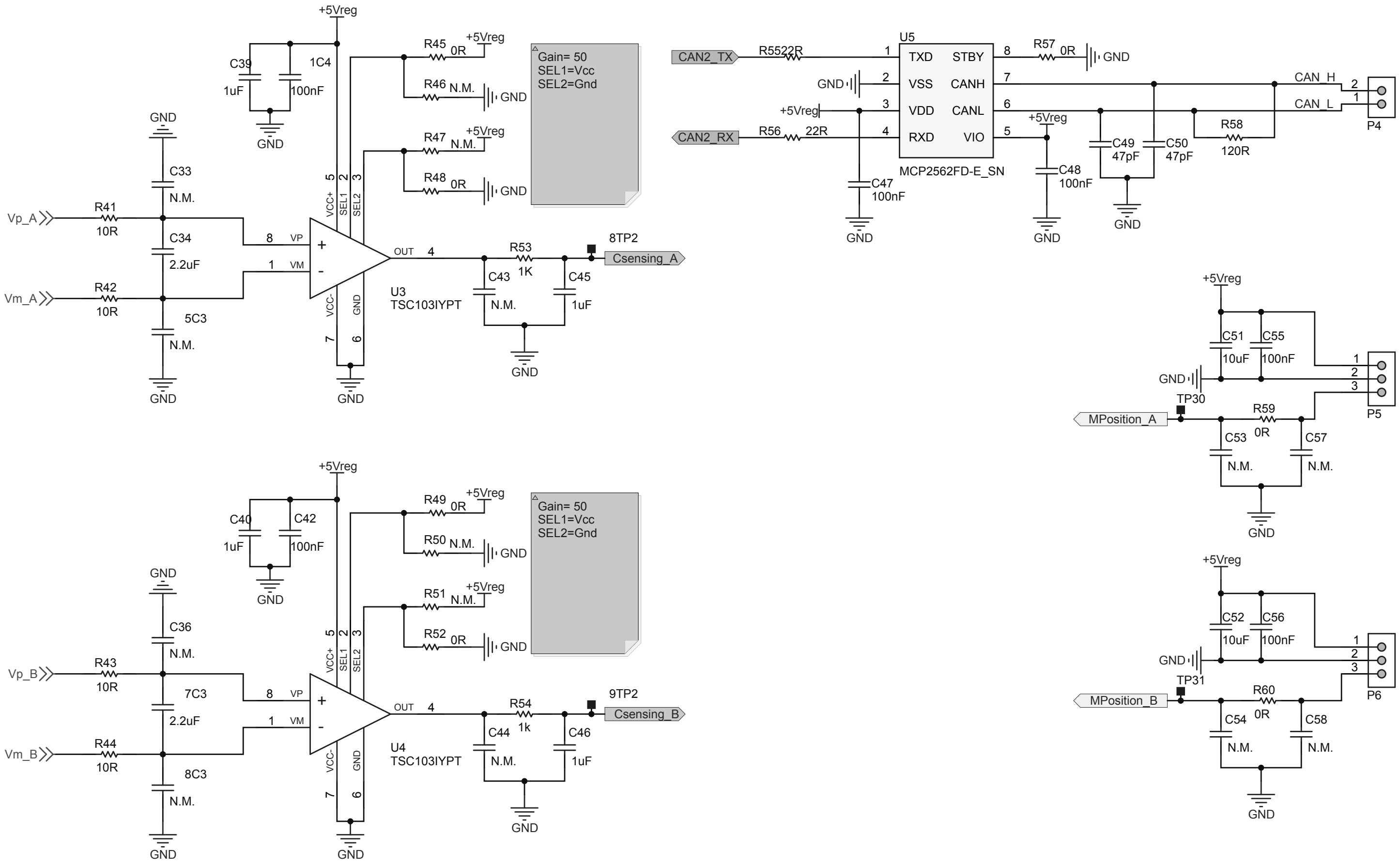


Figure 34. **AEK-MOT-TK200G1 circuit schematic (4 of 5)**



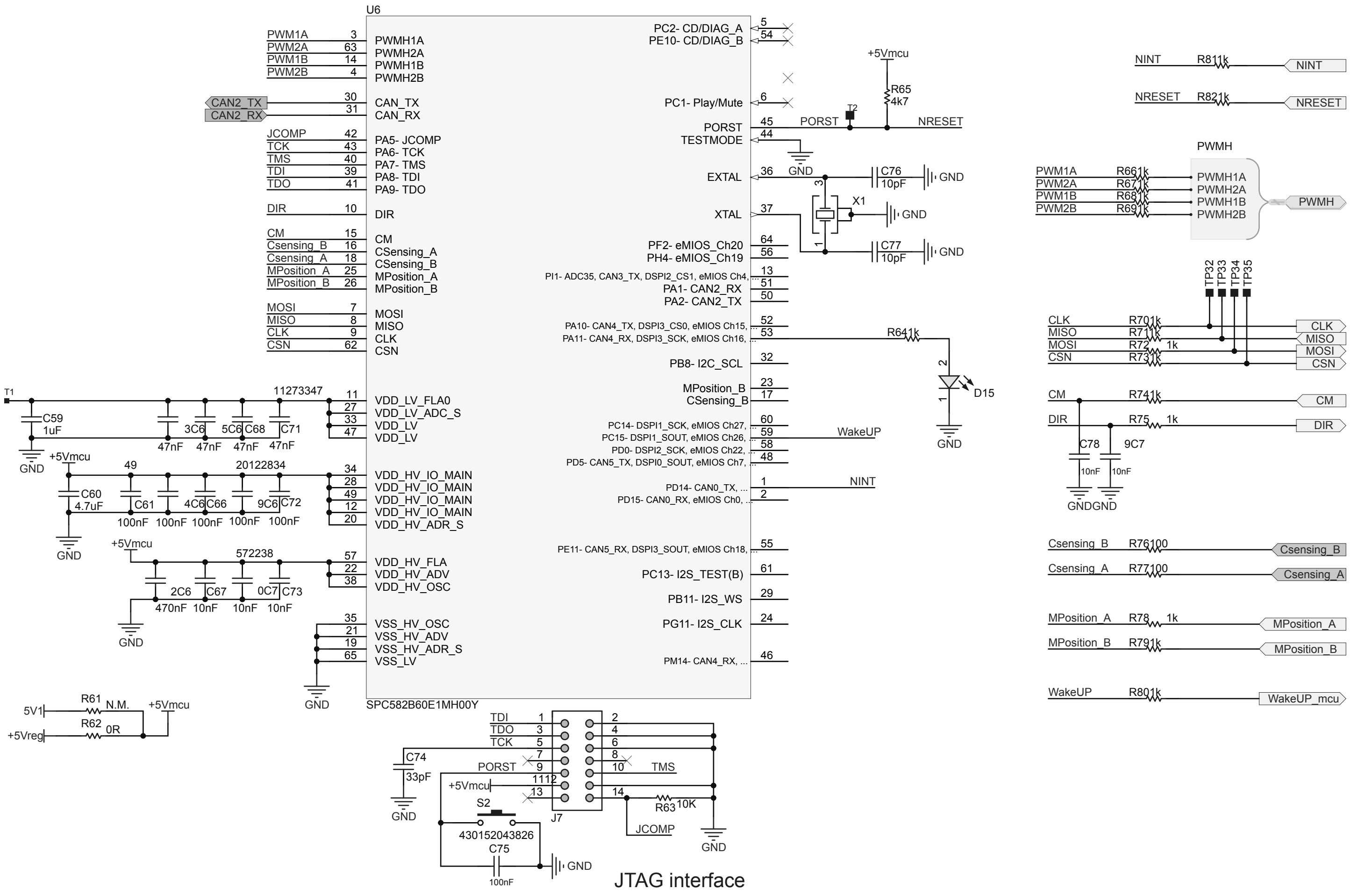


Figure 35. AEK-MOT-TK200G1 circuit schematic (5 of 5)

9 Bill of materials

Table 4. AEK-MOT-TK200G1 bill of materials

Item	Q.ty	Ref.	Value	Description	Manufacturer	Part Number
1	1	C1	47uF, WCAP-ASLI_D6.3H7.7, 50 V, ±20 %	SMD Electrolytic Capacitor	Wurth Elektronik	865080645012
2	4	C2, C9, C51, C52	10uF, 0805C, 25 V, ±20 %	SMD Ceramic Capacitor	TAIYO YUDEN	TMK212BC6106MG-T
3	1	C3	220uF, WCAP-ASLI_10x10.5, 50 V, ±20 %	SMD Electrolytic Capacitor	Wurth Elektronik	865080657018
4	5	C4, C7, C8, C16, C24	100nF, 0805C, 100 V, ±10 %	SMD Ceramic Capacitor	Wurth Elektronik	885012207128
5	7	C5, C10, C67, C70, C73, C78, C79	10nF, 0603C, 25 V, ±10 %	SMD Ceramic Capacitor	TDK	CGA3E2X7R2A103K080AA
6	2	C6, C11	220nF, 0805C, 50 V, ±10 %	SMD Ceramic Capacitor	Wurth Elektronik	885382207008
7	2	C12, C13	2.2uF, 0805C, 25 V, ±10 %	SMD Ceramic Capacitor	Wurth Elektronik	885012207079
8	2	C14, C15	330uF, WCAP-ASLI_D12.5H14, 50 V, ±20 %	SMD Electrolytic Capacitor	Wurth Elektronik	865080662019
9	4	C17, C22, C23, C29	10nF, 0805C, 50 V, ±10 %	SMD Ceramic Capacitor	Wurth Elektronik	885012207092
10	8	C18, C19, C20, C21, C25, C26, C27, C28	NM, 0805C	SMD Ceramic Capacitor (not mounted)	-	-
11	13	C30, C41, C42, C47, C48, C55, C56, C61, C64, C66, C69, C72, C75	100nF, 0603C, 50 V, ±10 %,	SMD Ceramic Capacitor	Wurth Elektronik	885012206095
12	2	C31, C32	22nF, 0805C, 50 V, ±10 %	SMD Ceramic Capacitor	Wurth Elektronik	885012207094

Item	Q.ty	Ref.	Value	Description	Manufacturer	Part Number
13	10	C33, C35, C36, C38, C43, C44, C53, C54, C57, C58	N.M., 0603C	SMD Ceramic Capacitor (not mounted)	-	-
14	2	C34, C37	2.2uF, 0603C, 25 V, ±10 %	SMD Ceramic Capacitor	MURATA	GRM188C71E225KE11D
15	4	C39, C40, C45, C46	1uF, 0603C, 25 V, ±10 %	SMD Ceramic Capacitor	Wurth Elektronik	885012206076
16	2	C49, C50	47pF, 0603C, 50 V, ±5 %	SMD Ceramic Capacitor	Wurth Elektronik	885012006055
17	1	C59	1uF, 0805C, 50 V, ±10 %	SMD Ceramic Capacitor	Wurth Elektronik	885012207103
18	1	C60	4.7uF, WCAP-CSGP_1210_H=2.5mm, 50 V, ±10 %	SMD Electrolytic Capacitor	Wurth Elektronik	885012209048
19	1	C62	470nF, 0805C, 50 V, ±10 %	SMD Ceramic Capacitor	Wurth Elektronik	885012207102
20	4	C63, C65, C68, C71	47nF, 0603C, 50 V, ±10 %	SMD Ceramic Capacitor	Wurth Elektronik	885012206093
21	1	C74	33pF, 0603C, 50 V, ±5 %	SMD Ceramic Capacitor	Wurth Elektronik	885012006054
22	2	C76, C77	10pF, 0603C, 50 V, ±5 %	SMD Ceramic Capacitor	Wurth Elektronik	885012006051
23	1	D1	SM6T36CAY, SMB C2	Automotive 600 W, 30.8 V TVS in SMB	ST	SM6T36CAY
24	1	D2	LED Red, 2V, WL-SMCW_0805,	WL-SMCW SMT Mono-color Chip LED Waterclear, size 0805, Red, 2V, 140deg_150080RS75000	Wurth Elektronik	150080RS75000
25	7	D3, D4, D8, D9, D10, D11, D12	STTH3R02AFY, SOD128 Flat	Automotive 200 V, 3 A ultrafast diode	ST	STTH3R02AFY
26	1	D5	LED Amber, 2V, WL-SMCW_0805	WL-SMCW SMT Mono-color Chip LED Waterclear, size 0805, Amber, 2V, 140deg_150080AS75000	Wurth Elektronik	150080AS75000
27	2	D6, D7	LED Green, 3.2V, WL-SMCW_0805	WL-SMCW SMT Mono-color Chip LED Waterclear, size 0805, Green, 3.2V, 140deg_150080GS75000	Wurth Elektronik	150080GS75000
28	2	D13, D14	LED Blue, 3.2V, WL-SMCW_0805	WL-SMCW SMT Mono-color Chip LED Waterclear, size 0805, Blue, 3.2V, 140deg_150080BS75000	Wurth Elektronik	150080BS75000

Item	Q.ty	Ref.	Value	Description	Manufacturer	Part Number
29	1	D15	150080YS75000, WL-SMCW_0805	WL-SMCW SMT Mono-color Chip LED Waterclear, size 0805, Yellow, 2V, 140deg	Wurth Elektronik	150080YS75000
30	1	J1	12V, 691213510002	WR-TBL Serie 2135 Horizontal Entry Modular, Rising Cage Clamp, pitch 5.08mm, 2p	Wurth Elektronik	691213510002
31	4	J2, J3, J4, J5	691102710002, 691102710002	WR-TBL Serie 102 Horizontal Entry Modular, Pressure Clamp, pitch 5mm, 2p	Wurth Elektronik	691102710002
32	1	J6	691135710004, 691135710004	WR-TBL Serie 1357 Horizontal Cable Entry, Pressure Clamp, THT, pitch 5mm, 4p	Wurth Elektronik	691135710004
33	1	J7	61201421621, WR-BHD 2.54 mm Male Box Header	Male Box Header WR-BHD, THT, Vertical, pitch 2.54 mm, 14 pins	Wurth Elektronik	61201421621
34	2	JP1, JP3	Header 3x1, WR-PHD 3pin 2.54 mm THT Pin Header	Header, 3-Pin, Single row	Wurth Elektronik	61300311121
35	1	JP2	Header 2x1, 61300211121	Header, 2-Pin, Single row	Wurth Elektronik	61300211121
36	4	M1, M2, M3, M4	Hole 3mm, M3	Mountig hole, D=3mm	Wurth Elektronik	970100365
37	3	P1, P2, P3	61300211121	WR-PHD Pin Header, THT, pitch 2.54mm, Single Row, Vertical, 2p	Wurth Elektronik	61300211121
38	1	P4	61900211121	WR-WTB THT Male Vertical Locking Header, pitch 2.54mm, 2p	Wurth Elektronik	61900211121
39	2	P5, P6	61900311121	WR-WTB THT Male Vertical Locking Header, pitch 2.54mm, 3p	Wurth Elektronik	61900311121
40	1	Q1	BC847, SOT23	NPN Bipolar Transistor	Nexperia	BC847
41	3	Q2, Q4, Q5	BSS138, SOT23	N-Channel Enhancement Mode Vertical DMOS FET	Onsemi	BSS138
42	1	Q3	STL64N4F7AG,PowerFLAT 5x6 WF/STL260N4F7, PowerFLAT 5x6	Automotive-grade N-channel 40 V, 7.0 mOhm typ., 64 A STripFET F7 power MOSFET in a PowerFLAT 5x6 package/N-channel 40 V, 1.05 mOhm typ., 120 A STripFET F7 power MOSFET in a PowerFLAT 5x6 package	ST	STL64N4F7AG, or STL260N4F7
43	4	Q6, Q7, Q8, Q9	STL76DN4LF7AG, PowerFLAT 5x6 double island WF	Automotive-grade dual N-channel 40 V, 5 mOhm typ., 40 A STripFET F7 power MOSFET in PowerFLAT 5x6 double island package	ST	STL76DN4LF7AG
44	1	R1	2k7, 0603R, 1/8 W, ±1 %	SMD Resistor	VISHAY	MCT06030C2701FP500

Item	Q.ty	Ref.	Value	Description	Manufacturer	Part Number
45	22	R2, R6, R7, R37, R53, R54, R64, R66, R67, R68, R69, R70, R71, R72, R73, R74, R75, R78, R79, R80, R81, R82	1k, 0603R, 1/8 W, ±1 %	SMD Resistor	PANASONIC	ERJH3EF1001V
46	9	R3, R13, R14, R20, R21, R26, R27, R33, R34	100k, 0805R, 1/4 W, ±5 %	SMD Resistor	PANASONIC	ERJT06J104V
47	2	R4, R5	10k, 0805R, 1/4 W, ±5 %	SMD Resistor	PANASONIC	ERJT06J103V
48	1	R8	NM, 0603R	SMD Resistor (not mounted)	-	-
49	2	R9, R10	6k8, 0603R, 1/8 W, ±1 %	SMD Resistor	PANASONIC	ERJH3EF6801V
50	8	R11, R12, R22, R23, R24, R25, R35, R36	62R, 0805R, 1/4 W, ±5 %	SMD Resistor	PANASONIC	ERJT06J620V
51	8	R15, R16, R18, R19, R28, R29, R31, R32	NM, 0805R	SMD Resistor	-	-
52	2	R17, R30	0.4R, 2512R, 1 W, ±1 %	Thick film resistor	Yageo	PT2010FK-7W0R4L
53	2	R38, R65	4k7, 0603R, 1/8 W, ±1 %	SMD Resistor	PANASONIC	ERJH3EF4701V
54	2	R39, R40	442R, 0603R, 1/8 W, ±1 %	SMD Resistor	PANASONIC	ERJH3EF4420V

Item	Q.ty	Ref.	Value	Description	Manufacturer	Part Number
55	4	R41, R42, R43, R44	10R, 0603R, 1/8 W, ±1 %	SMD Resistor	PANASONIC	ERJH3EF10R0V
56	8	R45, R48, R49, R52, R57, R59, R60, R62	0R, 0603R, 1/8 W, ±1 %	SMD Resistor	PANASONIC	ERJH3G0R00V
57	5	R46, R47, R50, R51, R61	N.M., 0603R	SMD Resistor	-	-
58	2	R55, R56	22R, 0603R, 1/8 W, ±1 %	SMD Resistor	PANASONIC	ERJH3EF22R0V
59	1	R58	120R, 0603R, 1/8 W, ±1 %	SMD Resistor	PANASONIC	ERJH3EF1200V
60	3	R63, R83, R84	10K, 0603R, 1/8 W, ±1 %	SMD Resistor	PANASONIC	ERJH3EF1002V
61	2	R76, R77	100, 0603R, 1/8 W, ±1 %	SMD Resistor	PANASONIC	ERJH3EF1000V
62	2	S1, S2	430152043826	Switch	Wurth Elektronik	430152043826
63	2	T1, T2	TEST POINT, TEST POINT_1.27MM_SMD	Simple pin (for 2,54mm pitch array)	Any	Any
64	1	U1	LD1117S50TR, SOT-223	Adjustable and fixed low drop positive voltage regulator	ST	LD1117S50TR
65	1	U2	L99DZ200GTR, LQFP 64 10x10x1.4	Automotive front door device with LIN and HS-CAN providing dual H-bridge driving	ST	L99DZ200GTR
66	2	U3, U4	TSC103IYPT, TSSOP-8L	High-voltage, high-side current sense amplifier	ST	TSC103IYPT
67	1	U5	MCP2562FD-E_SN, SO-8,	CAN BUS TRANSCEIVER	Microchip	MCP2562FD-E/SN
68	1	U6	SPC582B60E1MH00Y, TQFP 64 10x10x1.0	32-bit power architecture MCU for automotive general purpose applications - Chorus family	ST	SPC582B60E1MH00Y
69	1	X1	830059537, WE-XTAL_CFPX-104	WE-XTAL Quartz Crystal, SMT, CFPX-104, 40MHz, +/-20ppm	Wurth Elektronik	830059537
70	1	-	60900213421	Jumper 2.54mm	Wurth Elektronik	60900213421
71	4	-	970100365	Nylon spacer M3x10mmF/F	Wurth Elektronik	970100365
72	4	-	97790603111	Nylon screw M3x6mm	Wurth Elektronik	97790603111
73	1	-	61900211621	Receptacle housing	Wurth Elektronik	61900211621

Item	Q.ty	Ref.	Value	Description	Manufacturer	Part Number
74	2	-	61900113722DEC	WR-WTB 2.54 mm Female Crimp Contact	Wurth Elektronik	61900113722DEC
75	1	-	60900213421	WR-PHD 2.54mm Jumper with Test Point	Wurth Elektronik	60900213421

10 Board versions

Table 5. AEK-MOT-TK200G1 versions

PCB version	Schematic diagrams	Bill of materials
AEK$MOT-TK200G1A [1]	AEK$MOT-TK200G1A schematic diagrams	AEK$MOT-TK200G1A bill of materials

1. *This code identifies the AEK-MOT-TK200G1 evaluation board first version. It is printed on the board PCB.*

11 Regulatory compliance information

Formal Notice Required by the U.S. Federal Communications Commission

FCC NOTICE

This kit is designed to allow:

(1) Product developers to evaluate electronic components, circuitry, or software associated with the kit to determine

whether to incorporate such items in a finished product and

(2) Software developers to write software applications for use with the end product.

This kit is not a finished product and when assembled may not be resold or otherwise marketed unless all required FCC equipment authorizations are first obtained. Operation is subject to the condition that this product not cause harmful interference to licensed radio stations and that this product accept harmful interference. Unless the assembled kit is designed to operate under part 15, part 18 or part 95 of this chapter, the operator of the kit must operate under the authority of an FCC license holder or must secure an experimental authorization under part 5 of this chapter 3.1.2.

Formal Product Notice Required by Industry Canada Innovation, Science and Economic Development

Canada compliance:

For evaluation purposes only. This kit generates, uses, and can radiate radio frequency energy and has not been tested for compliance with the limits of computing devices pursuant to Industry Canada (IC) rules.

À des fins d'évaluation uniquement. Ce kit génère, utilise et peut émettre de l'énergie radiofréquence et n'a pas été testé pour sa conformité aux limites des appareils informatiques conformément aux règles d'Industrie Canada (IC).

Formal product notice required by EU

This device is in conformity with the essential requirements of the Directive 2014/30/EU (EMC) and of the Directive 2015/863/EU (RoHS).

Revision history

Table 6. **Document revision history**

Date	Revision	Changes
17-May-2022	1	Initial release.

Contents

List of tables

List of figures