AEKD-BLINDSPOTB1

Data brief

Set of assembled boards for blind-spot detection simulation



Features

- Full set of evaluation boards to drive vehicle blind-spot application development:
  - AEK-MCU-C4MLIT1: main control board with SPC58EC80E5 automotive microcontroller running firmware to interpret detected magnetic field variations
  - AEK-LED-21DISM1: advanced automotive-grade LED driver board used to activate the vehicle in blind-spot warning LED
  - EV-VN7050AS: high-side driver used to drive the DC gearbox motor of the conveyor belt
- Unique power supply path for all boards and the hardware assembly
- 5 V, 2 A AC-DC included
- Part of the AutoDevKit™ initiative
- RoHS and WEEE compliant

Description

The AEKD-BLINDSPOTB1 set of assembled evaluation boards is used with the AEKD-BLINDSPOTA1 car model demonstration assembly to simulate vehicle detection in automotive blind-spot regions. It achieves this by interpreting magnetic field variations detected by a Hall sensor mounted near a stationary model car, caused by magnets mounted on another model car approaching on a conveyor belt driven by a 12 V_{DC} gearbox motor with 40 RPM and high torque.

The goal of the training kit is to support firmware development of blind-spot applications using the SPC5-Studio integrated development environment with AutoDevKit plugin extension, which includes the source code.

The system can be tested using the STSW-BLINDSPOT firmware downloadable from www.st.com.

Product summary	
Blind-spot board panel	AEKD-BLINDSPOTB1
Firmware for AEKD-BLINDSPOTB1	STSW-BLINDSPOT
Development environment	SPC5-Studio
community.st.com/autodevkit	AutoDevKit
AutoDevKit library plugin for SPC5-STUDIO	STSW-AUTODEVKIT
SPC5-UDESTK Debugging Software for Windows	SPC5-UDESTK-SW
MCU discovery board for SPC5 Chorus 4M automotive microcontroller with CAN transceivers	AEK-MCU-C4MLIT1
Digitally controlled LED driver board for automotive lighting applications	AEK-LED-21DISM1
VN7050AS evaluation board based on VIPower M0-7 technology	EV-VN7050AS
Applications	Chassis and Safety

1 Overview

Blind-spot detection is implemented in modern transportation vehicles to prevent accidents in common overtaking scenarios and other dangerous circumstances involving vehicles occupying areas that cannot seen directly by the drivers of other vehicles. These warning systems detect the presence of vehicles in blind-spot zones and alert the driver.

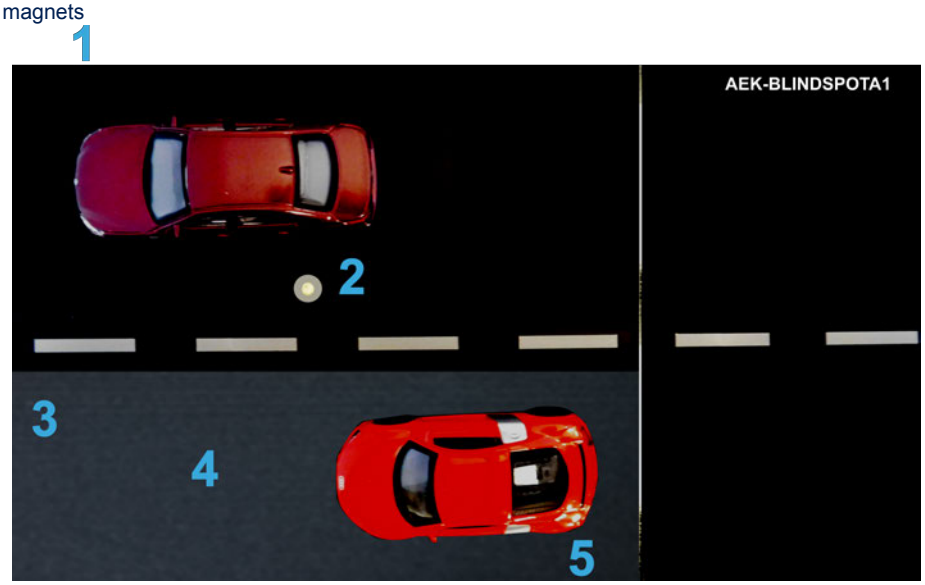
In this simulation kit, the system consists of a plexiglass structure to simulate the loads and a set of boards to drive the application (AEKD-BLINDSPOTB1). Two cars are placed on the demo: one is stationary, while another car with embedded magnets moves on the conveyor belt driven by a gear-DC motor. A Hall sensor under the belt measures the magnetic field around it and a LED near the stationary car indicates when the blind-spot warning signal is triggered.

The application is controlled by:

- a 32-bit automotive microcontroller (SPC58EC80E5)
- a VIPower actuator board (EV-VN7050AS) with a high side driver used to drive the conveyor belt geared DC motor
- a LED driver board (AEK-LED-21DISM1) drives the warning LED

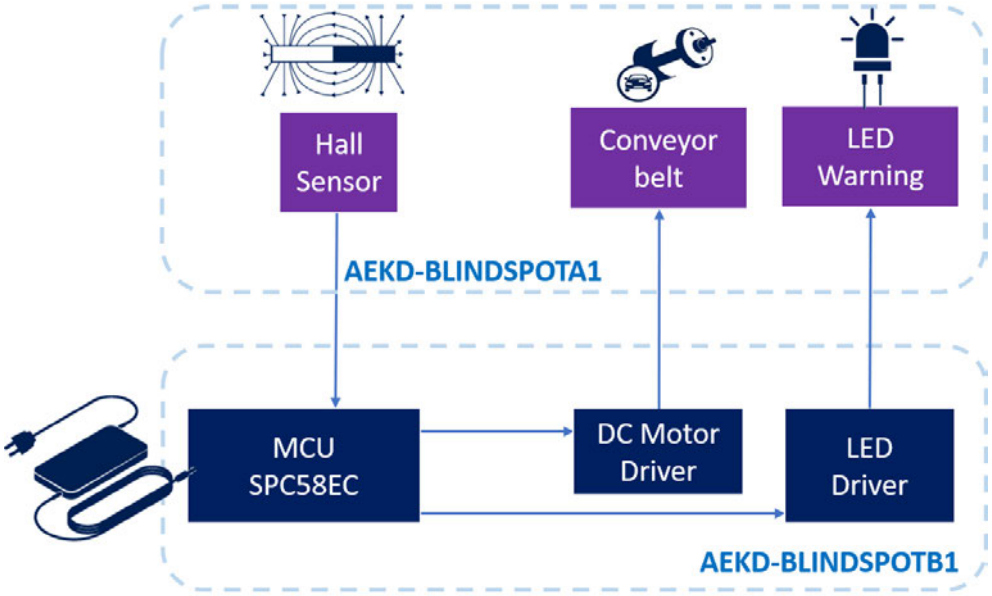
Figure 1. AEKD-BLINDSPOTA1 hardware assembly

1. Side for board connection
2. LED
3. DC motor
4. Hall sensor
5. Car with magnets



1.1 Block diagram

Figure 2. **AEKD-BLINDSPOTA1 plus AEKD-BLINDSPOTB1 blind-spot detection kit component block diagram**



1.2 How to create a blind-spot detection application using AutoDevKit

Step 1. Create a new project and select the MCU platform to use, i.e. SPC58 Chorus 4 Mb

Step 2. Add SPC5 Init Component for clock configurator, interrupt management and an operating system abstraction layer together with bootstrap files

Step 3. Add AutoDevKit Init Component for Board View Editor

Step 4. Add low-level drivers for MCU peripherals management

Step 5. Pick AutoDevKit components according to the blind-spot warning system to build: LED driver, DC motor driver for the conveyor belt and the Hall sensor capture component

Step 6. Configure the added AutoDevKit components

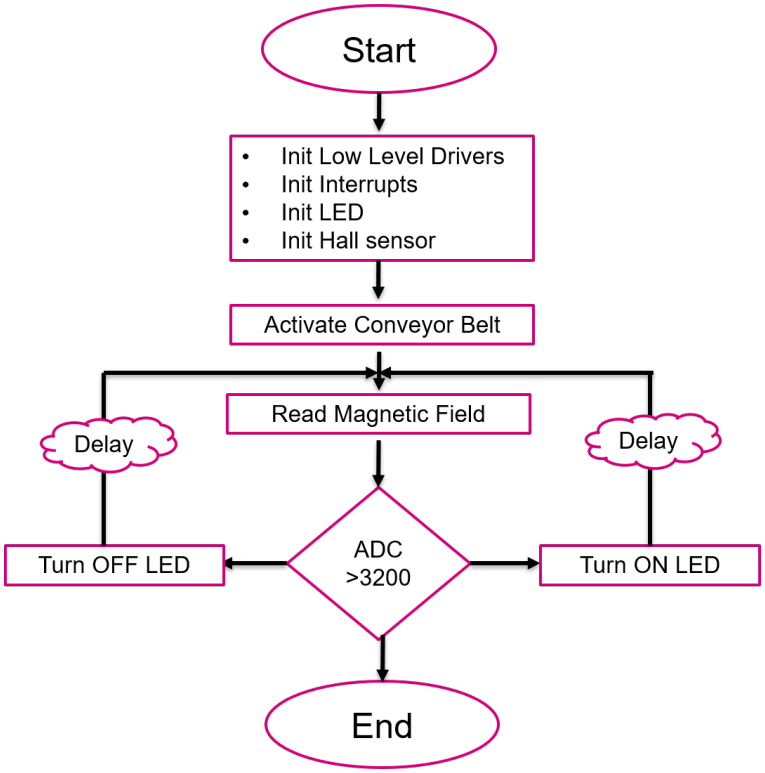
Step 7. Run automatic pin allocation

Step 8. Verify the assigned pins in the PinMap Editor

Step 9. Connect your boards using the Board View

Step 10. Write the `main()` function using AutoDevKit with high-level APIs

Figure 3. `main()` function operation block diagram



Step 11. Compile the code

Step 12. Download the code using PLS UDE

Step 13. Debug your application

Revision history

Table 1. Document revision history

Date	Version	Changes
13-Feb-2020	1	Initial release.
04-Mar-2020	2	Minor text changes.

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, please refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2020 STMicroelectronics – All rights reserved