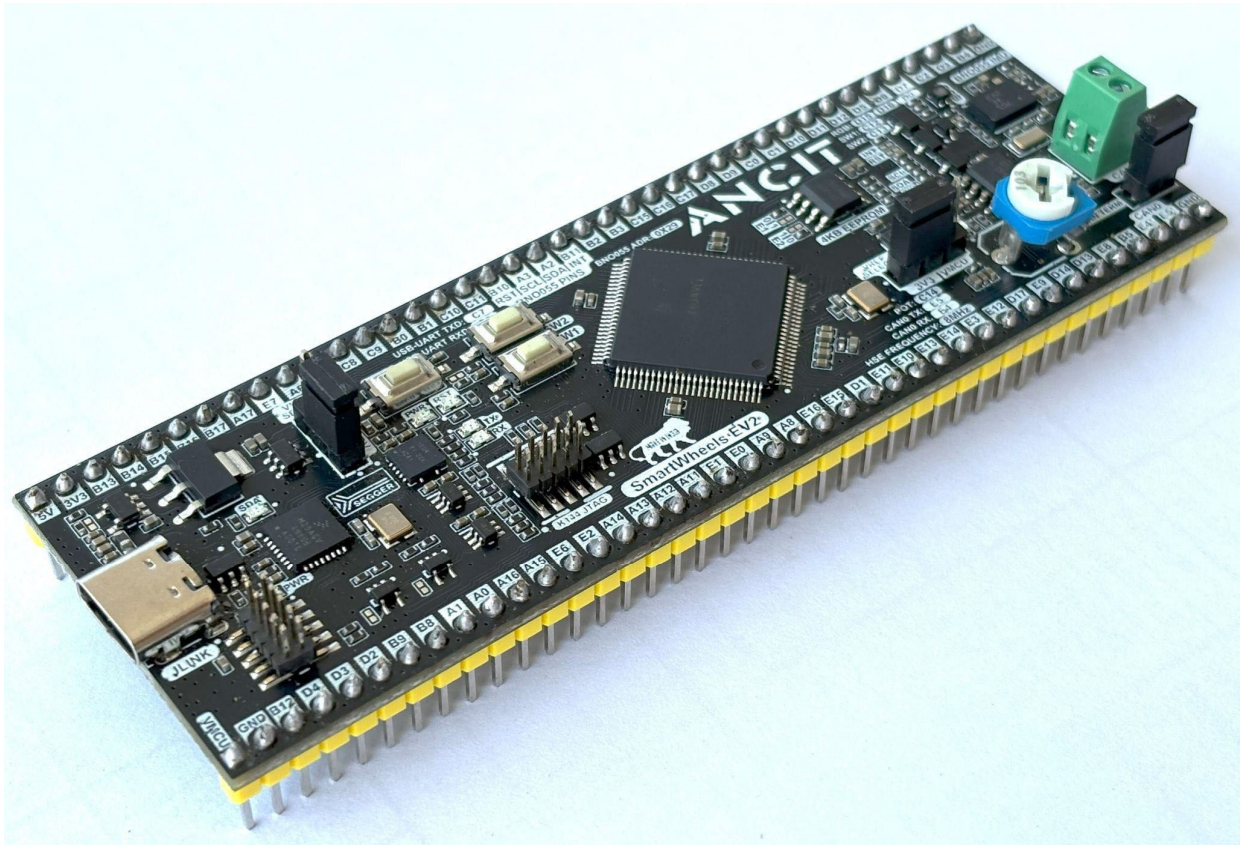
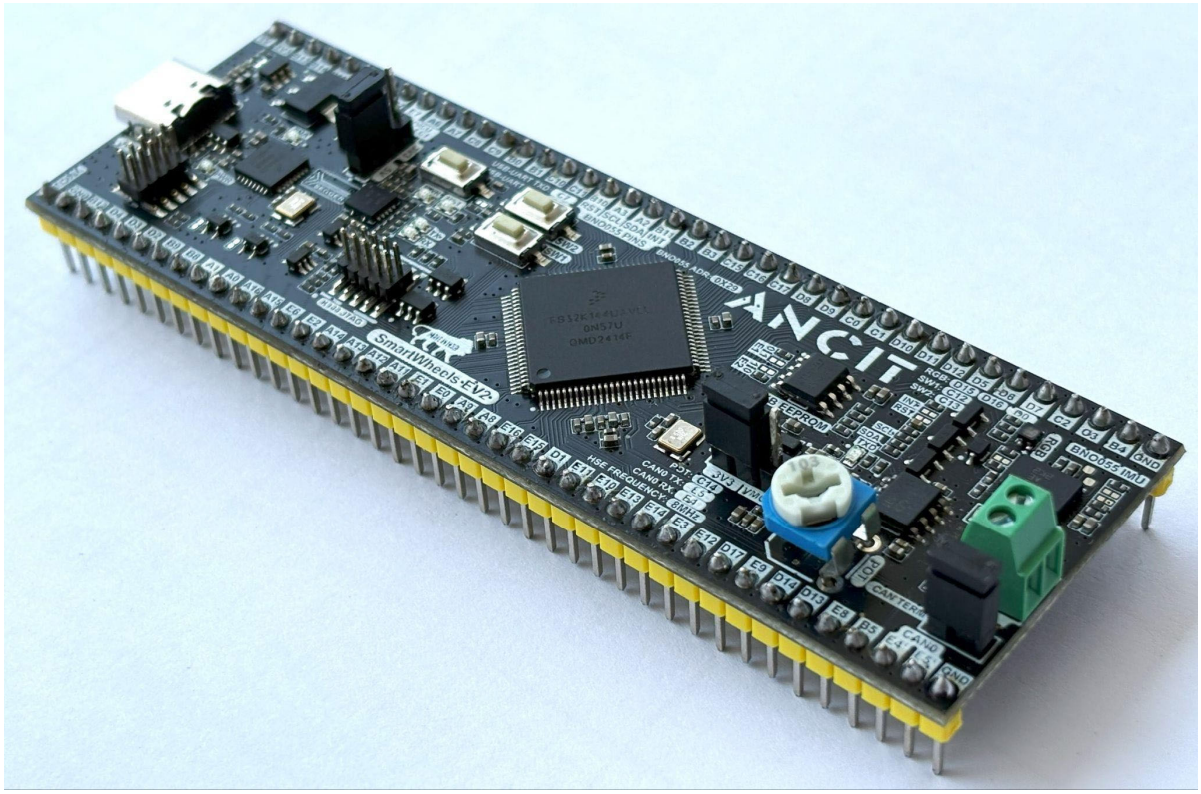
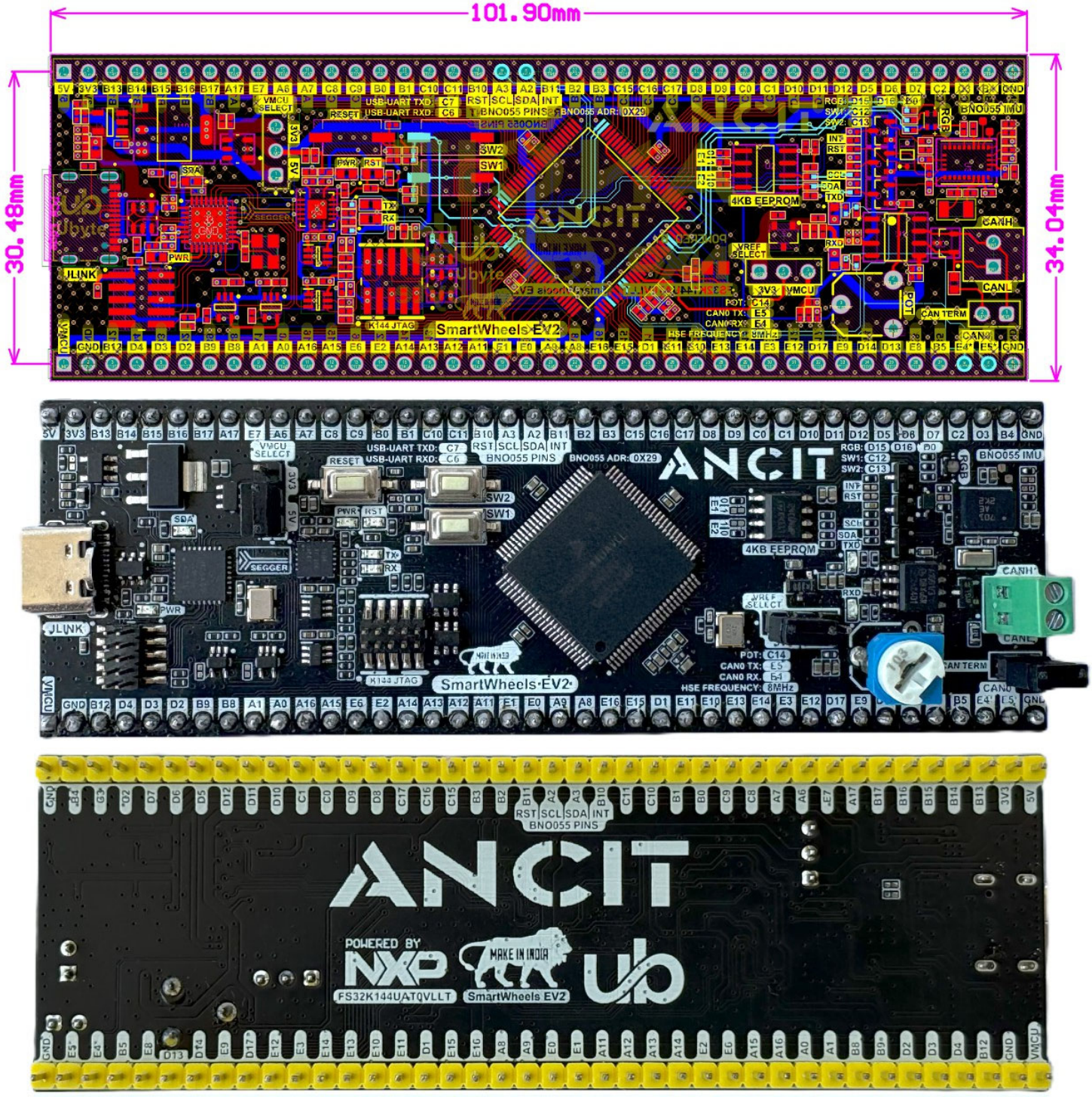

ANCIT
SmartWheels-EV2+
JLINK


ANCIT
SmartWheels-EV2+

1. S32K144 SmartWheels Micro EV2 Dimensions



INDEX

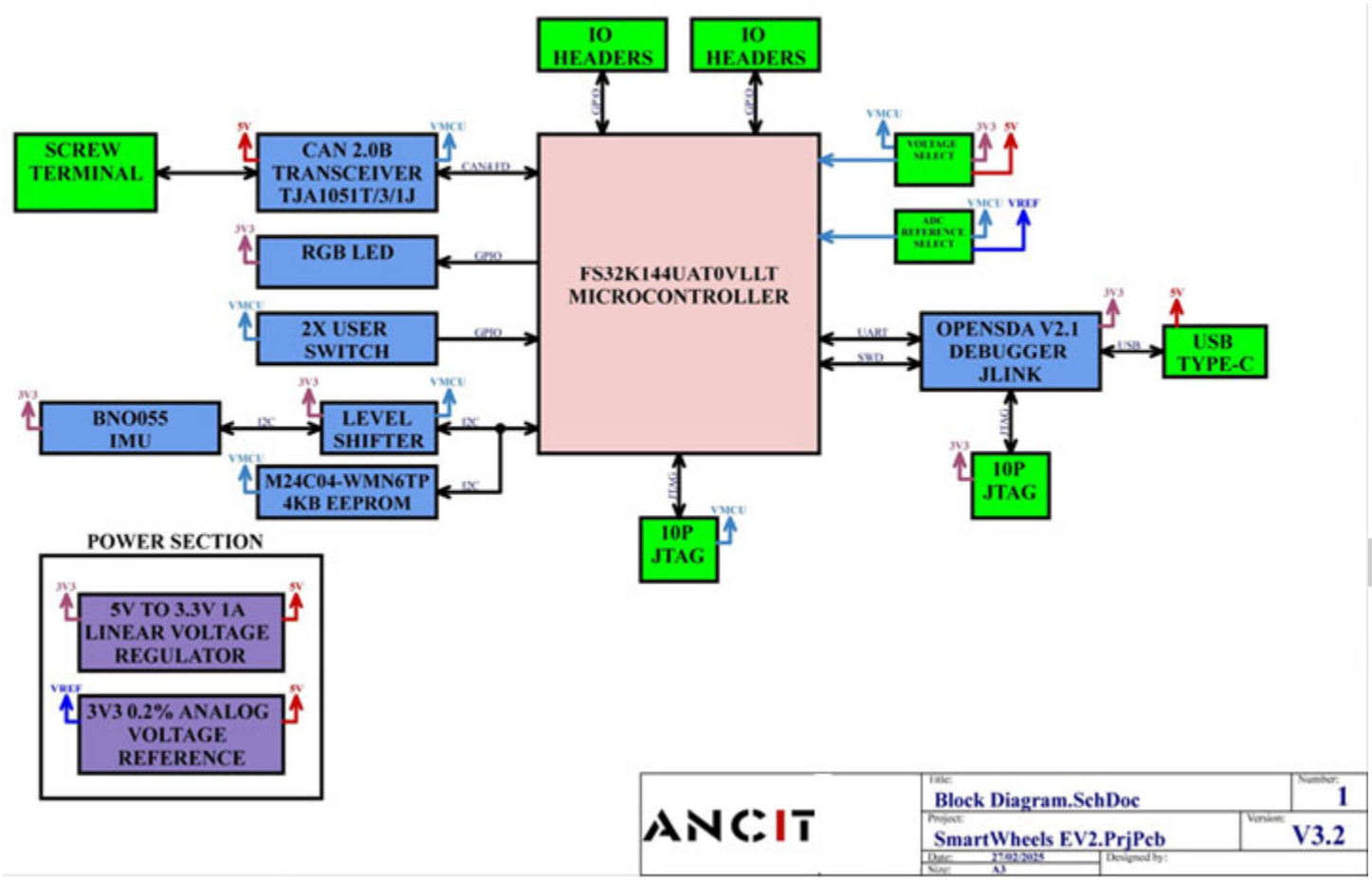
2. Introduction

The SmartWheels EV2 is a low cost development board based on the NXP S32K144UAT0VLLT microcontroller. The S32K144 offers functional safety compliance with ISO26262 ASIL Class B. The SmartWheels EV2 is equipped with onboard Segger J-Link Debugger along with CAN2.0B transceiver and Bosch BNO055 IMU sensor. This makes the SmartWheels EV2 a complete development package in itself. The SmartWheels EV2 also allows the user to switch between 3.3V and 5V logic along with the ability to change the ADC Reference voltage between 3.3V and 5V. It makes the SmartWheels EV2 extremely flexible for prototyping. The SmartWheels EV2 is made in a breadboard Compatible form factor offering 74 GPIOs for utilizing various peripherals of the S32K144 microcontroller. The board also integrated a RGB LED along with 2 push buttons.

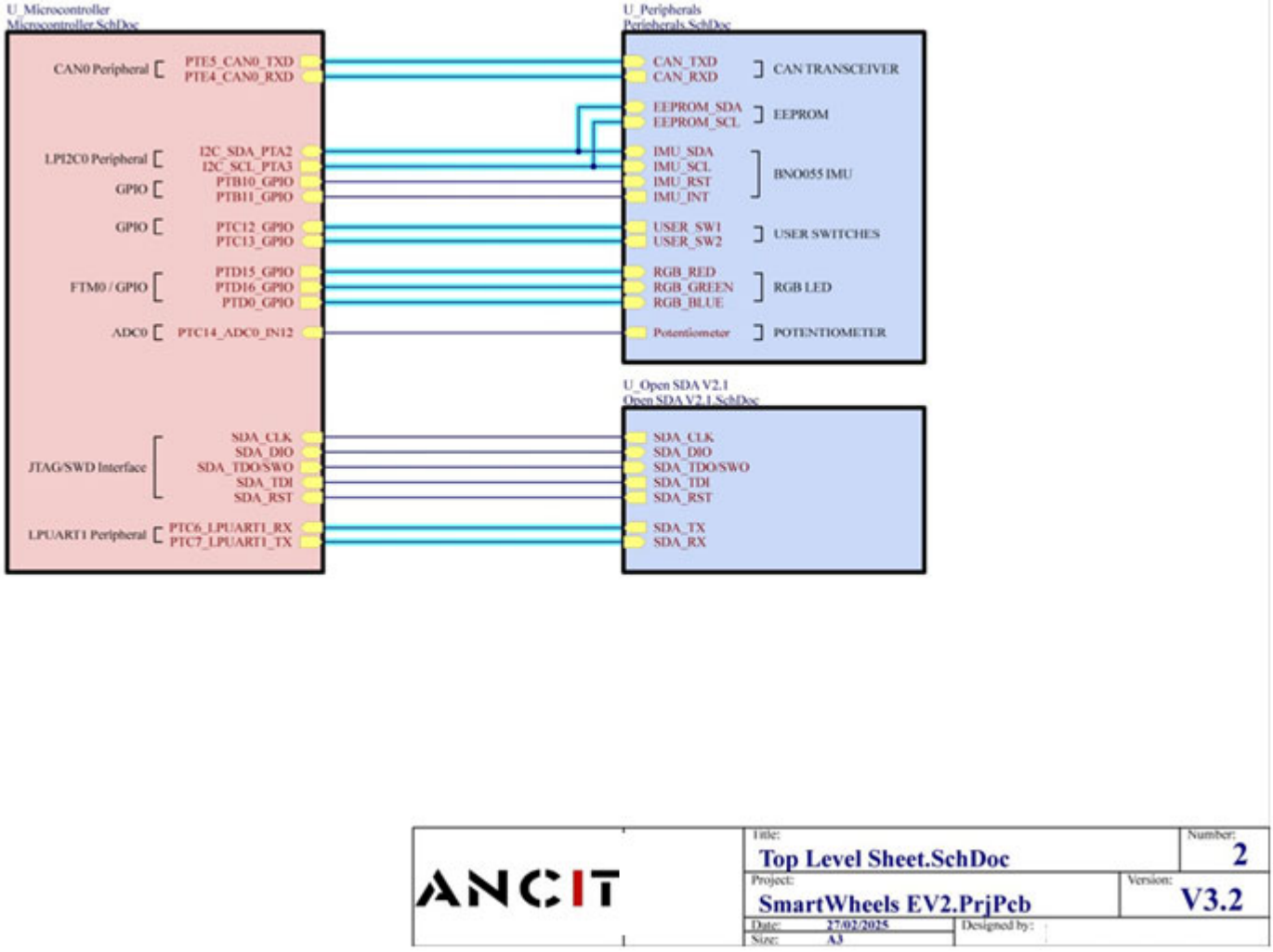
Key Features

- NXP S32K144UAT0VLLT chipset
- Onboard Segger J-Link Debugger
- USB Type-C for power supply and debugging
- Bosch BNO055 IMU sensor
- USB to UART through Segger J-Link.
- CAN2.0B Transceiver supporting CAN FD with extended frame
- 4KB EEPROM
- RGB LED
- Two User Push Buttons
- 10P JTAG for external debugger
- 8 MHz high-speed external oscillator (HSE)
- High-precision (3.3V ± 0.2%) voltage reference for ADC
- Switchable ADC reference voltage between 3.3V and 5V

3. Hardware Block Diagram



4. Top Level Sheet



ANCIT	Title: Top Level Sheet.SchDoc	Number: 2
	Project: SmartWheels EV2.PrjPcb	Version: V3.2
	Date: 27/02/2025 Size: A3	Designed by:

5. Powering the SmartWheels EV2

The SmartWheels EV2 can be powered by 4 different methods. It is highly recommended that the user supplies a voltage of 5V to the SmartWheels EV2 in order to use all sensors and peripherals on the SmartWheels EV2. The onboard Voltage Reference and the CAN transceiver require 5V to stay operational. Please note that powering the SmartWheels EV2 with 5V is not the same as selecting 5V on “VMCU SELECT" Jumper. It means that the SmartWheels EV2 board must be supplied with 5V. The user has full control of choosing appropriate MCU voltage without affecting the operation of any onboard peripheral or sensor.. However They are listed below:

1. J-Link (USB Type-C)
2. 5V Power Pin
3. 3V3 Power Pin
4. JTAG Connector

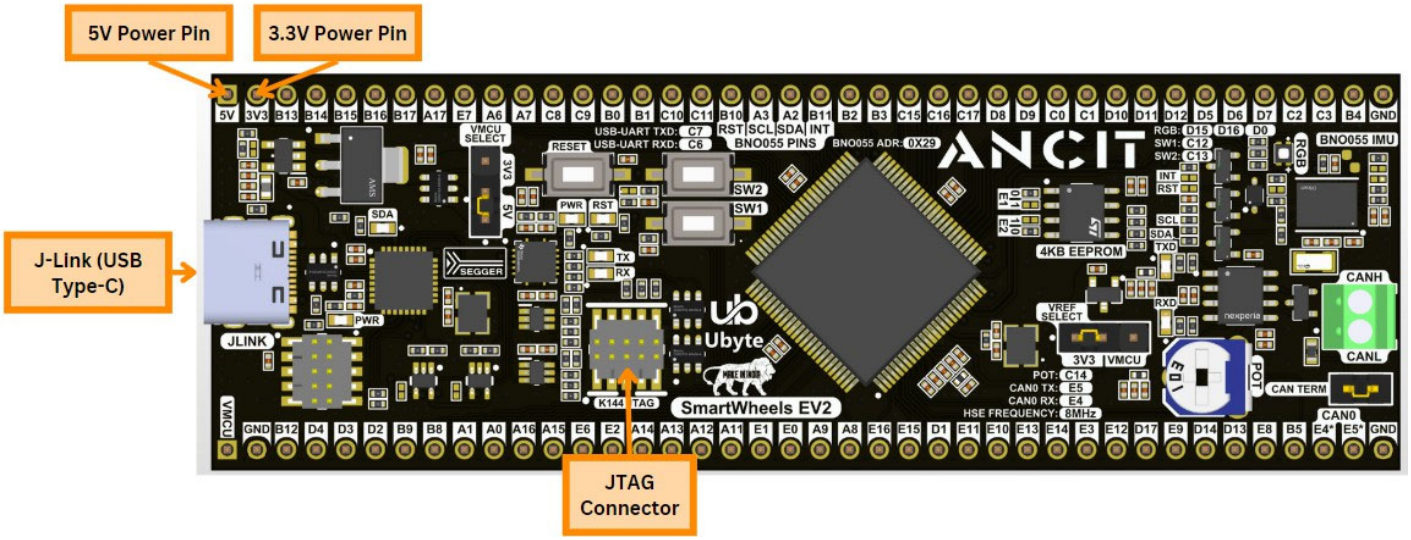


Fig 1 : Power Inputs for SmartWheels EV2

1. J-Link Power (USB Type-C)
   - The SmartWheels EV2 will be powered when the user connects the “JLINK” USB port to a host PC. This will power up the entire board through the USB Type-C connector.
   - Transient Voltage Suppression and ESD Protection have been added to the USB Type-C Port.
   - The “PWR" LED indicates that the USB type-C is supplying power to the SmartWheels EV2. Both the power LEDs on the SmartWheels EV2 will light up indicating that J-Link and the SmartWheels EV2 is receiving power.
   - The User can draw upto 1A of current from the 5V power pins and 500mA of current from the 3.3V power pin with powering the SmartWheels EV2 over USB Type-C.
   - The user can choose between the operating voltage of 3.3V and 5V for the microcontroller when powering through the USB Type-C.

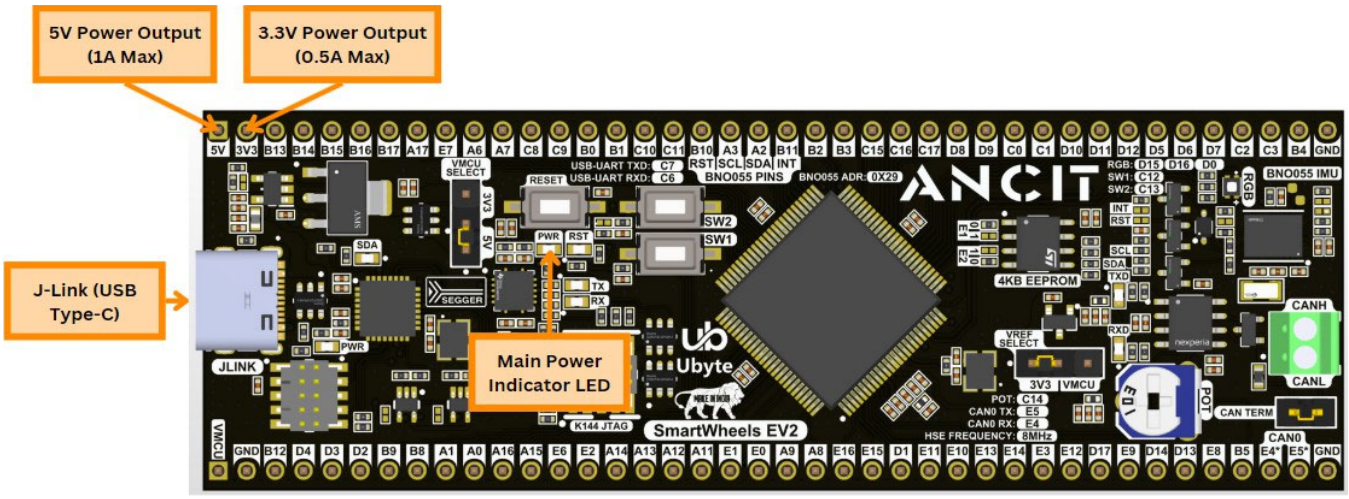


Fig 2 : Powering through J-Link USB Type-C

2. 5V Power Pin
   - The SmartWheels EV2 can be powered from a 5V external power supply directly. This is possible when the user provides power to the power pins labeled “5V.” It must be ensured that the external source is capable of providing continuous current of 1A at 5V.
   - The user must ensure that the SmartWheels EV2 is never powered from “VMCU" pin. This may damage the onboard peripherals present on the SmartWheels EV2.
   - The user can draw upto 500mA of current from 3.3V power output when powering the SmartWheels EV2 from a 5V power pin.

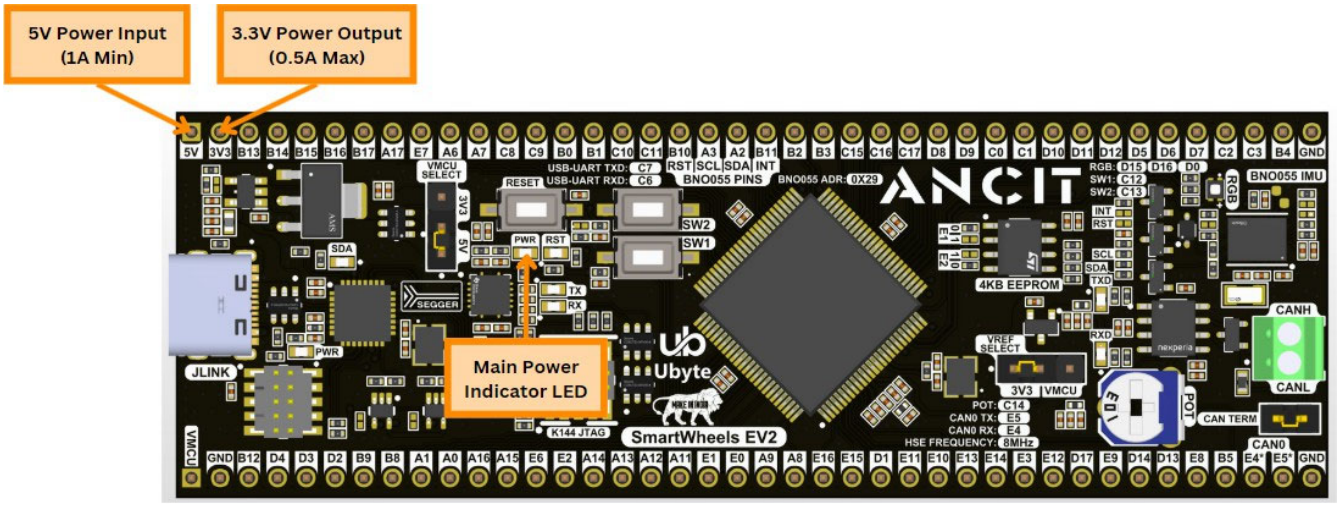


Fig 3 : Powering through 5V Power Pin

   - The user can choose between the operating voltage of 3.3V and 5V for the microcontroller when powering through the 5V power pin.
   - The “PWR" LED is mentioned in Fig 3. indicates that power is being received by the SmartWheels EV2.

3. 3V3 Power Pin
   - The SmartWheels EV2 can be powered from a 3.3V external power supply directly. This is possible when the user provides power to the power pins labeled “3V3.” It must be ensured that the external source is capable of providing continuous current of 1A at 3.3V.
   - The user must change the “VMCU SELECT” jumper to “3V3” as 5V will not be present on the SmartWheels EV2 when powering through the 3.3V Power Pin. The user must also change the “VREF SELECT” jumper to “VMCU.”
   - The onboard Voltage reference is inoperational when powering the SmartWheels EV2 with 3.3V.
   - The user must also note that the CAN Transceiver will not be operational when powering the SmartWheels EV2 with 3.3V Power Pin as CAN requires 5V to operate. All other peripherals will remain operational.
   - The user must ensure that the SmartWheels EV2 is never powered from “VMCU" pin. This may damage the onboard peripherals present on the SmartWheels EV2.
   - The user can not choose between the operating voltage of 3.3V and 5V for the microcontroller when powering through the 3.3V power pin as only 3.3V will be available on the SmartWheels EV2.
   - The “PWR" LED is mentioned in Fig 4. indicates that power is being received by the SmartWheels EV2.

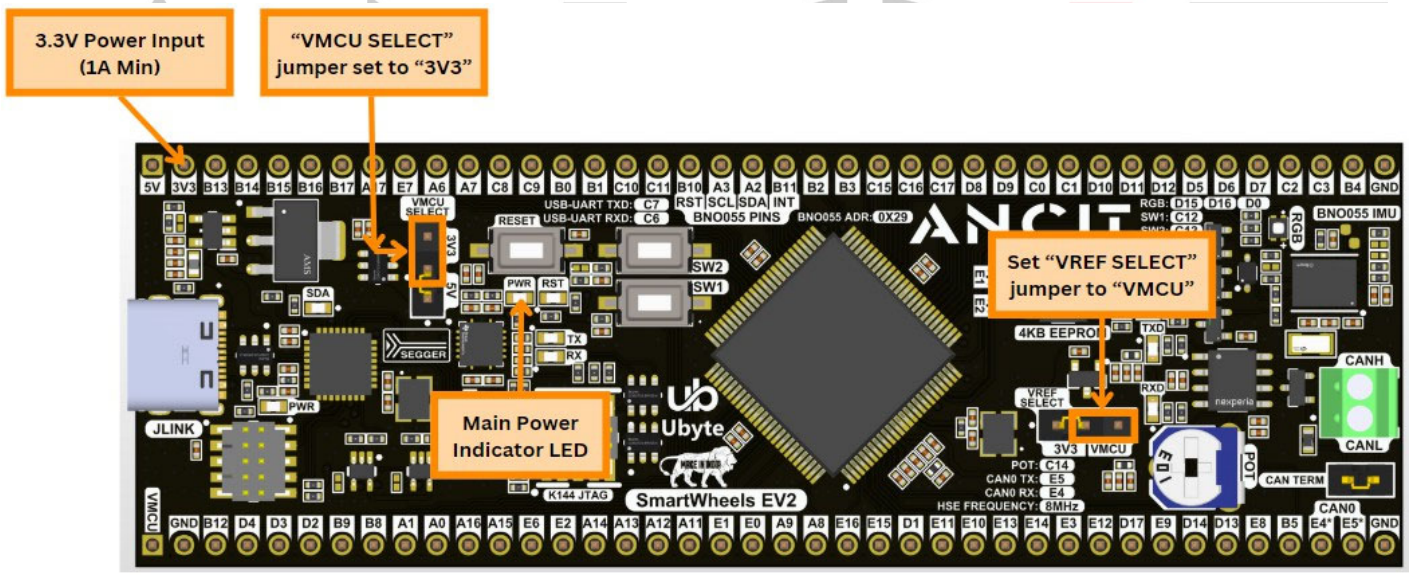


Fig 4 : Powering through 3V3 Power Pin

4. Powering through JTAG Connector
   - The SmartWheels EV2 can be powered from a “K144 JTAG" header directly. Although it is not recommended to power the SmartWheels EV2 from JTAG. The power pin of the JTAG is only meant for providing the Target Voltage to the external debugger so the voltage levels of the signals can be adjusted accordingly on external debuggers.
   - However if the user still chooses to power the SmartWheels EV2 from External Debugger, then the user must ensure that the “VTGT" or the “VTarget" voltage of the debugger is set to 3.3V when the “VMCU" jumper is at “3V3.” Providing 5V to the SmartWheels EV2 when the “VMCU SELECT" jumper is set at

"3V3" will damage the onboard peripherals connected on the SmartWheels EV2. If the user is powering the SmartWheels EV2 with 3.3V, the user must also change the "VREF SELECT" jumper to "VMCU."

- The user can also power the SmartWheels EV2 with 5V through the "K144 JTAG" header through external debugger only when the "VMCU SELECT" jumper is set to "5V."
- The "PWR" LED is mentioned in Fig 5. indicates that power is being received by the SmartWheels EV2.
- The risk involved in powering the board through the "K144 JTAG" header is the reason why this powering method is not recommended.
- The user won't be able to use the CAN Transceiver when powering the board with 3.3V through External JTAG over the "K144 JTAG" header.
- The onboard Voltage reference is inoperational when powering the SmartWheels EV2 with 3.3V.

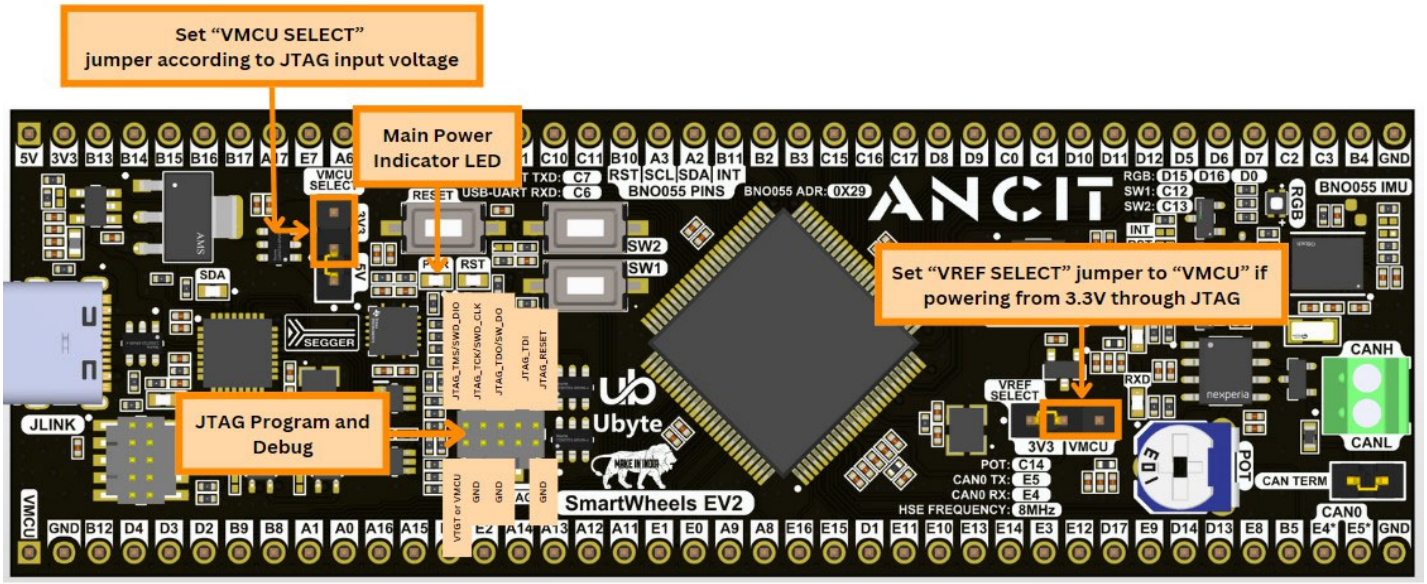


Fig 5 : Powering through JTAG Connector

6. Switching Operating Voltage of S32K144 MCU

- The SmartWheels EV2 allows the user to switch the operating voltage of the S32K144 microcontroller when powering through USB Type-C or the 5V power pin at 5V. Therefore all GPIOs on SmartWheels EV2 can be switched between 5V and 3.3V operation. This allows the user to use SmartWheels EV2 for a variety of applications that may require different logic level operations. This also makes it suitable for places where the SmartWheels EV2 is being used on a custom board where the user might have only 5V or 3.3V operation possible.
- Please note that before the user makes any changes to the “VMCU SELECT" jumper, it’s highly advised to power down the SmartWheels EV2 board. Changing the jumper settings on the “VMCU SELECT" header may damage the microcontroller.
- To switch the operating voltage of S32K144 Microcontroller on SmartWheels EV2, the user simply needs to locate the “VMCU SELECT " jumper. For selecting a MCU operating voltage of 5V, the user needs to place the jumper on “5V" on the “VMCU SELECT" jumper. Similarly for selecting a MCU operating voltage of 3.3V, the user needs to place the jumper on “3V3" on the “VMCU SELECT " jumper.
- It must also be noted that the Reference Analog to Digital Converter also operates on VMCU. Therefore if the user selects the MCU voltage of 3.3V, the ADC reference must be 3.3V or lower. The SmartWheels EV2 is designed so that the user won’t be able to switch the ADC reference voltage higher than VMCU voltage. Please refer to the “Switching Voltage Reference on SmartWheels EV2” to know more.

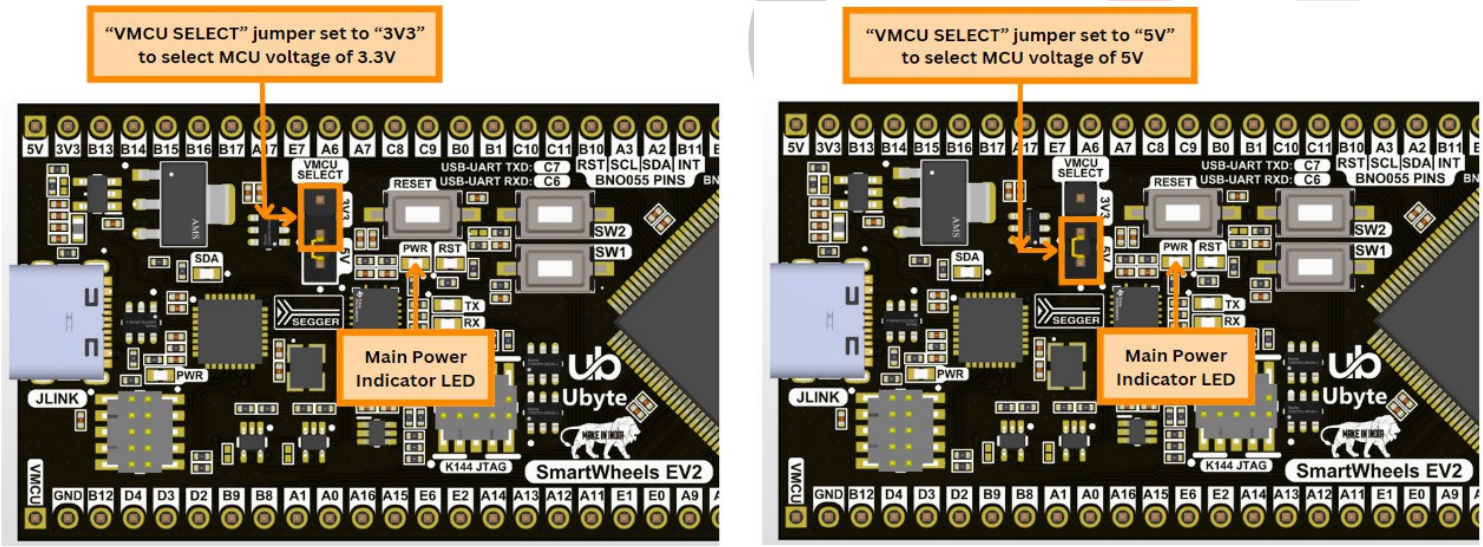


Fig 6 : Selecting VMCU Voltage Level

7. Switching Analog Reference Voltage

- The SmartWheels EV2 allows the user to switch the Analog to Digital Converter Reference Voltage. The SmartWheels EV2 comes equipped with a 3.3V ±0.2% Voltage Reference for the ADC. However some users require the Analog to Digital Convert to sample input voltages upto 5V therefore SmartWheels EV2 provides the option to switch between 3.3V ±0.2% Voltage Reference and VMCU Logic Level.
- The VMCU (operating voltage of S32K144 MCU) can therefore be set to 5V in order to make the Analog to Digital converter operate at a 5V Voltage reference. Switching the ADC Reference Voltage to 5V required the user to switch the VMCU Voltage to 5V as Analog to Digital converter of the S32K144 uses VMCU as analog to digital converter supply voltage and the reference voltage can not be higher than the ADC operating Voltage.
- Switching between 3.3V ±0.2% Voltage reference and VMCU logic level instead of switching between 3.3V ±0.2% and 5V logic ensures that the user can't damage the microcontroller accidentally by supplying higher voltage on the ADC reference than input power voltage of the ADC. The 5V voltage reference is taken directly from the 5V power supply therefore it must be noted that the user may experience unstable readings when using 5V as voltage reference. Proper filtering is provided for 5V in order to minimize the noise at the VRef pin on the microcontroller.
- The user must disconnect all power to the SmartWheels EV2. Changing the voltage reference levels without powering down the board may damage the microcontroller.
- To use 5V for ADC reference, the user simply needs to change the "VREF SEL" jumper to "VMCU" and "VMCU SELECT" jumper to "5V" to effectively change the reference voltage of the ADC to 5V. To use the 3.3V ±0.2% Voltage reference, the user simply needs to change the "VREF SEL" jumper to "3V3" to select the 3.3V ±0.2% voltage reference as the ADC reference voltage. The user is free to switch between 3.3V and 5V on the VMCU when using 3.3V voltage reference.

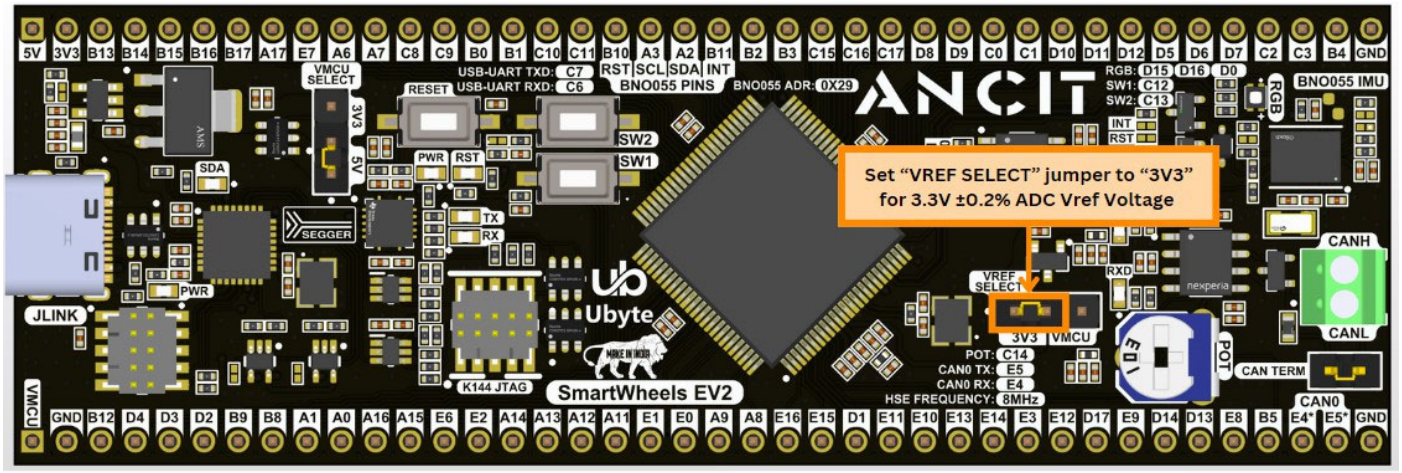


Fig 7 : ADC VRef set to 3.3V ±0.2% Voltage Reference

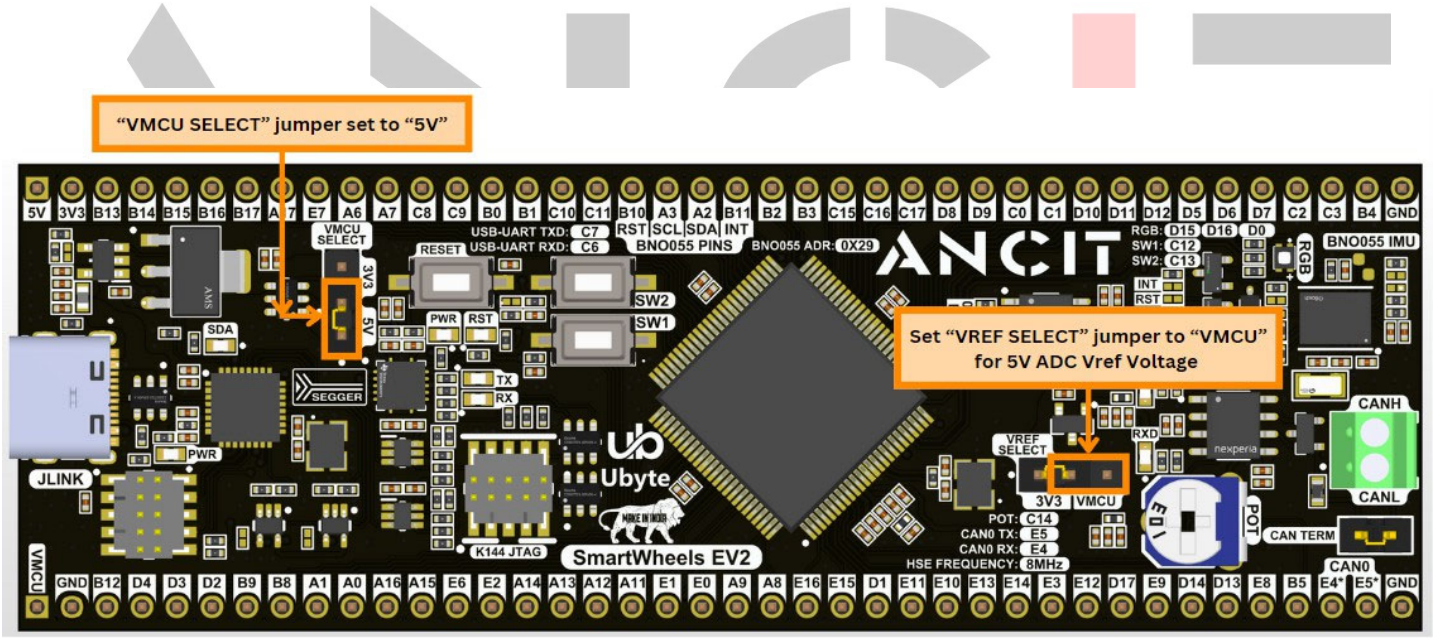


Fig 8 : ADC VRef set to 5V

8. Segger J-Link Debugger

The SmartWheels EV2 is equipped with an official Segger J-Link debugger that can be used for programming and debugging the SmartWheels EV2 through SWD interface. The Segger J-Link debugger can be accessed by the user through the “JLINK” USB Type-C port on the SmartWheels EV2. It must be ensured by the user that the software utilities and drivers for the segger J-Link are installed on the host PC. If the user is not having any external power source for the SmartWheels EV2, he can power the entire board through USB Type-C itself while debugging. Refer to the “Powering the SmartWheels EV2” section to know more. With the onboard Segger J-Link, the user can access all Segger J-Link utilities. The onboard Debugger can also be used to program and debug the microcontroller from the NXP S32 Design Studio directly making it ideal for development and prototyping. You can learn more about programming and debugging the SmartWheels EV2 from future documentation releases.

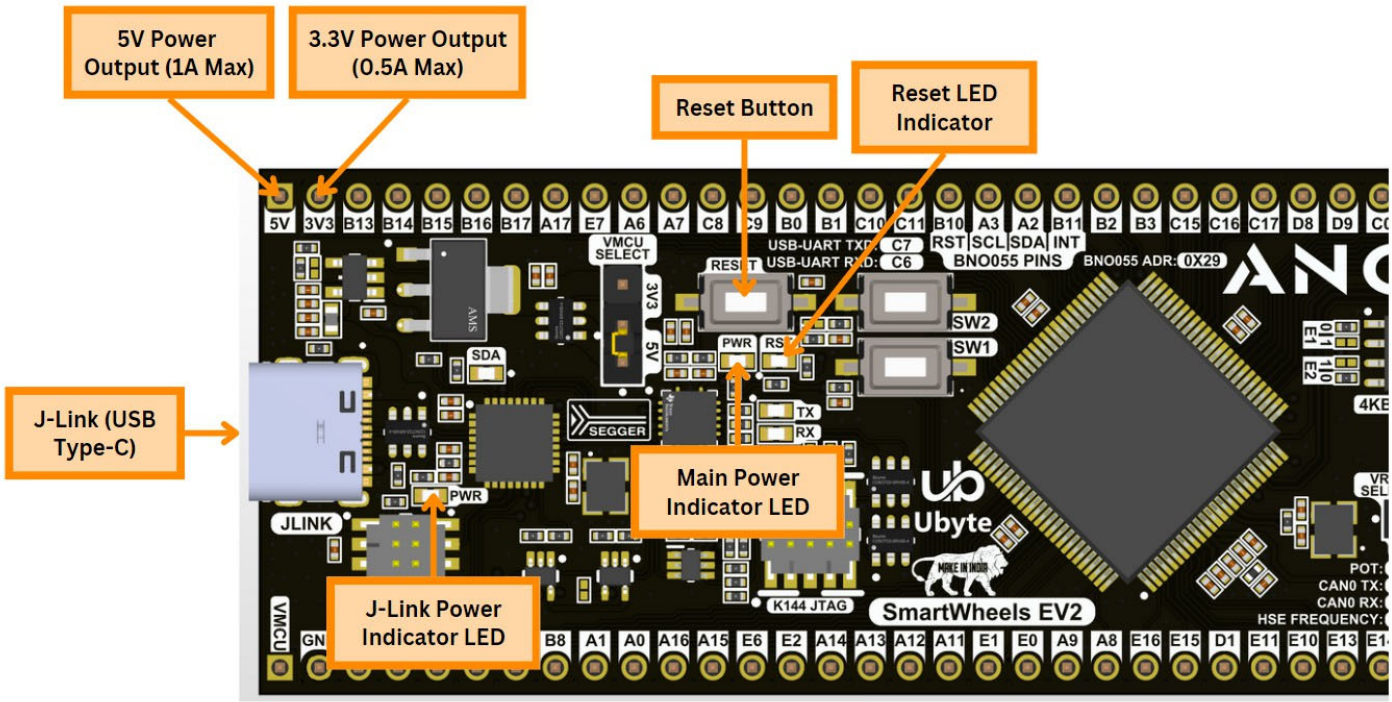


Fig 9 : J-Link Jumper Settings and Indicator LEDs

Normal operation of the Segger J-Link debugger is signified by 2 LED indicators present on the PCB. “PWR” indicator located close to the USB type C port indicates that the J-Link is receiving power. Another LED indicator shown in Fig 10. is used to signal various operations of the Segger J-Link listed in the table below.

Debugger Mode	State Description	J-Link LED Pattern
Bootloader	Prior to USB enumeration	Off
Bootloader	Idle – running normally with no error conditions	Blinking: 500ms on, 500ms off
Bootloader	Error	2 seconds off followed by 8 rapid on/off blinks
Application	Prior to USB enumeration	Off
Application	Running normally with no error conditions and no USB activity	On
Application	USB activity (for example, MSD or CDC)	Blinking
Application	Error	2 seconds off followed by 8 rapid on/off blinks

The user won’t usually see any warning related to bootloader on the onboard J-Link. When the J-Link is detected by the PC successfully, the LED indicator turns on. While uploading firmware or debugging, the LED indicator will blink signifying normal operation.

The J-Link can hard reset the microcontroller and therefore the user might see the “RST" LED indicator blinking while uploading the firmware on the SmartWheels EV2. The “RST" LED lights up when the Reset Pin of the microcontroller is pulled low. This can either be done by the onboard J-Link debugger or when the user presses the “RESET” button on the SmartWheels EV2. Pulling the Reset Pin low initiates a Reset on the SmartWheels EV2.

USB to UART on J-Link

The J-Link can also be used like a USB to UART Converter. When the user connects the SmartWheels EV2 to PC through the “JLINK” USB port, a COM port is also created that can be found by going to “Device Manager” on Windows 7/10/11. Therefore the user can use any Serial monitor application to send and receive messages through UART on the COM port that is created by J-Link.

Following pins are used for the USB to UART features:

Peripheral Used: LPUART1

LPUART1_RX : PTC6 (Uart Receive pin for microcontroller)

LPUART1_TX : PTC7 (Uart Transmit pin for microcontroller)

The TX and RX data line of the J-Link debugger have dedicated Indicator LEDs for indicating the transmission and reception of data through the J-Link Debugger interface. The “TX" LED indicates that the data is being transmitted from the SmartWheels EV2 to the host PC through the J-Link COM port. The “RX” LED indicates that the data is being received on the SmartWheels EV2 from the host PC through the J-Link COM port.

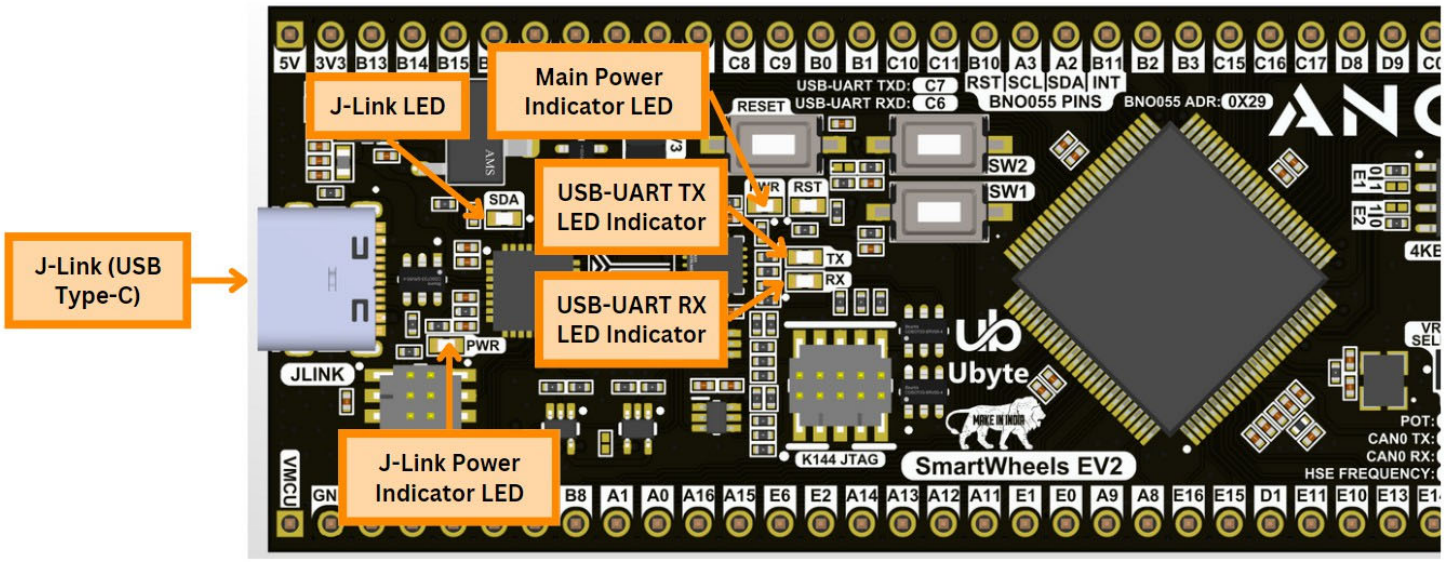


Fig 10 : J-Link USB to UART Indicator LEDs

MCU JTAG

The “K144 JTAG" can be used by the user to program and debug the target microcontroller over SWD or JTAG interface using external debuggers using the standard 10P 1.27mm connector. The user can purchase an adapter board from ANCIT Consulting in case the user requires a 20P 2.54mm JTAG connector to connect to an external debugger. The user needs to ensure that the onboard J-Link debugger is not connected to the host PC over USB. Therefore ensuring that the onboard debugger doesn’t cause issues while debugging the microcontroller using external debuggers. Therefore the user can power the SmartWheels EV2 using the 5V

power pin. Please refer to the “Powering the SmartWheels EV2” section for more information on how to power the board with the 5V power pin. The onboard J-Link debugger has level shifters to switch to necessary logic level as per the voltage on VMCU voltage automatically. Therefore when connecting to external debuggers, the VTGT pin of the “K144 JTAG" is used to supply target voltage to the external debugger so that the debugger is able to adjust to the logic levels of the JTAG/SWD interface.

The pinout and the location of the “K144 JTAG” connector is shown on Fig 11

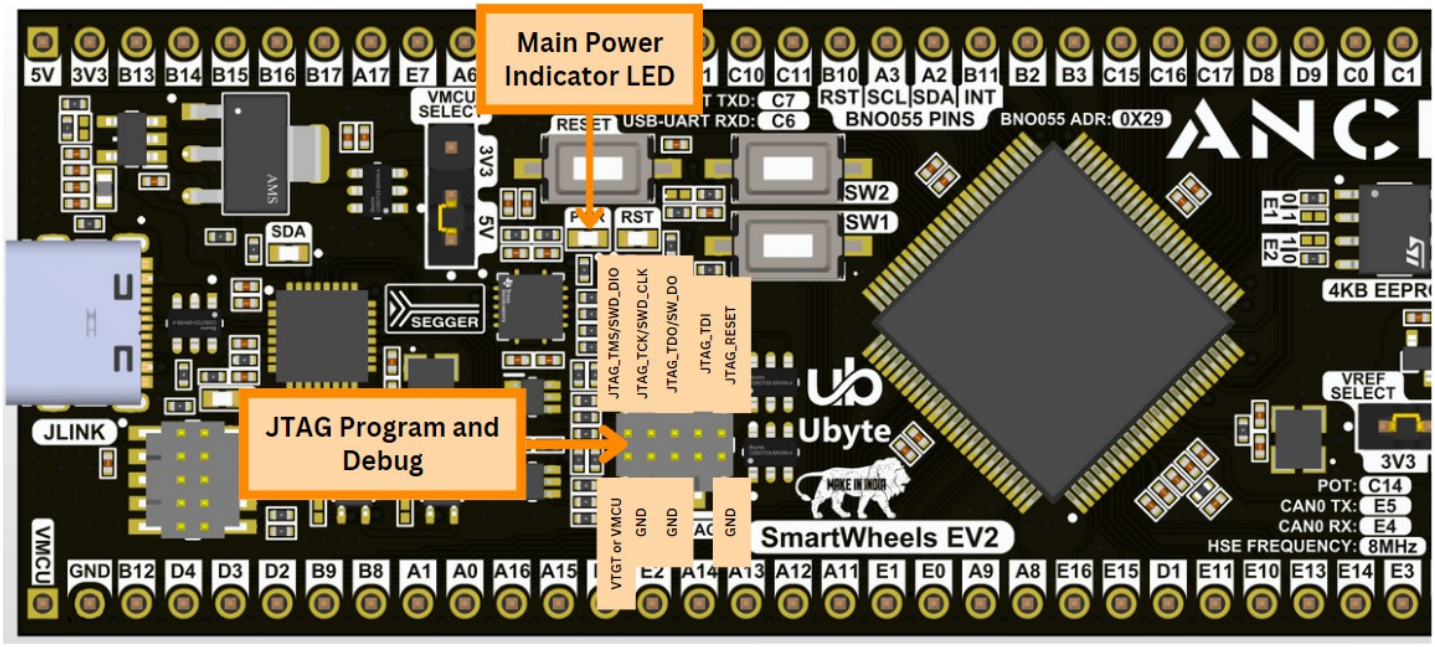


Fig 11 : JTAG connector on SmartWheels EV2

9. CAN Transceiver

The SmartWheels EV2 is equipped with a CAN transceiver. The CAN transceiver supports CAN FD upto 5 Mbit/s.

The CAN transceiver has the following specifications:

- MPN of Transceiver IC used: TJA1051T/3/1J
- Datasheet Link: High-speed CAN transceiver (nxp.com)
- Interfacing Pins:
  - Peripheral Used: CAN0
  - CAN0_TX: PTE5
  - CAN0_RX: PTE4
- CAN Termination: 120Ω Standard Termination (Can be disabled)
- Maximum Stable Data Rate: 5 Mbit/s
- Sleep Modes: No sleep modes available
- CAN Phy Standards: ISO 11898-2:2016 and SAE J2284-1 to SAE J2284-5 compliant
- AEC-Q100 qualified: Yes

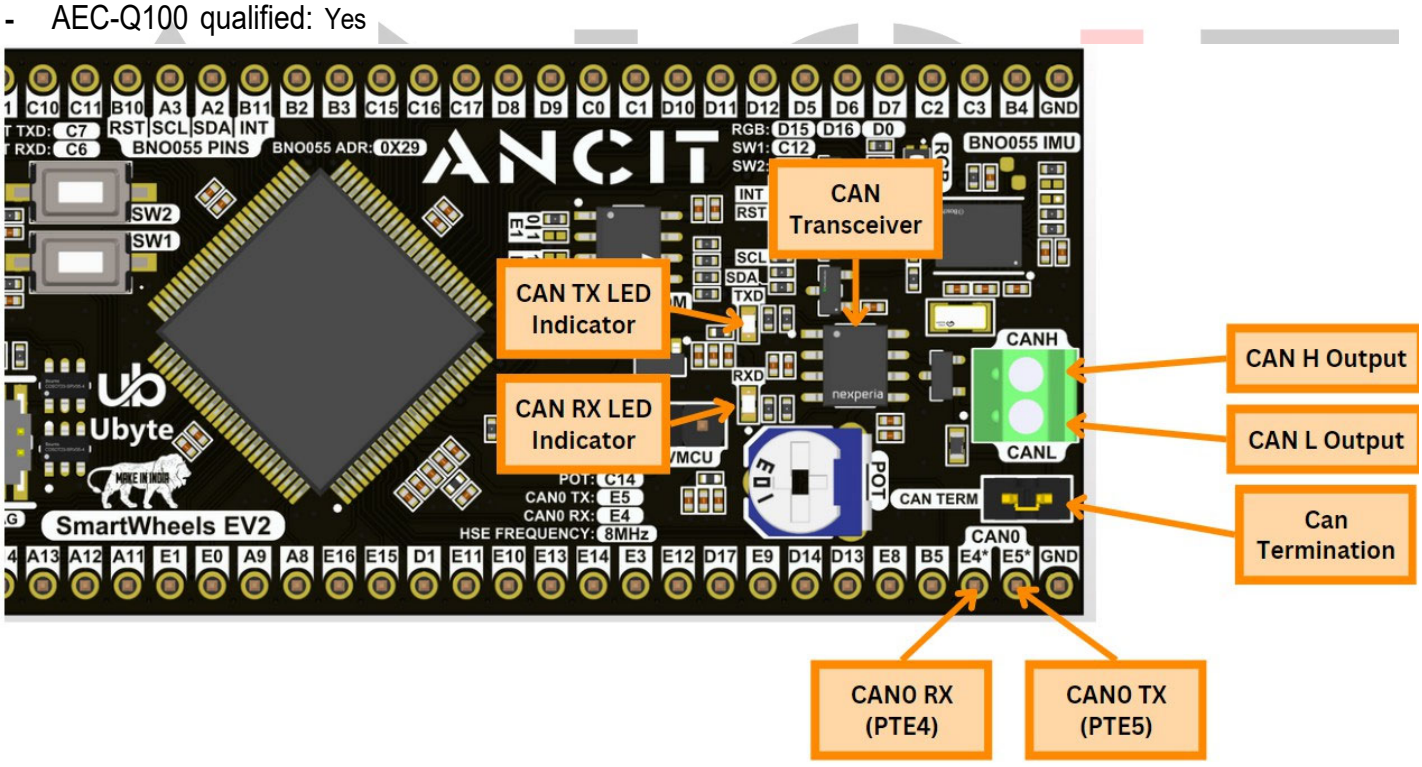


Fig 12 : CAN output, CAN Termination and Data Indicators LEDs

Other specifications for the CAN transceiver can be found in the datasheet link provided. The Data lines CAN0_TX and CAN0_RX have dedicated LED indicators. “RXD” LED indicates that the data is being received on CAN bus. The “TXD” LED indicates that the data is being transmitted on the CAN bus. It must be noted that the “TX1” and “RX1” LEDs will light up simultaneously when data is being transmitted on the CAN bus. This is

a characteristic of CAN bus which allows the microcontroller to monitor the state of the CAN bus when data is being transmitted to detect collisions.

The user can enable and disable the CAN termination on the CAN transceiver by using the “CAN TERM” jumper located right next to the CAN output screw terminals. Removing the jumper will disable the CAN termination while placing the jumper will enable the CAN termination.

The user can access the CANH and CANL output of the CAN transceiver through the screw terminal.

The user must also note that the CAN0_TXD and CAN0_RXD data lines are given on the IO headers as well. They are internally connected to the onboard CAN Transceiver. The user can remove the jumper resistors to disable the onboard CAN transceiver to use external CAN transceivers on the same datalines if needed. Please refer to the picture below to check the jumpers to be removed in order to disable the onboard CAN Transceiver.

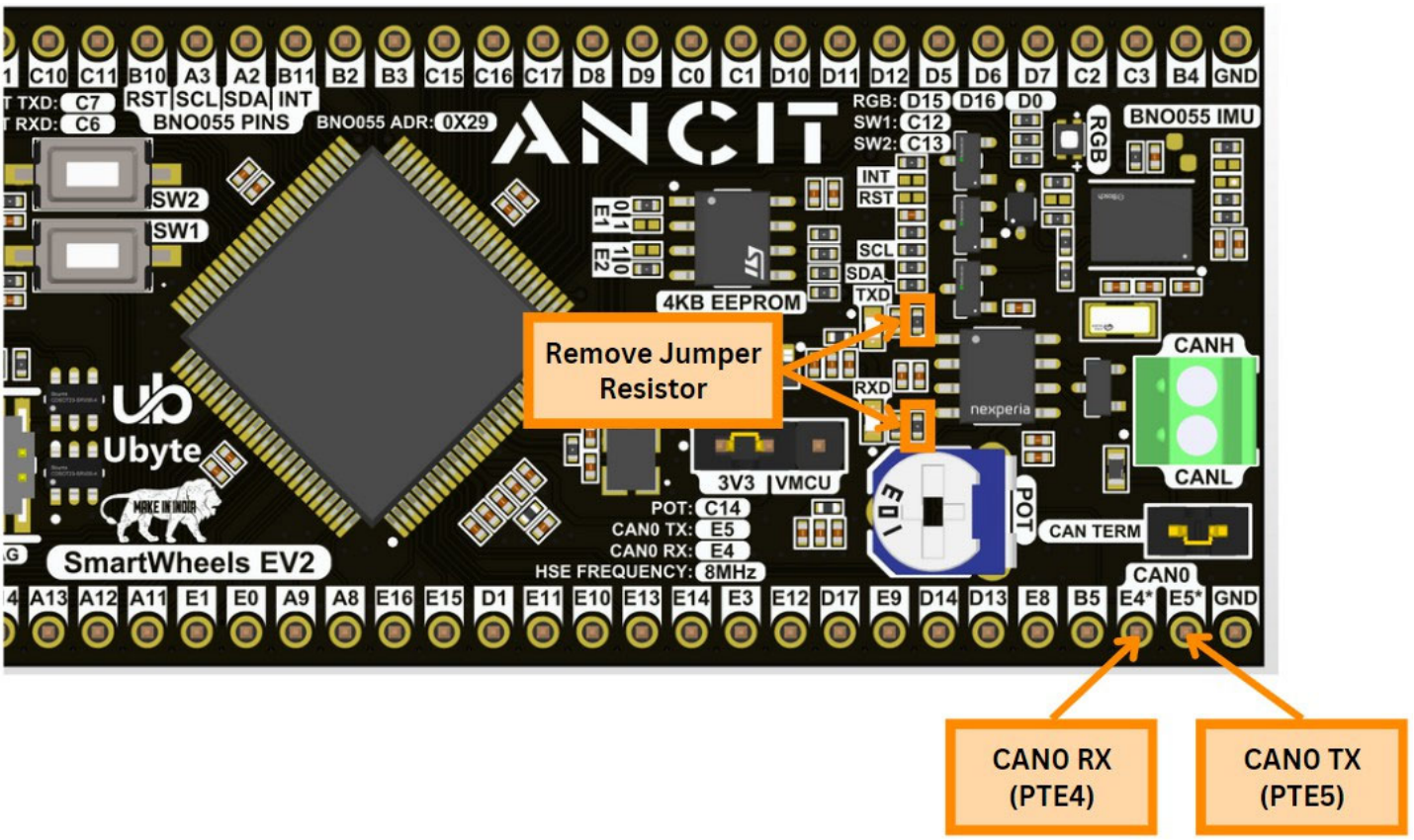


Fig 13 : Isolating CAN Transceiver from Microcontroller

10. RGB LEDs, Potentiometer and User Push Buttons

The SmartWheels EV2 is equipped with one RGB LEDs, one potentiometer and two User push Buttons. The specifications and pin mapping of the RGB LED, Potentiometer and User Push Buttons is given below:

RGB LED:

- Red LED: PTD15
- Green LED: PTD16
- Blue LED: PTD0

Potentiometer:

- ADC pin: PTC14
- Output Voltage: 0 to 3.3V
- Potentiometer Value: 10K ohms

User Switch:

- User Switch 1: PTC12
- User Switch 2: PTC13
- Default State of User Switches: Pull up

The RGB LEDs are driven when the microcontroller pin connected to the LED is driven low. Therefore the output to the LEDs must be inverted. The LEDs used are common anode LEDs having their anode connected to 3.3V. The user can also implement PWM operation to control the brightness of the onboard RGB LEDs.

The potentiometer can be used as an input for analog to digital converter so that the user can learn to use the ADC and also use the onboard potentiometer for various demo examples. Please note that when the ADC reference is changed to 5V, the onboard potentiometer won't be able to produce the full scale readings on the ADC pins. The User buttons are labeled as "SW1" and "SW2" are active low. Pressing the button will short the corresponding microcontroller pin to ground. Hardware debouncing is also implemented to the user buttons.

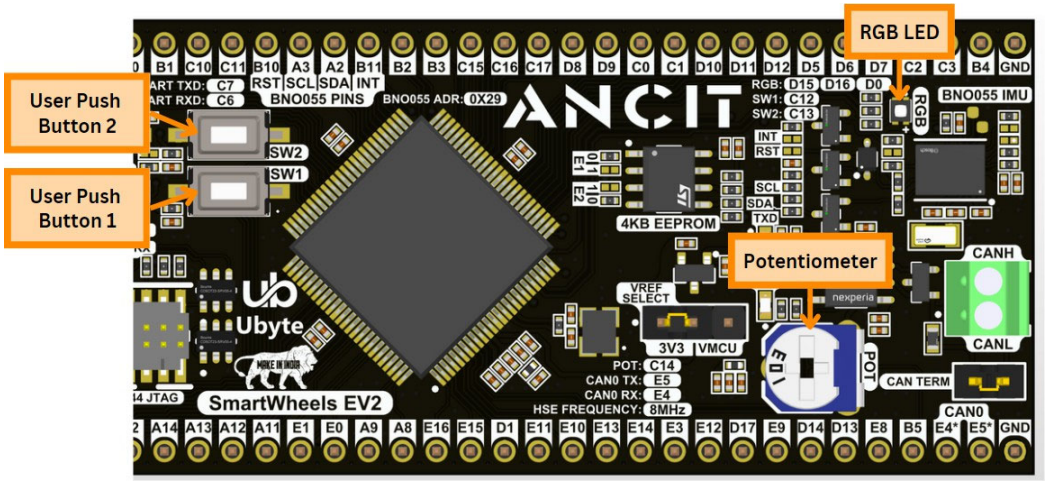


Fig 14 : RGB LEDs and User Buttons on SmartWheels EV2

11. Bosch BNO055 IMU Sensor

The SmartWheels EV2 is equipped with a Bosch BNO055 IMU sensor. The BNO055 IMU sensor makes the SmartWheels EV2 suitable for telematics applications. The BNO055 has an advanced triaxial 16 bit gyroscope, a versatile, leading edge triaxial 14 bit accelerometer and a full performance geomagnetic sensor. The BNO055 gives output of quaternion, euler angles, rotation vector, Linear acceleration, Gravity and Heading. The user can access the specifications of the BNO055 IMU from the datasheet linked below. ANCIT Consulting will be releasing further documentation and software resources to interface the BNO055 with S32K144 microcontroller. Please visit ANCIT Consulting Website to know more.

The CAN transceiver has the following specifications:

- MPN: Bosch BN0055
- Datasheet Link: BNO055 Datasheet (bosch-sensortec.com)
- Interfacing Pins:
  - Peripheral Used: I2C
  - I2C_SDA: PTA2
  - I2C_SCL: PTA3
  - BNO055 Reset (RST): B10 (disabled by default)
  - BNP055 interrupt (INT): B11 (disabled by default)
- I2C Address: 0X29
- Operating Voltage: 3.3V (onboard Level shifter)

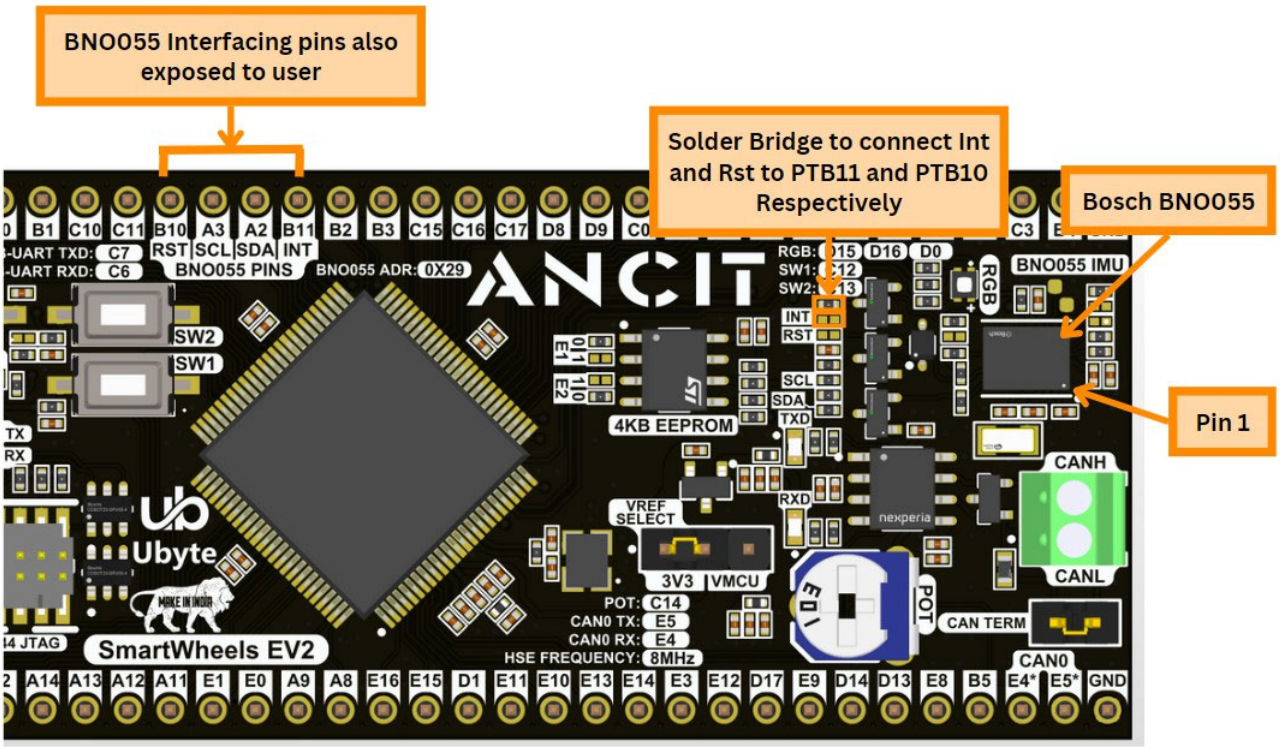


Fig 15 : Bosch BNO055 IMU location and Orientation

- The BNO055 is located towards the right side of the SmartWheels EV2. The user can take reference of pin number 1 for determining the orientation of the BNO055 on the SmartWheels EV2 as shown in Fig 15.
- The BNO055 IMU has an operating voltage of 3.3V. Therefore the SmartWheels EV2 is equipped with a level shifter so that the user can operate the BNO055 even if the microcontroller operating voltage is set to 5V.
- 4 pins of the S32K144 microcontroller are used to interface the BNO055 out of which 2 are disabled by default. All 4 pins are also available to the user through the IO headers so that they can be used for other functions as well. The “INT” and “RST” pins of the BNO055 are not connected by default to the microcontroller. Refer to “Isolating BNO055 IMU from MCU” section to know more.
- The S32K144 microcontroller has only 1 I2C peripheral so the I2C pins that are exposed to the IO headers are also shared by the onboard BNO055 IMU. The user must ensure that the external I2C sensor connected to the SmartWheels EV2 must not have the same I2C address as the BNO055 IMU. The default I2C address of BNO055 is “0X29.” In case of a conflict in I2C address, the user must change the I2C address of the external sensor/peripheral or the user can change the I2C address on the onboard BNO055 IMU as well. Refer to the “Changing I2C address of BNO055" section for more details.
- The user can interface the BNO055 with Standard/Fast I2C interface only.

Changing I2C address of BNO055 IMU

- The user has the option of changing the I2C address of the BNO055 IMU from the default “0X29” I2C address. The 2 resistors which determine the I2C address of the BNO055 are labeled as “ADR" on the SmartWheels EV2. One of the resistors will be unpopulated. The user needs to solder the unpopulated resistor pads in order to effectively change the I2C address of the BNO055 to “0X28.”

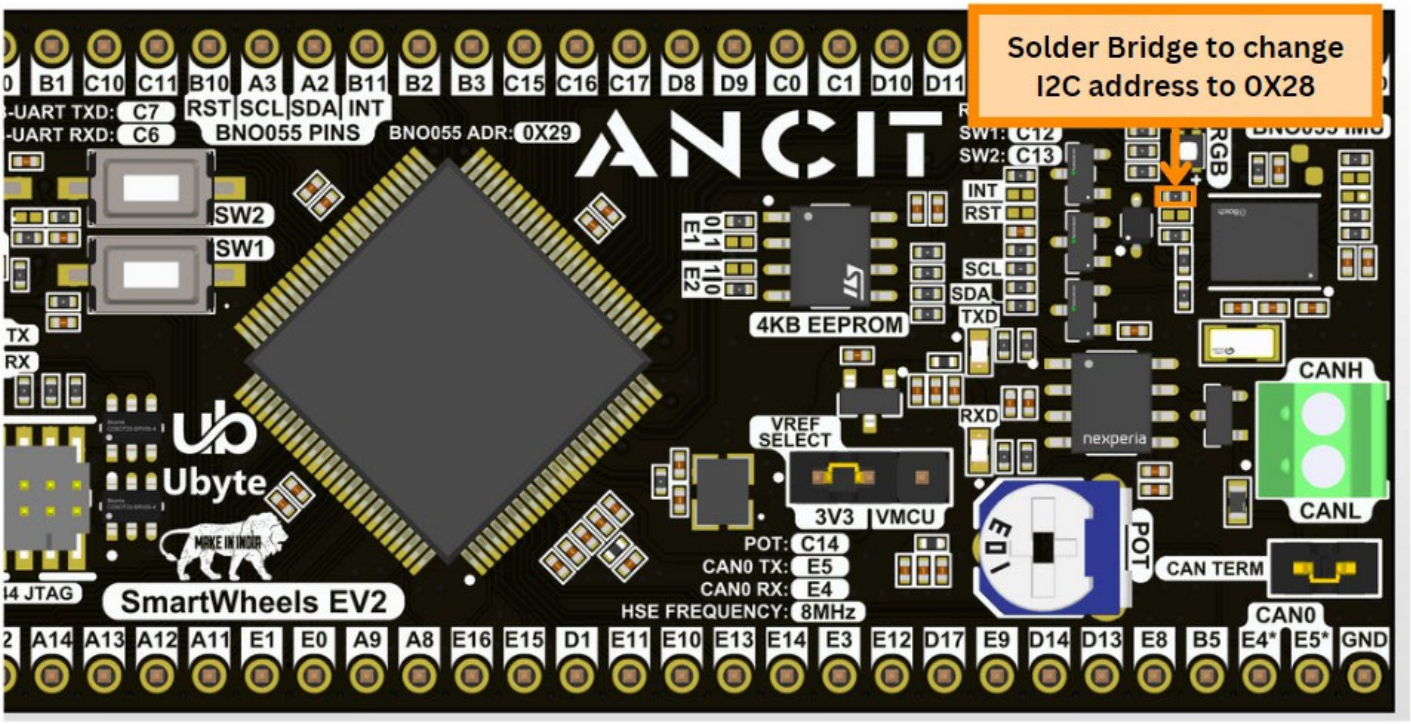


Fig 16 : I2C Address Selection Jumper Resistors

- The process of changing the I2C address requires precise soldering skills therefore the users who are not experienced with soldering should not attempt to change the I2C address on the SmartWheels EV2. The user may accidentally solder bridge the pads to other surrounding components leading to a short circuit. This will damage the SmartWheels EV2.
- The user can remove the solder bridge to restore the I2C address of the BNO055 to “0X29.”

Isolating BNO055 IMU from MCU

- The user can isolate the BNO055 IMU from the S32K144 microcontroller entirely. This can be done if the user doesn’t want any other peripherals to interfere with the I2C pins of the SmartWheels EV2 microcontroller exposed on the IO headers.
- GPIO pin PTB10 is used to send the Reset signal to the BNO055 IMU and GPIO pin PTB11 is used as an input for interrupt signal from BNO055 IMU. Both these pins of BNO055 are isolated from the S32K144 microcontroller so that the user can use these MCU GPIOs for other applications with external peripherals. However if the user wishes to interface the Reset and interrupt pins of the BNO055 IMU, they need to solder bridge the jumper resistor pads as shown in the Fig 17. above. This will connect the Reset and Interrupt signals from the BNO055 to PTB10 and PTB11 pins of the microcontroller respectively. The user must ensure that the Reset pin of the BNO055 is driver high for normal operation of BNO055 IMU.
- The user can also isolate the BNO055 IMU from S32K144 MCU by removing the jumper resistors as shown in Fig 17. Removing these jumper resistors will effectively disconnect the BNO055 from the S32K144 microcontroller therefore the user can use the PTB10, PTB11, PTA2 and PTA3 without any interference from the onboard BNO055 IMU. The jumper resistors that are used for isolating the BNO055 from S32K144 are also labeled as “INT”, “RST”, “SDA” and “SCL” so the user is able to identify the jumper resistors effectively.
- Please note that desoldering and soldering the jumper resistors require precision soldering skills. Therefore the user must be skilled in soldering to solder and desolder jumper resistors from SmartWheels EV2. The user may damage the SmartWheels EV2 during the process.

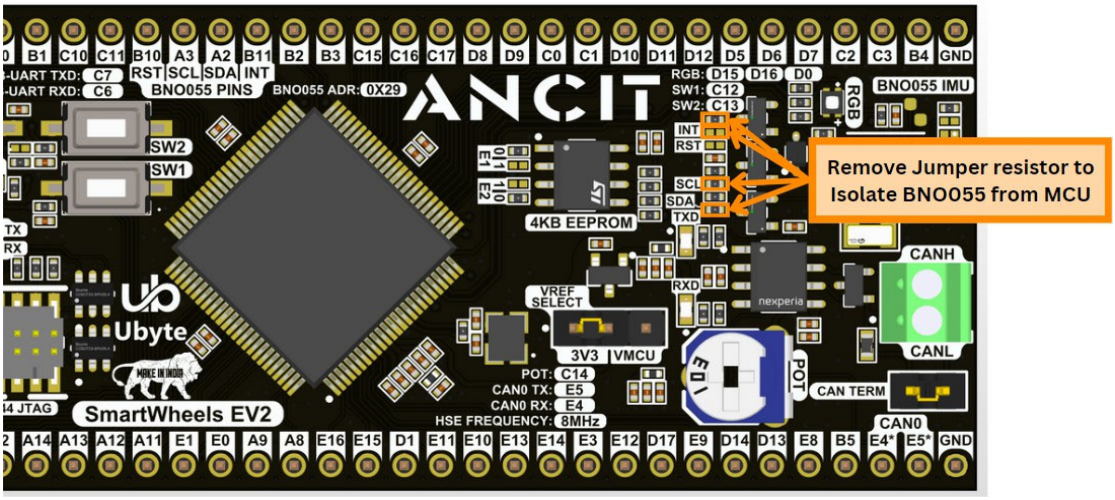


Fig 17 : I2C Address Selection Jumper Resistors

Demo examples for interfacing the BNO055 IMU are available on request. Please contact the ANCIT sales team for more information.

12. Specifications of M24C04 4KB EEPROM

- Memory Size: 4Kbit
- Write Cycle Endurance: 1,200,000 times
- Interface: I2C
- Connected to LPI2C0. SDA: PTA2 ; SCL: PTA3
- I2C read address: 0xA1 and I2C write address: 0xA0

Other Specification can be referred to from the datasheet of the IC: m24c04-r.pdf

The M24C04 EEPROM shared the I2C bus with the Bosch BNO055 IMU sensor. The same I2C pins are also available to the user through the pin headers.

E0 and E1 Pins can be pulled high or low by I2C Address Selection Jumper therefore changing the Read and Write address of the EEPROM. The user can desolder the placed jumpers “0” and make a solder bridge at “1” for either of E0 and E1 pins.

Demo Examples for using theM24C04 EEPROM are available. Please contact the sales team for more information.