NCN5140S Evaluation Board Manual

NCN5140TSCGEVB, NCN5140BSCGEVB

INTRODUCTION

The NCN5140S System−In−Package (SIP) is a highly integrated KNX® solution, requiring a minimal external BoM. The NCN5140TSCGEVB and NCN5140BSCGEVB are two certified hardware designs using the NCN5140S. These offer a fully featured 8−button switch application with RGB LEDs. Either touch or tactile buttons are used.

Both designs went through CE certification and are fully compliant with the KNX standard.

A binary containing the appropriate application software is downloadable from the ON Semiconductor[1] website. After programming the binary, a fully compliant and pre−certified KNX−application is achieved.

BOARDS OVERVIEW

There are two fully certified evaluation boards available for the NCN5140S. The NCN5140BSCGEVB contains 8 tactile buttons and the NCN5140TSCGEVB contains 8 capacitive touch buttons. Both boards have 8 RGB LEDs, one per button. The color and brightness of every individual LED can be easily configured by the installer through ETS. Every LED can be assigned a color/brightness depending on the function/state of the associated button.

Table 1. ALL CHANGES ALLOWED TO BE MADE TO THE HARDWARE DESIGNS

Allowed Changes
Adding a common−mode choke
Adding a TVS−diode
Change the amount of buttons (between 1 to 8)
Change the location of the buttons/LEDs
Change the type of RGB−LEDs used
Select a different buffer capacitor
Adapt the Fan−In resistor

ON Semiconductor®

www.onsemi.com

EVAL BOARD USER'S MANUAL

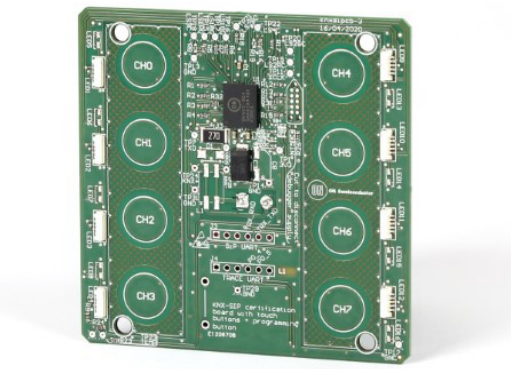

Figure 1. NCN5140TSCGEVB Switch Application Board Touch Version

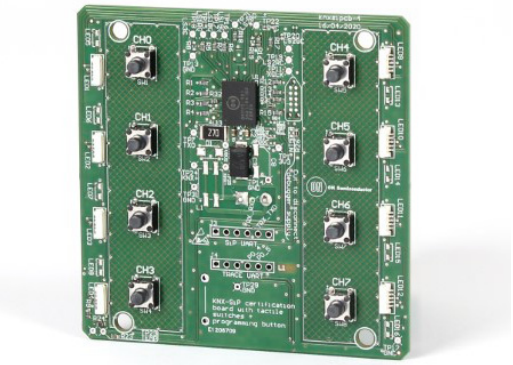

Figure 2. NCN5140BSCGEVB Switch Application Board Tactile Version

1. https://www.onsemi.com/ncn5140S

Design Customization

The hardware designs can be adjusted to fit many different application needs. The allowed hardware changes are limited in order to stay compatible with the delivered software and to stay compliant with the KNX specification. Table 1 lists everything that is allowed to change.

These changes allow to build custom KNX switch applications with anything between 1–8 buttons. The look and feel of the switch can be fully customized to your specific needs.

The functionality of the boards cannot be changed.

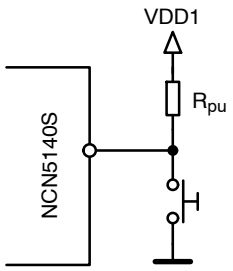

Figure 3. Pull−up Resistors Used on the Tactile Switches

The boards are always delivered with the appropriate software in binary form. This software binary does not have any degree of freedom.

An appropriate ETS database is downloadable from the ON Semiconductor website. This database file has to be customized with the manufacturer logo, custom product descriptions, ...

For the ETS database it is allowed to remove group objects to decrease the functionality of the device.

Software Configuration

As the software binary is fixed, device configuration during production is done over the KNX bus. This is necessary to program individualization parameters such as serial number, amount of channels used, ... For more information on device programming refer to the section *Device Configuration*.

FUNCTIONAL DESCRIPTION

Tactile Buttons

The NCN5140BSCGEVB contains 8 tactile buttons. The PCB−layout for tactile buttons is fairly simple. The state of the button is determined by a high or low signal on the GPIO−pin. The signals are debounced in software. All the pins going to a tactile switch, are pulled high using a 100 kΩ resistor. Refer to appendix A for the complete net list.

The layout of the buttons is allowed to change. It is recommended to keep the traces going to the buttons away from the LED traces. This increases the noise immunity of the whole circuit.

Capacitive Touch Buttons

On the NCN5140TSCGEVB 8 capacitive touch buttons are used. For capacitive touch buttons the layout is critical for proper operation. A good layout is required in order to achieve good button sensitivity and Signal−to−Noise Ratio (SNR). The most important layout guidelines are listed in the datasheet of the NCN5140S. For a complete guide on capacitive sensing applications refer to Cypress™ AN85951[2]. Strictly follow these guidelines to achieve reliable functionality!

Number of Buttons

It is allowed to change the number of buttons to anything between 1 and 8. The selected number has to be programmed in the software through the KNX−bus. Refer to section *Download Configuration* for more information.

When reducing the amount of buttons on the design, always remove the channels with the highest number. Otherwise the ETS database will no longer be compatible with the hardware design.

E.g. when going for a 6 button application, remove the buttons from channel 7 and 8.

RGB LEDs

For every button on the board, there is a corresponding RGB−LED. The amount of LEDs driven by the software, is scaled according to the number of buttons programmed in the software. As the microcontroller pin−out is fixed in software, all the RGB LEDs are tied to a specific button. The schematic clearly marks which button and LED belong to which channel. Always keep these together in the layout.

When removing a channel, remove the button and its corresponding LED. Removing an arbitrary LED will make the software incompatible.

In ETS the individual LEDs can be programmed to display any of the following colors: White, Red, Green, Blue, Yellow, Cyan, Magenta and Orange. The brightness of every individual LED can be set to a value between 0 % to 100 %.

Resistor Sizing

The sizing of the series resistors for the LEDs is very important. First of all it determines the current the LED will draw at maximum brightness. Choosing a higher resistor value results in a lower maximum current but also in a lower maximum brightness.

Red, green and blue LEDs have a different light output efficiency. To achieve the same brightness, the current through a blue LED must be much higher than through a red LED.

To get correct color mixing, selecting the right resistor value for the different colors is very important. When the three colors are at full brightness, the observed color must be white. Incorrect resistor values lead to one color being brighter shifting the full brightness color away from white.

2. https://www.cypress.com/file/46081/download

First the correct forward current must be selected for every color. This is done using the Luminous Intensity – Forward Current graphs in the datasheet of the LED. Select a maximum brightness and determine for every color which forward current is needed to reach it.

Make sure the total current I_{ft} out of the I/O connected to the common anodes of the RGB LEDs never exceeds 20 mA. Higher currents might damage the microcontroller GPIO pins.

Now determine the forward voltage for every color based on the forward current. With these two figures the series resistor value can be calculated:

$$R = \frac{V_{DD1} - V_f - V_{OH} - V_{OL}}{I_f}$$

$$R = \frac{3.3\ V - V_f - 0.6\ V - 0.6\ V}{I_f} \quad \text{(eq. 1)}$$

The parameters V_{OH} and V_{OL} are the GPIO output high and low levels (see figure 4). These levels are specified in the Cypress PSoC 4: PSoC 4100S Plus Datasheet[3].

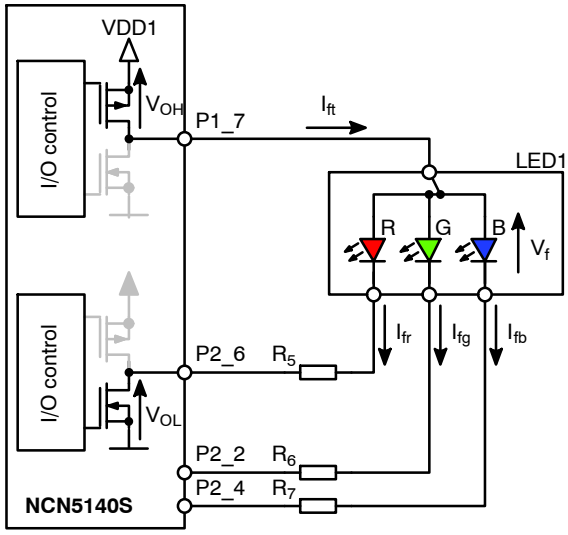

Figure 4. Voltage Drop Across the GPIO Switching Transistors

Driving the RGB LEDs

Driving 8 RGB LEDs individually is not possible as this would require 8 x 4 = 32 GPIOs. As the NCN5140S only has 23 GPIOs, a time multiplexing scheme is used to drive the LEDs. Time multiplexing the LEDs has the additional advantage of lowering the current consumption. Driving 8 LEDs at once at 20 mA each results in a total consumption of 160 mA. Although this is within the allowed 200 mA total GPIO current, it exceeds the current limit of 100 mA on V_{DD1}.

To control all 8 LEDs at once, they are connected in two strings of 4 using the principle shown in figure 5. All the common anodes of the RGB LEDs are routed to an individual GPIO pin. This GPIO directly drives the anode and delivers the required LED current. Every four LEDs, the cathodes are connected together per color. Every color goes to an individual IO with a series resistor. This results in (2 x 3 cathodes) + (2 x 4 anodes) = 14 GPIO pins being used.

To drive the LEDs, the individual anodes are pulled high one at the time. For a fixed period of time every LED will light up with a certain brightness. During this period all the other LEDs are off. The human eye is too slow to see the LEDs turning on and off if it is done fast enough. Switching an LED on and off with a frequency of at least 100 Hz will avoid any visible flickering.

During the time that the anode is high, the cathodes are switched on and off using PWM. The cathode of every color is driven individually using PWM making it possible to generate a wide range of colors. The duty−cycle value of the PWM signals are adjusted for every LED when its common anode is driven high. This will generate a different color/brightness for every LED individually. Figure 5 shows how this is done for two RGB LEDs.

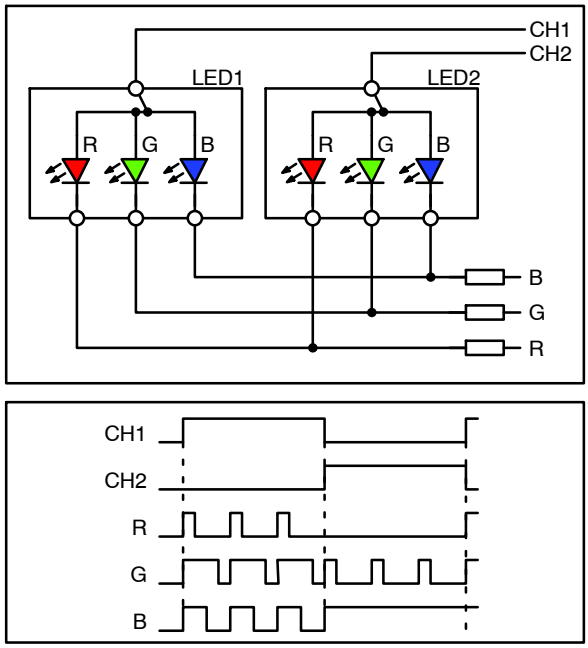

Figure 5. LED Time Multiplexing Control Scheme

Pin−Out

The microcontroller has a limited number of available pins. All the 24 available GPIO pins are used by the application. When the design has less than 8 buttons and RGB LEDs, some GPIO pins are left floating.

It is important that all the buttons and LEDs are routed to the right GPIO pin. The software determines which pin is used for which purpose. Failing to correctly route all the peripherals will make the hardware incompatible with the delivered software.

Table 4, appendix A shows the net list for both designs. Follow this net list exactly while making your design.

3. https://www.cypress.com/file/396611/download

Board Configuration

Some pins coming out of the NCN5140S are used for configuring the transceiver. Use the configuration from table 2 in order to make certain the board is compatible with the delivered software.

Table 2. SETTING OF THE CONFIGURATION PINS

TREQ	MODE	UC1	UC2	Mode Description
0	*0*	*0*	*0*	*9−bit UART−Mode, 19 200 bps*

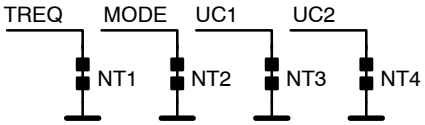

Figure 6. Net ties on the Evaluation Boards Put the KNX Chip in a Default Configuration

On the evaluation boards, net ties pull the pins to ground as shown in figure 6.

KNX−Bus Connection

The two boards have a KNX−connector on the backside. This connector mates with the 243–211 terminal block from Wago.

This is the standard connector recommended by the KNX organisation and used on almost all KNX devices.

Common Mode Choke

If desired, a common mode choke such as the Murata 50475C can be mounted on the footprint of L1. This helps to block common mode disturbances. The provided PCB designs do not require this choke as they pass all CE related testing without it.

Before soldering the choke on the foreseen footprint of L1, the tracks shorting the component must be cut.

Figure 7. The 243−211 Wago Terminal Block

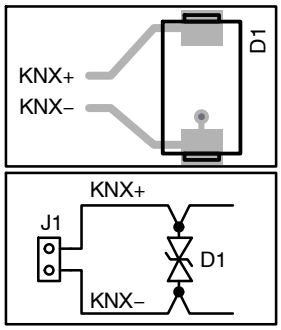

Figure 8. Optimal TVS Layout

TVS−Diode

Internally the NCN5140S has a TVS−diode at its input terminals. By default this TVS−diode is sufficient to pass all transient related tests required by EN 50491 and the KNX standard [3]. If it is desired to reach a higher transient level and the internal TVS diode seems insufficient, it is possible to mount an additional TVS on the PCB.

This external TVS diode must be bidirectional as there is no reverse polarity protection diode on the PCB.

It is important to optimize the layout of the TVS diode in order to effectively protect the transceiver against surges. When a surge occurs, the TVS diode must clamp the voltage to a level which doesn't damage the transceiver. To achieve this, the series resistance should be as low as possible.

To do this, connect KNX− directly to the TVS diode before routing it to the rest of the PCB ground as shown in figure 8. This ensures that the current path to the TVS has the lowest resistance and no transient current will flow to the transceiver.

Programming Mode

Every device in a KNX−network has an individual address used for programming the device. This address is assigned by putting the device in programming mode and downloading it through ETS.

Most KNX devices make use of a dedicated programming button to enter programming mode.

This button is usually located at the back of the device and is accompanied by a programming LED. This LED indicates if the device is in programming mode or not.

These evaluation boards also contain a programming button at the back. The programming LED was omitted. Instead the LEDs on the front of the device start blinking when it is in programming mode.

Aside from the programming button there is also a custom implementation to put the device in programming mode. By pressing the button of channel 1 and any other channel simultaneously for at least 5 seconds, the device will enter programming mode. This allows the installer to change the individual address even when the device is installed on the wall. Once installed the backside of the device is inaccessible.

A programming button on the back of the device is usually more convenient for installers as it is standard on all KNX devices.

When omitting the programming button on the back of the device, leave the ANAO/PROG pin floating.

Serial Wire Debugging (SWD)

The Cortex®−M0+ microcontroller has an SWD−interface which is used for programming and debugging. To connect to the target, only a data and clock line are needed (SWDCLK and SWDIO). This low signal count allows for the use of small debug connectors.

Most manufacturers only use a couple of copper pads on the PCB to connect the debugger. These pads are used in combination with POGO pins during production. This saves the cost of an expensive connector. The evaluation board designs contain a footprint for a similar system. The debug footprints on the PCBs interface with a Tag−Connect cable. This is a cable containing spring loaded contacts which push against the pads on the PCB.

The footprint on the PCB is very small so the overall design can be reduced in size. At the other end of the Tag−Connect cable, there can be a number of different connectors. For more information on all the options visit the Tag Connect[4] website.

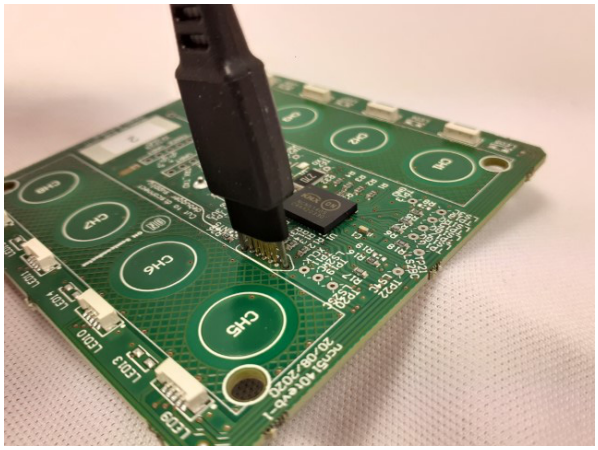

Figure 9. The Tag−Connector Used for SWD Programming

The standard pin−out as specified[5] by Keil is used. Normally pin 7 is not used on the 10−pin version. On these boards pin 7 is being utilized to externally supply the microcontroller. This allows programming the board without connecting it to the KNX bus.

The supply voltage on pin 7 must never exceed 3.3 V as this would damage the hardware! When using an adapter board that utilizes another voltage on pin 7, cut NT5 to avoid any damage to the board. The board should in that case be supplied from the KNX bus or by forcing 3.3 V on TP1.

The standard SWD−interface requires some pull−up and pull−down resistors on some of the pins in order to function correctly. These are not present on the evaluation board and must be placed on the interface board between the programmer and the Tag−Connect cable. Figure 10 shows to which pins these should be connected. The recommended value for R_2 and R_3 is 4.7 kΩ.

Also some additional reset circuitry might be required to reset the device during programming. See section *Reset Circuitry*.

Reset Circuitry

The NCN5140S has two reset pins, RSTBIN and RSTBOUT. The RSTBOUT−pin comes from the NCN5130 and is used to keep the microcontroller in a reset−state during startup. In order to do this, the RSTBOUT must be connected to the reset−pin of the microcontroller named RSTBIN.

The RSTBOUT−pin on the NCN5130 is an open drain output. As the microcontroller has an internal pull−up resistor on its RSTBIN−pin, RSTBOUT and RSTBIN could be connected together directly. However in some situations this might prevent a programmer from being able to program the microcontroller.

When 3.3 V is being forced onto V_{DD1} instead of connecting the bus voltage to the board, the NCN5130 will pull its RSTBOUT−signal low. If in this situation RSTBOUT and RSTBIN are directly connected together, it is no longer possible to drive the RSTBIN−signal high. This means the microcontroller is permanently in reset and cannot be programmed. To solve this issue, a resistor of 2.2 kΩ (R_{32}) must be added between RSTBOUT and RSTBIN.

Now a push−pull driver will be able to directly drive the RSTBIN−signal and get the microcontroller out of reset. Most debuggers have an open−drain output to drive the microcontroller reset−signal and will not be able to overrule the open−drain of the NCN5130 directly. This can be solved by adding a buffer with a push−pull output as shown in figure 10.

Figure 11 shows how the resistor R_{32} prevents that the buffer destroys the open−drain output of the NCN5130 by limiting the current flowing into the pin. R_{32} is mounted on the evaluation boards by default. Refer to the schematics in annex C and D.

4. https://www.tag−connect.com/
5. http://www.keil.com/support/man/docs/ulink2/ulink2_hw_connectors.htm

FAN−IN SELECTION

Every KNX application consumes a certain amount of current from the bus which is specified in its datasheet. This current consumption is classified according to the fan−in−model [2]. The NCN5130 transceiver will make certain the application stays within the KNX specification. To adapt the transceivers limits to the right fan−in, the FANIN pin is used. For more information on the KNX fan−in−model refer to AND90055/D[6].

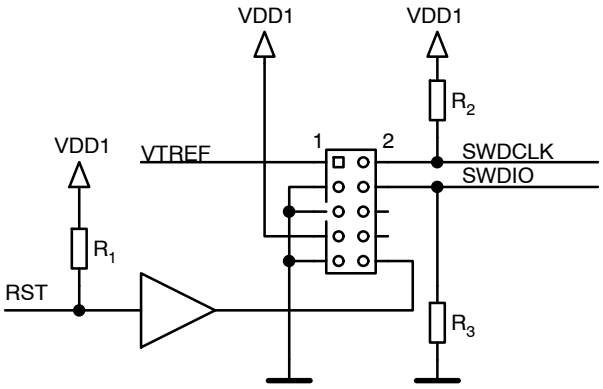

Figure 10. Debug Circuitry Necessary to Program the Device when V_{DD1} is Forced Externally

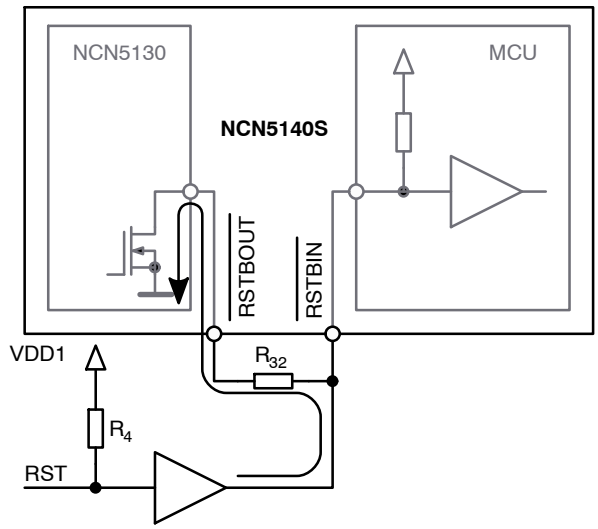

Figure 11. The Resistor R_{32} Limits the Current Flowing in the RSTBOUT−output

A regular KNX switch application has rather low power demands. The LEDs used in the evaluation board designs, consume a maximum current of 15 mA peak. As the LEDs are time multiplexed in two strings, the maximum current drawn at any given moment is 30 mA. The microcontroller consumes a maximum current of 10 mA. This gives a total maximum current consumption of 40 mA at the application side. This current comes from DC−DC1 which delivers 3.3 V.

At the transceiver side, the NCN5130 has a self consumption of 4.4 mA worst case. This current is drawn immediately from the bus. The following formula can be used to calculate the total bus power consumption:

$$I_{bus} = I_{transreceiver} + \left(\frac{I_{application}}{\eta} \times \frac{V_{DD1}}{V_{bus}}\right)$$

$$I_{bus} = 4.4\ mA + \left(\frac{40\ mA}{0.7115} \times \frac{3.3\ mV}{20\ V}\right) = 13.68\ mA \qquad \text{(eq. 2)}$$

The efficiency of the DC−DC converter was calculated using the *KNX Family Efficiency Calculator*[7]. Calculating the bus current consumption using this calculator yields the same result.

The maximum allowed bus current for a fan−in of 10 mA is 12 mA. This is too low for the required 14 mA. The next available fan−in is 12.5 mA and the allowed bus current is scaled accordingly to 15 mA.

This is suitable for this application. Calculating the resulting FANIN resistor can be done using the following formula:

$$R_{FANIN} = \frac{434}{I_{coupler\_lim} - 4 \times 10^{-4}}$$

$$R_{FANIN} = \frac{434}{15\ mA - 4 \times 10^{-4}} = 29726\ \Omega \qquad \text{(eq. 3)}$$

This results in an E24 resistor of 30 kΩ. The next higher resistor value must be selected as the bus coupler current limitation will otherwise be above the allowed limit for this Fan−In. This value gives an actual bus coupler current limitation of:

$$I_{coupler\_lim} = 4 \times 10^{-4} + \frac{434}{R_{FANIN}}$$

$$I_{coupler\_lim} = 4 \times 10^{-4} + \frac{434}{30\ k\Omega} = 14.87\ mA \qquad \text{(eq. 4)}$$

This is still higher than the required 13.68 mA. If the value would be lower, then a more precise resistor value closer to the calculated value must be used. If that is not possible, use the next higher Fan−In value.

Table 3. SELECTION OF THE FAN−IN RESISTOR BASED ON THE AMOUNT OF LEDS USED

# LEDs	I_{LED} (mA)	$I_{bus,max}$ (mA)	Fan−In (mA)	Fan−In Resistor E24 (kΩ)
8	30	13.68	12.5	30
7	26.25	12.66	12.5	30
6	22.5	11.69	12.5	30
5	18.75	10.74	10	39
4	15	9.83	10	39
3	11.25	8.95	7.5	51
2	7.5	8.10	7.5	51
1	3.75	7.27	7.5	51

6. https://www.onsemi.com/pub/Collateral/AND90055−D.PDF
7. https://www.onsemi.com/pub/Collateral/KNX%20FAMILY%20EFFICIENCY%20CALCULATOR.XLSM

For applications which use less buttons, the maximum LED current will be lower. Use table 3 to select the right fan−in resistor for your application.

When using different LEDs than the ones used on the EVBs, repeat the above calculations either manually or using the calculator[8] to determine the right fan−in for your application.

Vfilt CAPACITOR

A key advantage of the NCN5140S is the possibility to design small form factor applications. To achieve a small design footprint the size of the Vfilt capacitor must be chosen accordingly.

Physical Size

There are different types of capacitors on the market which have their different electrical and mechanical properties. A first very popular type is the aluminium electrolytic capacitor. These capacitors have a large capacitance and high voltage rating for a low price. They are packaged in a metal can but are quite tall (≥ 5.9 mm). This prevents them from being used in low profile applications.

An alternative for the classic electrolytic capacitor is the polymer electrolytic capacitor. Instead of a liquid it uses a solid polymer as electrolyte. This increases the lifetime of the capacitor and lowers its ESR. Generally these also come in aluminium cans making them less suitable for low profile applications. There are low profile versions of these capacitors available, but at the time of writing not yet with a voltage rating of 35 V or above.

Ceramic capacitors are widely used as they are small and robust. Larger capacitors in the μF range are harder to find and are more expensive than their electrolytic counterpart. The main disadvantage is that at large DC−bias their remaining capacitance is low. These capacitors are low profile but not desirable due to the DC−bias effect and increased cost.

Tantalum capacitors are a good alternative for aluminium electrolytic capacitors when size matters. They are smaller than electrolytic capacitors for the same capacitance/voltage rating. Instead of aluminium cans they use low profile plastic casings. Which makes them a good choice for low profile applications. The disadvantage is that they are more sensitive to voltage spikes and reverse biasing. High voltage spikes can lead to catastrophic failure of the capacitor.

The two hardware designs have a footprint to mount an electrolytic capacitor and a second footprint to mount a tantalum capacitor (figure 12). The footprint size for the tantalum capacitor is 7343 metric.

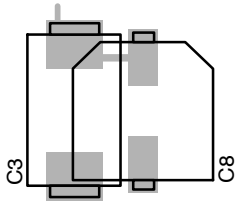

Figure 12. Option to Place an Electrolytic or Tantalum Capacitor

Dimensioning

For proper operation in the application the buffer capacitor must be dimensioned correctly. More information about how to dimension the capacitor can be found in AND90055/D[9].

Section *Fan−In Selection* calculates what the maximum current drawn from the bus in the worst case scenario. For this application this value is 13.68 mA.

First the buffer capacitor must be able to smooth load steps. The maximum load step that could occur at the application side is two LEDs switching on simultaneously. This corresponds with a load step of 30 mA at the application side which translates in a current−step of 6.95 mA at the bus side.

$$I_{bus} = \frac{30\ mA}{0.7115} \times \frac{3.3\ V}{20\ V} = 6.95\ mA \qquad \text{(eq. 5)}$$

To handle this load step, the buffer capacitor should be larger than 5.16 μF.

$$C > \frac{\Delta I^2_{step}}{2 \times (V_{BUS1} - V_{coupler\_drop} - V_{FILTL}) \times \frac{\Delta I_{coupler}}{\Delta t}}$$

$$C > \frac{6.95\ mA^2}{2 \times (20\ V - 2.8\ V - 9.4\ V) \times 0.6\ A/s} = 5.16\ \mu F \qquad \text{(eq. 6)}$$

Secondly it is important that the buffer capacitor is large enough allowing the system to store critical data prior to complete shutdown. The time between SAVEB and RESETB falling called $t_{warning}$ (figure 13) is the time in which the microcontroller has to write all the necessary data to non−volatile memory. SAVEB goes low 2 ms ($t_{busfilter}$) after V_{BUS} dropped below 20 V, while RESETB goes low when V_{filt} goes below 12 V.

The time necessary to program one row of flash[10] is 4 ms. Adding a margin of 2 ms gives a total $t_{warning}$ of 6 ms.

$$C > I_{system} \times \frac{t_{warning} + t_{busfilter}}{V_{BUS1} - V_{copuler\_drop} - V_{FILT}}$$

$$C > 13.68\ mA \frac{6\ ms + 2\ ms}{20\ V - 2.8\ V - 9.4\ V} = 14.02\ \mu F \qquad \text{(eq. 7)}$$

8. https://www.onsemi.com/pub/Collateral/KNX%20FAMILY%20EFFICIENCY%20CALCULATOR.XLSM
9. https://www.onsemi.com/pub/Collateral/AND90055−D.PDF
10. one row = 256 bytes

Taking some margin on the capacitor value, results in a minimum value of 22 μF. This results in an actual $t_{warning}$ of 10.55 ms.

$$t < \frac{C \times (V_{BUS1} - V_{coupler\_drop} - V_{FILT})}{I_{system}} - t_{busfilter}$$

$$t < \frac{22\ \mu F \times (20\ V - 2.8\ V - 9.4\ V)}{13.68\ mA} - 2\ ms = 10.55\ ms \quad \text{(eq. 8)}$$

Applications which need to write away more data on shutdown, should use a larger buffer capacitor. The minimum capacitor value of 22 μF is recommended for all NCN5140xSCGEVB applications regardless of the amount of LEDs used. If your actual application consumes more than what is calculated here, repeat these calculations.

SOFTWARE

The software binary provided with the boards is developed and certified by Weinzierl Engineering Gmbh[11]. When deriving a certificate, the certified software must stay unchanged. Only the amount of parameters/group objects can be less. If this is the case no additional testing is required.

For the ETS database only commercial details can be changed (Logo, coloring, ...).

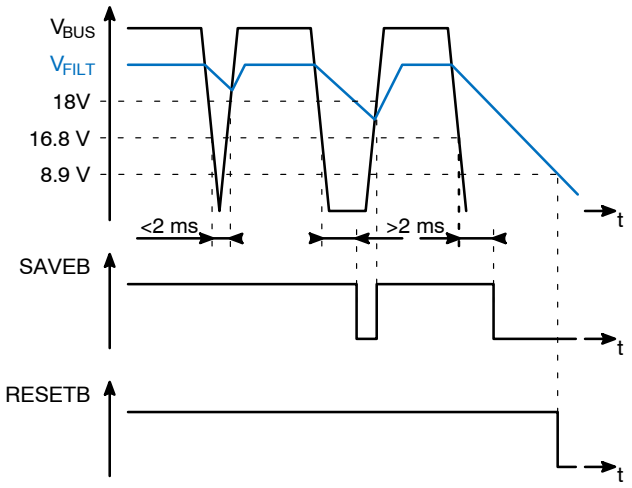

Figure 13. Behaviour of the SAVEB− and RESETB−pin on Shutdown

The software is downloadable from the ON Semiconductor website as a binary. The product entry exported through the Manufacturer Tool from the KNX Associaton, is provided together with the binary.

Flashing the Software

The NCN5140S is delivered with an empty flash memory. The binary provided with the device must be flashed during the production of the application. Section *Serial Wire Debugging (SWD)* discusses the Tag−Connector which can be used to program the device without an additional on−board connector. In production a similar but automated method is normally used.

To flash the device, the microcontroller must be powered up. There are two options listed below:

- *Forcing 3.3 V:* A first option is to force 3.3 V onto the board. On the evaluation boards pin 7 of the debug connector can be used. This way the microcontroller is directly powered from the external supply. The DC−DC converter of the NCN5130 will be disabled in this case. The NCN5130 will however also get powered and pull its RSTBOUT pin low. To still be able to program the microcontroller some additional circuitry is required as explained in section *Reset Circuitry*
- *Applying the bus voltage:* A second option is to apply the bus voltage (21 V to 30 V) to the KNX+/KNX− pins. In this case the open−drain RSTBOUT pin of the NCN5130 will be floating and the microcontroller can be programmed directly without any additional circuitry. This method requires less circuitry, but is slower. The startup time will be tens of ms in this case depending on the selected C_{filt} capacitor. The internal circuitry of the NCN5130 followed by DC−DC1 must be operational before the programming can start.

Programming the device is done through the SWD−interface. Pin 2 of the debug connector is the SWDIO pin and pin 4 the SWDCLK pin (figure 10). They both require an external pull−up/pull−down resistor at the debugger side as described in section *Serial Wire Debugging (SWD)*

Many programmers are available on the market to program ARM® Cortex−M0+ microcontrollers. When programming the device select Cypress CY8C4147AZI−Sxxx as the target microcontroller. It is not possible to use *PSoC Programmer* to flash the software. The tool only accepts special hex files which are created by the *PSoC Creator* IDE. Hex files created by the IDE, are also not compatible with other software packages. If it is desired to flash the device using a MiniProg or a KitProg from Cypress, then use *Cypress Programmer* instead.

DEVICE CONFIGURATION

Every KNX device contains several configuration parameters which have to be configured during production. Some of these properties have a fixed function defined by the KNX standard, such as the serial number. Others are device specific, such as the amount of buttons. All the different properties are explained here as well as how to configure them.

PID MANUFACTURER ID (PID = 12)

This is a 2 byte code assigned to every KNX manufacturer by the KNX Association. This code is unique and identifies who manufactured the device.

11. https://weinzierl.de/index.php/en/

PID SERIAL NUMBER (PID = 11)

Every KNX device that is manufactured **MUST** have a unique serial number. The KNX Association assigns a range of serial numbers to the manufacturer, which must then be flashed in every manufactured device.

PID ORDER INFO (PID = 15)

This property can contain manufacturer specific ordering information. It is a string with a maximum length of 10 bytes. Generally this is used to store a product name or an order number.

PID MANUFACTURER DATA (PID = 19)

The data type of this property isn't fixed. It is adapted to the data the manufacturer wishes to store in it. These 4 bytes can be used to store any manufacturer specific data. Very often this property is used to store the manufacturing date.

PID HARDWARE TYPE (PID = 78)

This property identifies a specific hardware version/type. It is used during an ETS download operation to check if the configuration is downloaded inside the right hardware. Downloading the configuration in incompatible hardware would otherwise lead to a non−functioning device.

The property is set to **0x00 01 0B 00 00 00** and must not be changed. Changing it renders the hardware incompatible with the ETS database.

Channel Count (PID = 251)

Sets the amount of channels used on this particular switch. A channel consists of both a button and an LED. This value must be between 1 and 8. Choosing a lower value than 8, removes the upper channels first. For example, when setting channel count to 7, channel 8 is no longer used. Always the lower channel numbers are used.

Button Type (PID = 252)

Select which type of hardware buttons are used. This can be either mechanical buttons **0x00** or capacitive touch buttons **0x01**. Selecting the wrong button type for your hardware, results in a malfunctioning device.

Button Config (PID = 253)

This property is only relevant when *Button type (PID = 252)* is set to 0x01. It is used to configure the parameters of the capacitive touch buttons. These parameters are used to tune the sensitivity of the touch buttons. It is necessary to adjust these parameters depending on the overlay material of the buttons and their specific layout. This property consists out of 10 bytes which are structured as follows:

- **Byte 1:** Mode
  Default value: 0x01
- **Byte 2:** Noise Threshold
  Default value: 40
- **Byte 3:** Negative Noise Threshold
  Default value: 40
- **Byte 4:** Hysteresis
  Default value: 100
- **Byte 5:** ON Debounce
  Default value: 3
- **Byte 6:** Low Baseline Reset
  Default value: 30
- **Byte 7:** Finger Threshold (High Byte)
  Default value: 0x01
- **Byte 8:** Finger Threshold (Low Byte)
  Default value: 0x90
  Total value: 0x190 = 400
- **Byte 9:** Finger Capacitance (High Byte)
  Default value: 0x01
- **Byte 10:** Finger Capacitance (Low Byte)
  Default value: 0x90
  Total value: 0x190 = 400

Above also the default values which are loaded into the evaluation boards are listed. What these parameters do and how to configure them is described below.

Theory of Operation

A capacitive touch button can be modelled as a capacitor in parallel with some parasitic capacitance coming from the traces going to the button. When a finger is brought close to the button, it's capacitance changes. It is these changes in capacitance which are detected and used to determine the button state.

To measure the button capacitance, there is a block that converts the capacitance into current. This current is then switched to an ADC using an analog multiplexer as illustrated in figure 14. This ADC will convert the current into a digital value which is then used to do the state detection. For more information on this topic refer to Cypress AN85951[12].

An algorithm in the microcontroller uses this raw digital value to determine if the button is pressed or not. It does this by comparing the measured value against a reference value, called the baseline. This baseline is variable because it is impossible to predict exactly what the button capacitance will be, making it impossible to compare it to a fixed value.

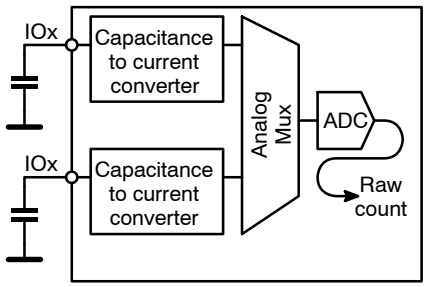

Figure 14. Capacitive Touch Sensing Method

12. https://www.cypress.com/file/46081/download

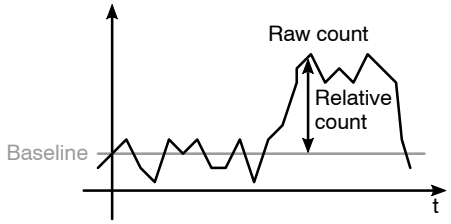

Figure 15. Raw Count Converted to Relative Count

The baseline is subtracted from the measured count value to obtain a relative value. This value is compared to the ***Finger Threshold*** to determine if the button is touched or not. If the measured relative value is above the threshold, the button is considered pressed, otherwise it is not.

A ***Hysteresis*** value is used around the Finger Threshold parameter to filter noisy transitions around the switching points. This Hysteresis is set above and below the Finger Threshold to avoid oscillations in the output signal.

The baseline value is constantly being updated in order to detect touches even in changing environments. For example, when dirt accumulates on the button, it's capacitance changes. If the baseline was fixed, this could be seen as the button being continuously pressed. In other scenarios it might lead to an unresponsive button. By updating the baseline, the static non−touched button capacitance is constantly being updated.

To avoid that the baseline is updated when touching the button, a ***Noise Threshold*** is used. This parameter determines above which threshold the baseline stops updating. This value is relative to the baseline. The value must be larger than the noise being measured by the ADC.

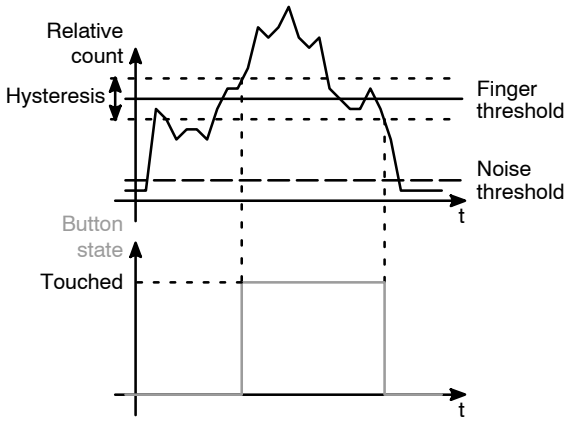

Figure 16. The Different Thresholds Used by the Touch Algorithm

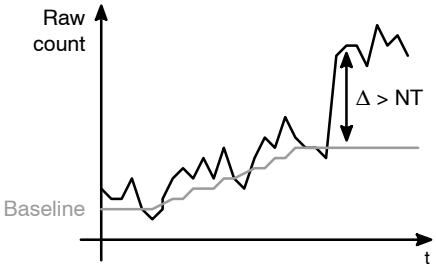

Figure 17. How the *Noise Threshold P*arameter is Used to Avoid Baseline Updates During Button Touches

Figure 17 shows how the baseline follows a slowly varying input signal and doesn't update when a large input signal variation occurs by pressing a button.

Negative spikes in the raw count signal can lower the baseline, making a button more sensitive. This can cause spurious unwanted button presses to occur. These negative spikes are observed when, for example, an ESD event occurs. In order to avoid the baseline from dropping in such a situation, the ***Negative Noise Threshold*** parameter is used.

Mode

It is possible to set all the different tuning parameters manually. To select manual tuning set this parameter to **0x01**. The only parameter not used in manual tuning mode is *Finger Capacitance*. All other parameters below must be set for a correct operation of the touch buttons.

A second option is to let an automatic tuning algorithm set the most optimal values for your application. Auto tune mode is selected by setting the *Mode* setting to **0x00**. The only *Button config* parameter used in automatic tuning mode is *Finger Capacitance*.

Noise Threshold

Above this threshold the baseline reference signals is never updated. This prevents the baseline signal from being updated while a button is being touched.

Negative Noise Threshold

The baseline reference is not updated below this point for a number of samples specified by the *Low Baseline Reset* parameter.

On Debounce

Selects the number of consecutive scans for which the sensor must be active before the buttons is marked as touched.

Low Baseline Reset

When the baseline is below the *Negative Noise Threshold* for *Low Baseline Reset* amount of samples, the baseline is updated. When the button is touched at startup, the baseline reference signal is set too high to detect touches. The *Low Baseline Reset* parameter resets the baseline after a configured amount of samples ensuring touches are detected again.

Finger Capacitance

This parameter is only used when *Mode* is set to **0x00** (auto tune). The auto tune algorithm will use this parameter to tune the buttons to a certain sensitivity. Setting this parameter, changes the sensitivity for all the enabled buttons.

The *fingerCap* sensitivity is given in femto Farad (fF). It expresses at what detected finger capacitance the button is triggered. The button sensitivity is inversely proportional to the Finger Capacitance value. This value can range from 100 fF to 1000 fF in steps of 20 fF.

Authorization

Not all properties in the device can be written freely. The properties are divided in different access levels which define who can access them.

- **Level 3:** Free access.
- **Level 2:** ETS parameter commissioning.
- **level 1:** Product manufacturer parameters.
- **level 0:** System manufacturer parameters.

The configuration properties defined here are located at level 1. These parameters can always be read out by anyone, but can only be written with the right authorization key. This authorization key is 4 bytes long.

After flashing the binary in the NCN5140S, this key will default to *12 34 56 78*. During production this key should be changed to a more secure value. This makes sure only the product manufacturer can alter these parameters.

ETS DATABASE

The matching ETS database for these boards can be downloaded from the ON Semiconductor[13] website. This database file can be loaded in ETS to start configuring the evaluation boards. For an actual end product, the network installer will use the database to configure the device.

When applying for a derived certificate, with a product based on the NCN5140S evaluation boards, a matching ETS database will have to be provided. This ETS database file has to match the derived product and also needs to contain manufacturer specific information.

Altering the database is done through the KNX Manufacturer tool. For more information on how to create a new OEM project in the manufacturer tool, refer to the KNX Association documentation[14].

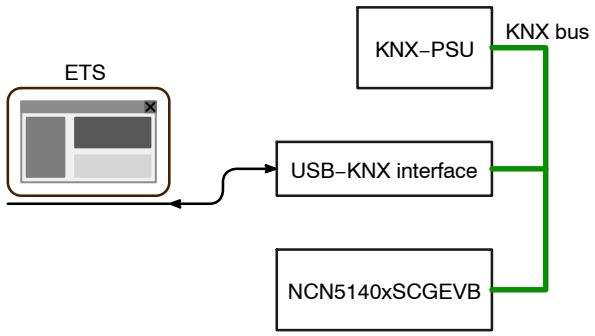

Figure 18. Minimal KNX Network to Test the Evaluation Board

GETTING STARTED

Setting up the NCN5140xSCGEVB boards for evaluation is only a matter of installing the appropriate software on the PC as the boards come fully pre−configured.

Hardware

The NCN5140xSCGEVB evaluation boards are pre−programmed with the certified binary. All the configuration parameters described in section *Device Configuration* are already loaded in the device. No additional programming or configuring of the device is necessary to get started.

Setting Up the Network

In order to test the device, a small KNX network has to be built. As a minimum a KNX power supply and a KNX−USB interface is needed. Other KNX devices such as a dimmer can be added to interact with the evaluation board.

A schematic representation of the minimal network setup is shown in figure 18.

Installing ETS

Configuring the devices in a KNX−network is done through the ETS software provided by the KNX Association. For more information on downloading and installing ETS[15] refer to the KNX Association website.

To get started with the NCN5140S evaluation boards, the free version of ETS is sufficient. It allows configuring up to 5 devices in a network.

13. https://www.onsemi.com/ncn5140S
14. https://support.knx.org/hc/en−us/articles/360000154519−Create−an−OEM−version−of−a−Product
15. https://www.knx.org/knx−en/for−professionals/software/ets−5−professional/index.php

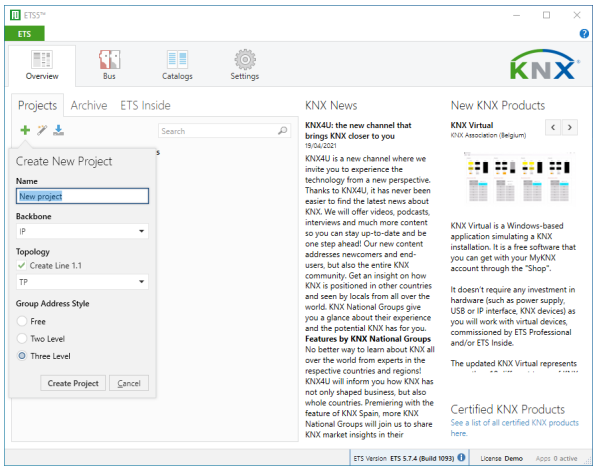

Figure 19. Create a New Project in ETS

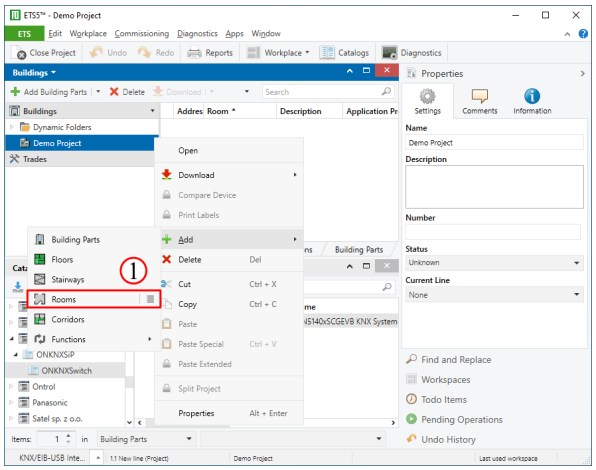

Figure 20. Add a Room to the Building in Which your Device is Installed

Configuring The Network

To configure the KNX network, a new ETS project has to be created. Launch ETS and create a new project by clicking on the plus icon shown in figure 19. Enter a name for your new project and leave the other settings at their default values as in figure 19.

After the project creation is complete, you will be greeted by an empty *Buildings* screen. This window is used by the network installer to create a plan of which devices are installed where in the building. This is needed to keep an overview of how everything is structured and to be able to easily adjust something later on.

It is not possible to add devices to the building itself. So in order to be able to add our evaluation board, add a room to the building. Do this by right clicking on the building in the *Buildings* view and selecting *Add* →*Rooms* ① as shown in figure 20. You will now be prompted to enter a room name. Enter an appropriate name and click *Ok*.

Once the room has been created, the evaluation board can be added to it. Adding a device is done through the ETS catalog. To open the catalog click on the *Catalogs* button ② in the top bar on the screen. In the new *Catalog* window all devices available in the online KNX catalog are displayed. In the list of manufacturers, search for ON Semiconductor. When unfolding the ON Semiconductor product list, the evaluation board can be found under *ONKNXSwitch*.

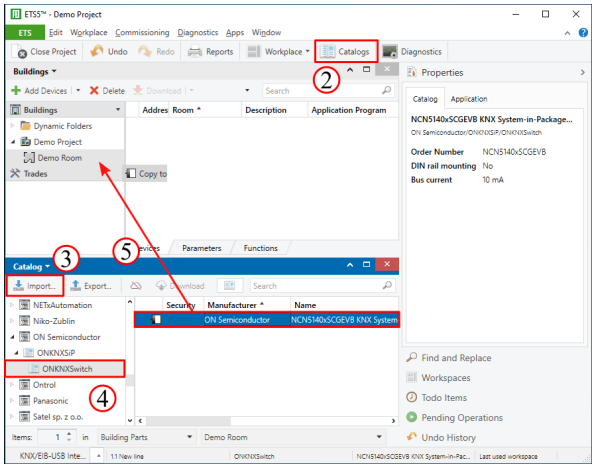

Figure 21. Add the Evaluation Board to the ETS Project

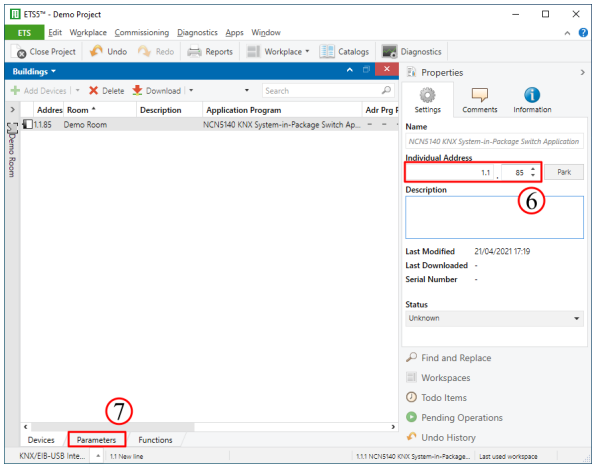

Figure 22. The Evaluation Board Added to the Room in ETS

If ON Semiconductor is not present in the manufacturer list for whatever reason, it is possible to import the product manually. Go to the ON Semiconductor[16] website and download the .knxprod file on the evaluation board product page. Import this file by clicking on the *Import* button ③ and navigating to the .knxprod file.

Once successfully navigated to the evaluation board as in figure 21 drag it ⑤ to the room created earlier.

Now that the device is added to a room, the individual address, parameters, etc... can be configured. First give an *Individual Address* ⑥ to the board. This address must be unique in your network. After doing this, switch to the *Parameters* tab ⑦ on the bottom of the screen.

In the *Parameters* tab it is possible to configure all the parameters available in the device. This goes from selecting the color/functionality of a button to configuring a specific timer function. For a complete list of all parameters and their associated group objects, refer to ON Semiconductor UM70043/D[17].

We will start by configuring the functionality of one button. Do this by first clicking on *Channel 1* ⑧. This opens the general settings for this channel.

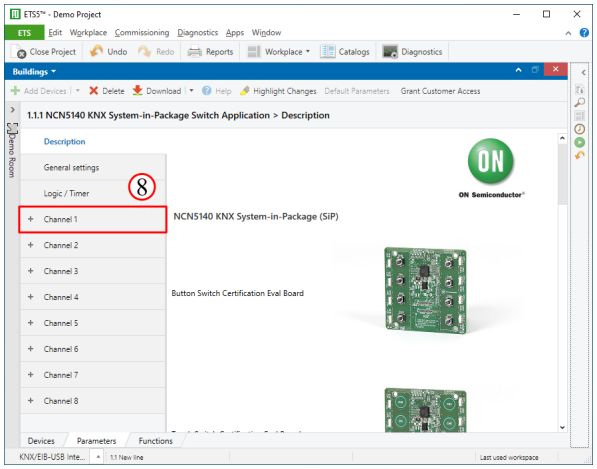

Figure 23. The Parameters Window for the NCN5140xSCGEVB Boards

In the general settings it is possible to first give a name to the channel ⑨. The name can for example be used to indicate what the button is used for, such as switching lights.

Every button can be assigned different functions. For demonstration purposes we will select the most simple one which is *Switching* ⑩. The LED associated with the button can be configured to indicate when the button is physically being touched/pressed through *Activity indication* ⑪. For more information on the other functions and options refer to ON Semiconductor UM70043/D[17].

In the *Switching* tab ⑫ the switching properties for the specified channel can be configured. Here we can specify to send out different commands based on different button events. A command can be transmitted on the KNX−bus based on a button press/release or a short/long button press. We choose to send out a command on a button press/release and select what command to send out from the drop down lists ⑬ and ⑭.

Next we will configure the behaviour of the LED ⑮. The LED was previously configured for *Activity indication*, so it will indicate if the button is pressed or not. In the LED configuration tab a default color can be selected for when the button isn't pressed ⑯. It is also possible to select that the LED must be off.

A different color can be set for when the button is pressed ⑰. The brightness value that can be entered is both for the default and the activity color.

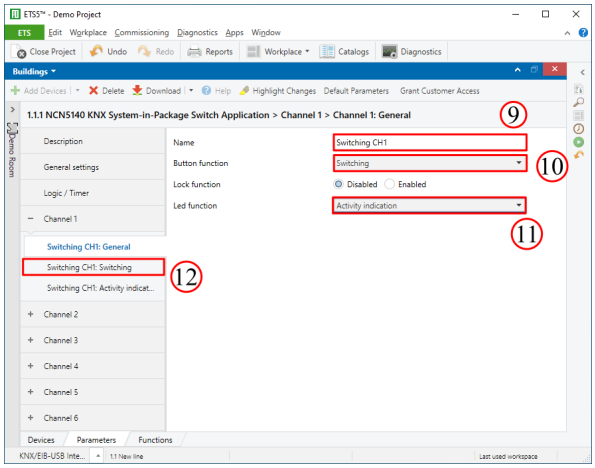

Figure 24. Configure Channel 1 to be Used for Switching

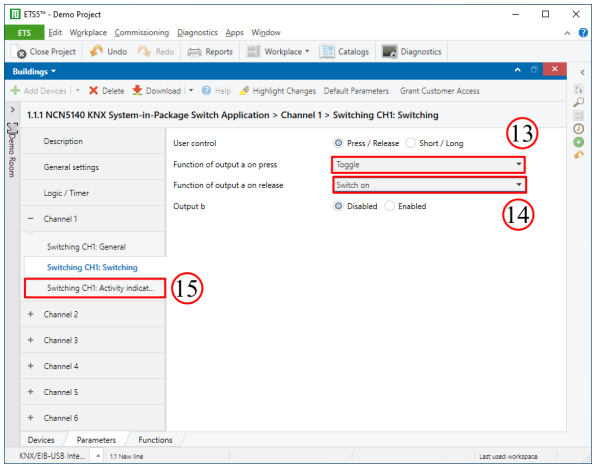

Figure 25. Options for the Switching Function

16. https://www.onsemi.com/ncn5140S
17. https://www.onsemi.com/pub/Collateral/UM70043−D.PDF

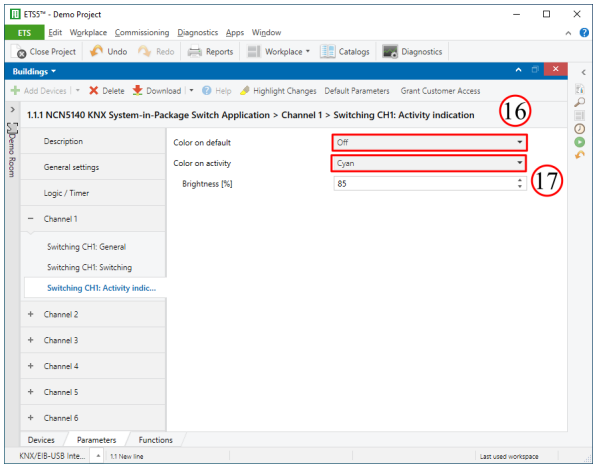

Figure 26. Configuring the Behaviour of the LED

The last part of the configuration is assigning group addresses. Every function in a KNX network is assigned a unique group address. Such functions are switching on the living room lights, dimming the bed room light, switching on/off the ventilation, ... In this configuration example we created a single function which is switching something on and off.

To assign group objects, click on the evaluation board in the *Buildings* view ⑱ and then on the *Group Objects* tab ⑲ on the bottom of the screen as shown in figure 27.

Now a list with all the group objects present in the device is shown. Note that the group objects shown are dependent on the configuration in the *Parameters* tab. Because we only configured one button for switching, only the switching group object for this button is shown.

To assign a new group address to the switching object, right click on it ⑳ and select *Link With*... ①a. This will open a new window as shown in figure 28.

In this window either an existing or a new group address can be assigned to the group object. As no group addresses have been created in this project yet, select *New* ②a. Create a new group address such as 2/0/0 and give it a name to indicate what it does.

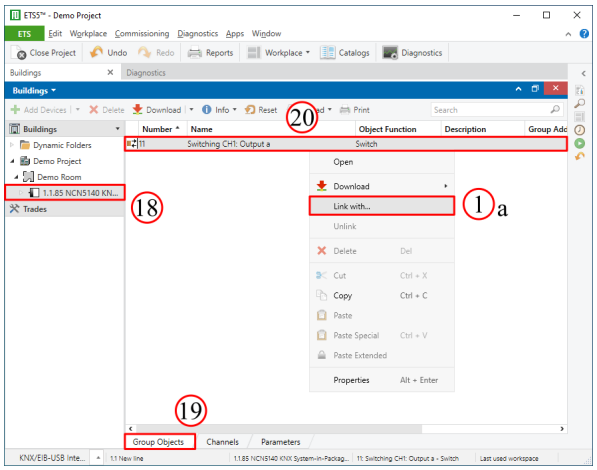

Figure 27. Assigning Group Objects

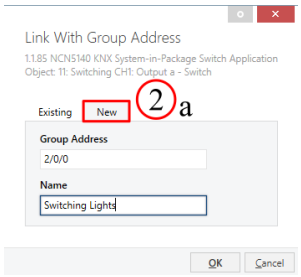

Figure 28. Linking the Group Object with a Group Address

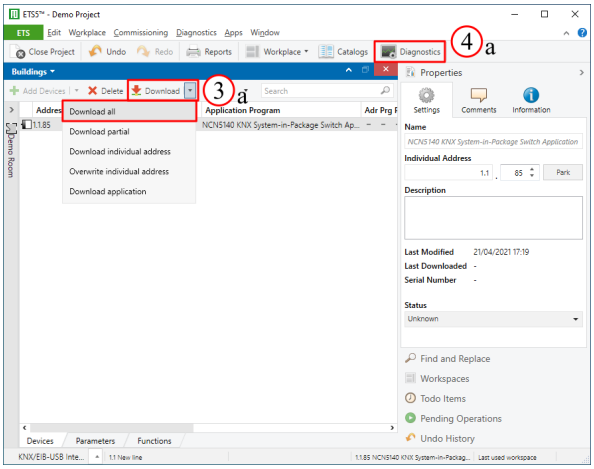

Figure 29. Download the Configuration into the Device

Now the configuration is complete and can be loaded in the device. First make sure that everything is connected correctly to the KNX network as described in section *Setting Up the Network*.

To download everything into the device, click on the arrow next to the *Download* button ③a on the top of the screen. Then select *Download all* and ETS will immediately start downloading the configuration. ETS will prompt you to press the programming button on the device in which you want to download the configuration. Do this by pressing the button on the back of the evaluation board.

Once the download has finished we can test if everything is working in the *Diagnostics* window of ETS. Open this window by clicking on the *Diagnostics* button ④a on the top of the screen. The *Group Monitor* will be opened which can be used to transmit and receive messages over the bus. To start monitoring the bus communication click on the *Start* button ⑤a.

Once the connection is established, press the CH1 button on the evaluation board. The group address 2/0/0 that was assigned to the switching object should now be transmitted over the bus as shown in figure 30.

After successfully loading a first configuration into the device, you can start experimenting with all the available ETS parameters.

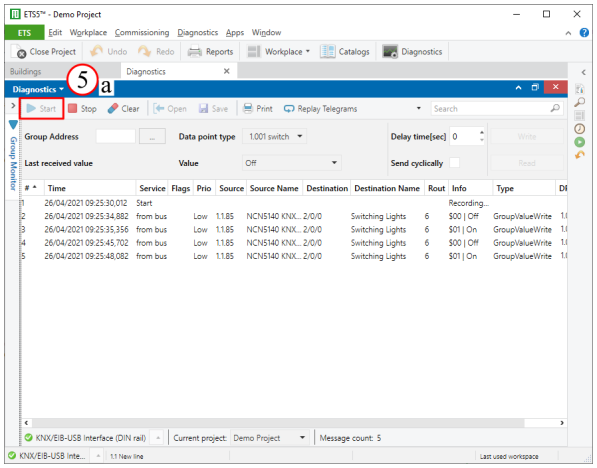

Figure 30. Monitoring the Bus Communication in the *Diagnostics* Window

Configuring The Device

The evaluation boards contain several individualization/configuration parameters listed in section *Device Configuration*. The evaluation boards come pre−programmed with the correct configuration parameters as listed in section *Device Configuration*.

These parameters together with other aspects such as the serial number, must be programmed into the device during production. For development purposes, it is possible to use Net'n Node from Weinzierl Engineering GmbH to program these.

A free license key can be obtained to use the software with limited functionality. This is sufficient to configure the evaluation boards for development purposes.

Installing Net'n Node

To install the software browse to the Weinzeirl website[18] and download the free version of Net'n Node. Apply to get a free license following the instruction on the registration page[19].

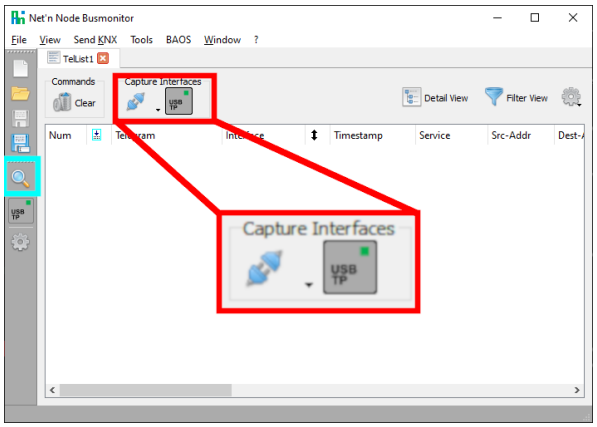

Figure 31. Net'n Node Connect to KNX USB/IP Interface

After receiving the license file, install the software by running the downloaded executable. Run through the install instructions and install the software in the desired location. The obtained license file (*.wzlic) has to be copied into the root directory of the Net'n Node installation. When running Net'n Node it should now recognize the license file.

Once the software is installed and running, it must connect to a KNX interface in order to communicate over the bus. The software can connect to several types of interfaces such KNX−IP or KNX−USB. For more information about what is supported, refer to the Net'n Node documentation. To automatically detect all interfaces connected to the PC, click on the magnifying glass (figure 31). Figure 31 shows a KNX to USB interface was found. When clicking on it, Net'n Node will connect to it. A message that the connection was successful will appear in the telegram list.

Net'n Node is now ready to be used to configure the evaluation boards.

Changing The Authorization Key

For development purposes, it is possible to change the authorization key using Net'n Node[20] from Weinzierl Engineering GmbH. For more information on installing and setting up Net'n Node refer to section *Installing Net'n Node*.

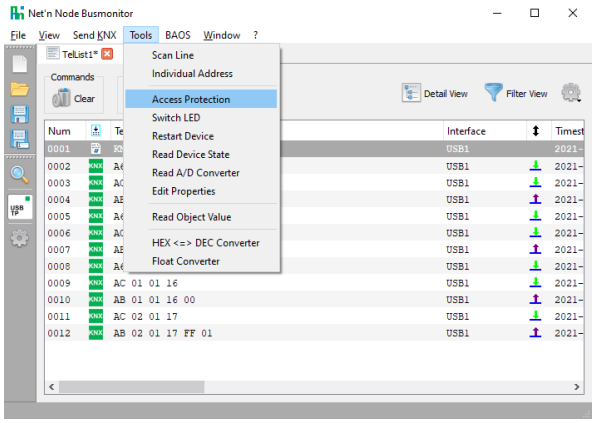

Figure 32. Open the *Access Protection* Window in Net'n Node

The authorization key can be changed through the *Access Protection* window in Net'n Node. Open this window by going to *Tools Access Protection*. In this window the individual address of the device being configured ① needs to be set first. The evaluation boards have individual address 15.15.255 by default. The currently installed authorization key has to be entered into the *Key* field ②. This key is *12 34 56 78* by default. To check if the key is correct, click on the *Test Key* button. The returned test key level must be 1.

After confirming that the currently installed key is correct, enter the new key into the *New Key* field and set the *Level to change* to 1 ③. Click on the *Set Key* button to program the

18. https://www.weinzierl.de/index.php/en/all−knx/software−tools−en/net−n−node−en
19. https://www.weinzierl.de/index.php/en/all−knx/software−tools−en/net−n−node−en−reg
20. https://www.weinzierl.de/index.php/en/all−knx/software−tools−en/net−n−node−en

new key into the device. If everything went well, this operation should finish without any errors.

Download Configuration

Once the right configuration for your application has been selected, it must be downloaded into the NCN5140S over the KNX−bus. During development this can also be done through Net'n Node. Refer to section *Installing Net'n Node* for more information on installing and setting up the software.

To edit the configuration loaded into the device, go to *Tools → Edit Properties*. This will open a new window shown in figure 35. With this window it is possible to read out the current configuration and edit it.

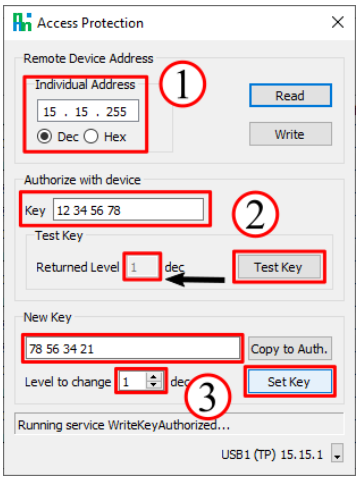

Figure 33. Change the Authorization Key through Net'n Node

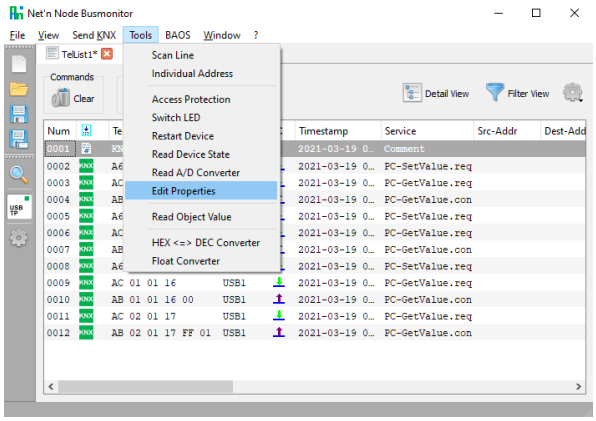

Figure 34. Open the *Edit Properties* Window

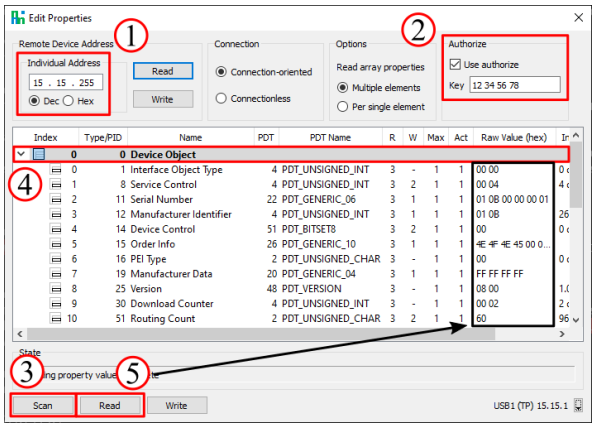

Figure 35. Overview of the *Edit Properties* Window

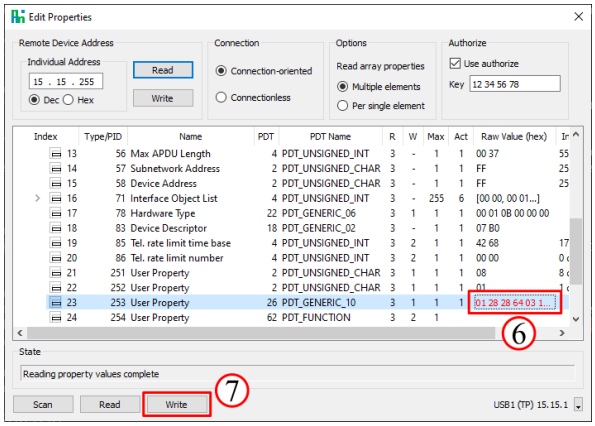

Figure 36. Changing a Property

First enter the individual address of the device ①. The individual address programmed into the evaluation boards is 15.15.255 by default. Next select *Use authorize* ② and enter the authorization key 12 34 56 78. If the authorization key was changed as described in section *Authorization* enter the newly set key here. For more information on authorization refer to section *Authorization*.

Once this setup is complete it is possible to scan the device ③ and get a list of all the configuration properties which are present. The scan will take some time and afterwards the configuration properties window ④ will be filled in. In this window unfold the *Device Object* ④. Under *Device Object* all the properties described in section *Device Configuration* are listed. However their *Value* column is still empty.

To read out the current value of the different properties, click on *Device Object* ④ and then press read ⑤. The *Raw Value* column will now be filled in.

To edit a property, double click on its *Raw Value* ⑥ and change it. Changed values will be marked in red as shown in figure 36.

Writing away the new property value is done one property at a time. Select the property to write by clicking on it and then click on the *Write* button ⑦. Once the new value is written in the device, the *Raw Value* will no longer be marked in red.

CERTIFICATION

These two hardware designs are fully KNX certified and both have a main certificate. The main certificates are for the full configuration with 8 channels, RGB LEDs and a programming button. Both designs are also fully CE compliant and certified.

Derived Certificate

Because a main certificate is already in place for these evaluation boards, it is possible to get a derived certificate from the KNX Association. This drastically reduces time−to−market as this allows for a simplified registration procedure and does not require any additional testing of the boards.

However there are some restrictions regarding what is allowed to change and still being able to apply for a derived certificate. All this is explained in detail in the KNX Association certification manual [1]. Below a short summary is given.

Modifications to a derived product must never exceed the following:

- Colour
- Marking
- Modified housing
- Position of terminals not affecting electrical safety
- Position of operational elements and indicators
- Changing the way of mounting the device
- Original software, modifications only related to reduced number of parameters, manufacturer code, device type, version number and commercial data.

Only after changing one or more of the above mentioned items, one must submit a KNX declaration on product modification. This will yield a derived certificate.

A derived certificate is less costly than a main certificate.

Registration

To become an OEM customer it is mandatory to contact the KNX organization and follow the introductory phase. Many parts of the standard flow can be skipped, but becoming a KNX member is mandatory. All members are obliged to pay an annual member fee. For more information on pricing refer to the KNX Association website[21].

CE Certification

Both NCN5140TSCGEVB and NCN5140BSCGEVB have been tested at an accredited EMC−lab and are fully compliant to the KNX standard [2]. If the board layout guidelines are strictly followed and the layout is kept close to the existing evaluation board designs, a final CE certificate can be obtained easily.

The backside of the boxed EVBs has not been tested for contact/air discharge ESD because once installed it is no longer accessible. During initialization and installation the installer should take the necessary precautions to prevent ESD.

The complete test report can be downloaded from the ON Semiconductor website.

BIBLIOGRAPHY

Standards

[1] *The KNX Standard v2.1 − KNX Certification of Products − Part 5: Procedure. KNX, 2013.*

[2] *The KNX Standard v2.1 − KNX Hardware Requirements and Tests − Part 4−1: Safety and Environemental Requirements − General. KNX, 2013.*

[3] *The KNX Standard v2.1 − KNX Hardware Requirements and Tests − Part 4−2: Safety and Environemental Requirements − EMC Test Set−ups. KNX, 2013.*

21. https://support.knx.org/hc/en−us/articles/360000038460−Become−a−KNX−Member
https://support.knx.org/hc/en−us/articles/ 360000041039−Certification−introduction

A NET LIST FOR THE MICROCONTROLLER

Table 4. MICROCONTROLLER PINS NET LIST

Pin No. (NCN5140S)	Pin No. (MCU)	Pin Name	Function	Description
18	24	P0_0	GPIO/CapTouch	Tactile/touch button CH5
19	25	P0_1	GPIO/CapTouch	Tactile/touch button CH4
20	29	P0_5	UART TX SCB1/GPIO	TRACE UART Tx line or CH6 anode
23	37	P1_0	PWM line 2	LS2 Green cathode
24	38	P1_1	GPIO	CH5 anode
25	39	P1_2	PWM line 3	LS2 Red cathode
26	40	P1_3	GPIO	CH2 anode
29	41	P1_4	PWM line 6	LS2 Blue cathode
30	42	P1_5	GPIO	CH1 anode
31	43	P1_6	GPIO	CH4 anode
32	2	ANAO/PROG	GPIO	Programming button input
34	3	P2_1	GPIO/CapTouch	Tactile/touch button CH0
35	4	P2_2	PWM line 5	LS1 Green Cathode
36	5	P2_3	GPIO/CapTouch	Tactile/touch button CH1
37	6	P2_4	PWM line 0/NTC	LS1 Blue cathode/NTC temp sensor
40	7	P2_5	GPIO/CapTouch	Tactile/touch button CH2
41	8	P2_6	PWM line 1	LS1 Red cathode
42	9	P2_7	GPIO/CapTouch	Tactile/touch button CH3
17	13	P3_2	SWDIO/GPIO/CapTouch	SWD data line or Tactile/touch button CH6
16	14	P3_3	SWDCLK/GPIO/CapTouch	SWD clock line or Tactile/touch button CH7
28	22	P4_2	CMOD	CMOD capacitor
27	23	P4_3	GPIO	CH3 anode

B CURRENT CONSUMPTION

The figures in table 5 give an indication of the typical bus current consumption of the complete application. These numbers were obtained from a single measurement on an EVB under test. The consumption will deviate from board to board due to component tolerances. Table B should only be used as a rough indication of what the board consumes worst case.

Table 5. CURRENT CONSUMPTION OF THE WHOLE APPLICATION WITH V_{bus} = 20 V AND LEDS SET TO WHITE, 100% BRIGHTNESS

# LEDs	NCN5140TSGEVB I_{bus} [mA]	NCN5140BSGEVB I_{bus} [mA]
0	5.2	5.9
1	6.1	6.9
2	7.0	7.6
3	7.9	8.4
4	8.6	9.2
5	9.5	9.8
6	10.4	10.8
7	11.2	11.7
8	11.9	12.5

C NCN5140TSCGEVB

Schematic

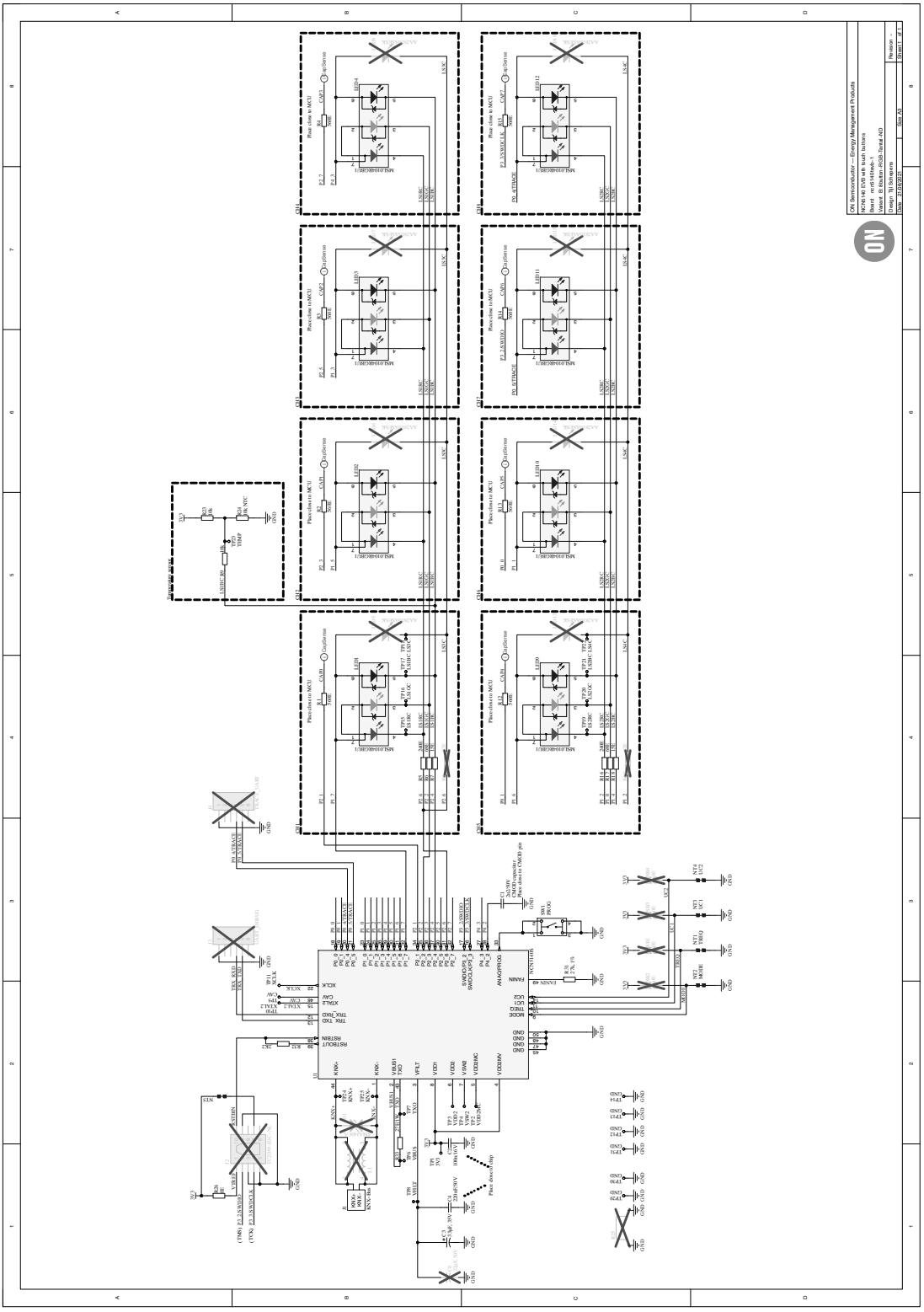

Figure 37. Schematic

BoM

Table 6. BILL OF MATERIALS

Designator	Qty	Description	Value	Part Number
SW1	1	WS−TASV SMD Tact Switch 4.5X4.5 mm		430181038816
LED1, LED2, LED3, LED4, LED9, LED10, LED11, LED12	8	Side View LED, RGB, 6.9X2.2 mm, SMD, Rectangular, R 20 mA, G 20 mA, B 20 mA		MSL0104RGBU1
R26	1	Resistor	0 Ω	RC0603JR−070RL
R32	1	Resistor	2.2 kΩ	RC0603FR−072K2L
C1	1	Capacitor	2.2 nF/50 V	GCM188R71H222KA37D
R9, R23	2	Resistor	10 kΩ	RC0603FR−0710KL
R24	1	Resistor	10 kΩ NTC	ERT−J1VG103FA
R7, R18	2	Resistor	15 Ω	ASC0603−15RFT5
R33	1	Resistor	27 Ω/1 W	352027RJT
R31	1	Resistor	27 kΩ, 1 %	CR0603−FX−2702ELF
C3	1	T598, Tantalum, Polymer Tantalum, 33 μF, 20 %, 35 VDC, SMD, Polymer, Molded, Low ESR, AEC−Q200, 65 mΩ, 7343, HeightMax = 3.1 mm	33 μF, 35 V	T598D336M035ATE065
R6, R17	2	Resistor	68 Ω	MCWR06X68R0FTL
C2	1	Capacitor	100 nF/16 V	GCM188R71C104KA37D
C4	1	Capacitor	220 nF/50 V	C1608X7R1H224K080AB
R5, R16	2	Resistor	240 Ω	MCWR06X2400FTL
R1, R2, R3, R4, R12, R13, R14, R15	8	Resistor	560 Ω	RC0603FR−07560RL
U1	1	KNX System−in−Package (SiP)	NCN5140S	NCN5140S
J1	1	Male connector for WAGO 243−211	Pitch: 5.75 mm/diameter: 1 mm/100 V/6 A	13.14.125

D NCN5140BSCGEVB

Schematic

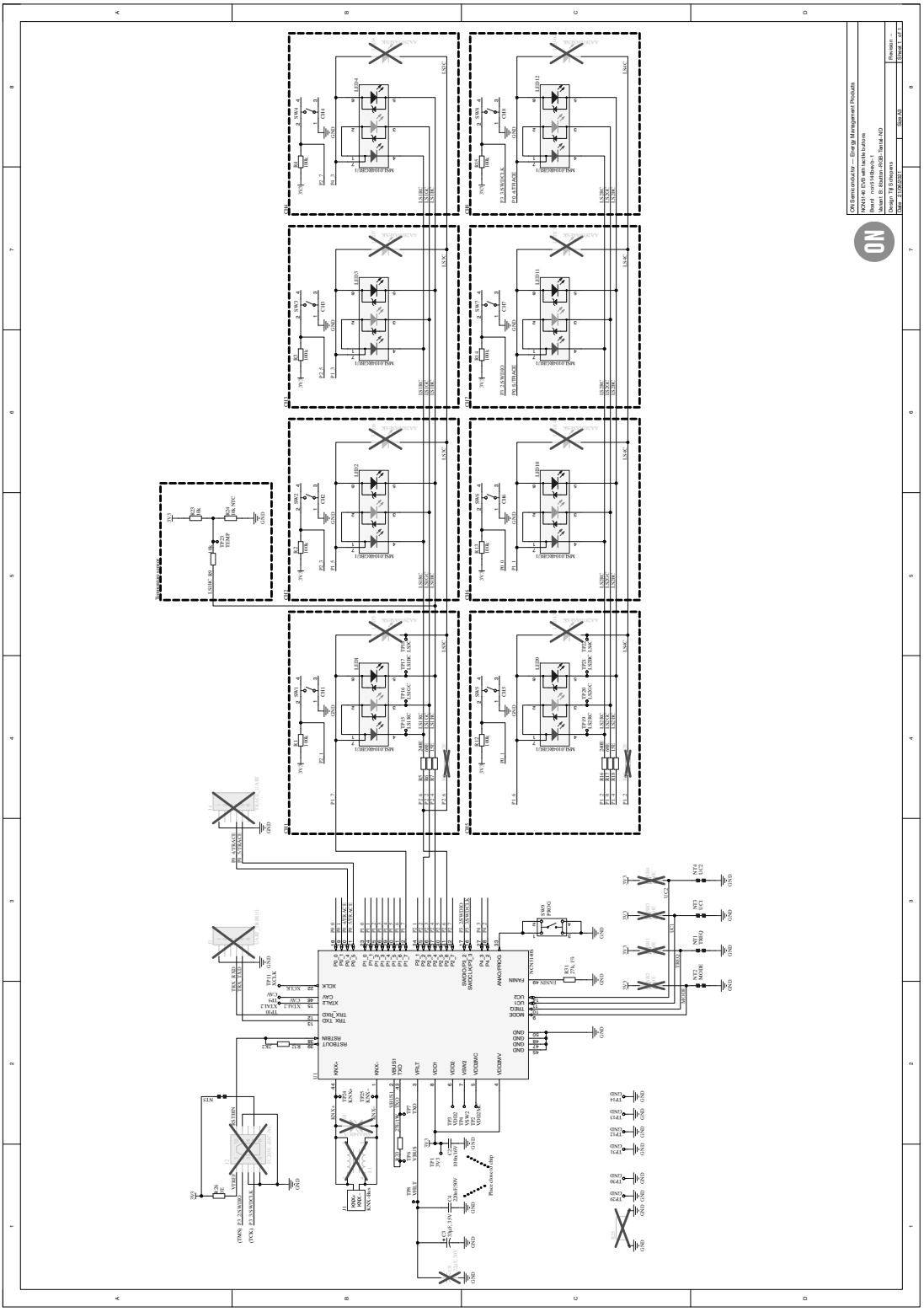

Figure 38. Schematic

BoM

Table 7. BILL OF MATERIALS

Designator	Qty	Description	Value	Part Number
SW1	1	WS−TASV SMT Tact Switch 6x6 mm, height 7 mm, 160gf		430182070816
SW2	1	WS−TASV SMT Tact Switch 6x6 mm, height 7 mm, 160gf		430182070816
SW3	1	WS−TASV SMT Tact Switch 6x6 mm, height 7 mm, 160gf		430182070816
SW4	1	WS−TASV SMT Tact Switch 6x6 mm, height 7 mm, 160gf		430182070816
SW5	1	WS−TASV SMT Tact Switch 6x6 mm, height 7 mm, 160gf		430182070816
SW6	1	WS−TASV SMT Tact Switch 6x6 mm, height 7 mm, 160gf		430182070816
SW7	1	WS−TASV SMT Tact Switch 6x6 mm, height 7 mm, 160gf		430182070816
SW8	1	WS−TASV SMT Tact Switch 6x6 mm, height 7 mm, 160gf		430182070816
SW9	1	WS−TASV SMD Tact Switch 4.5X4.5 mm		430181038816
LED1, LED2, LED3, LED4, LED9, LED10, LED11, LED12	8	Side View LED, RGB, 6.9X2.2 mm, SMD, Rectangular, R 20 mA, G 20 mA, B 20 mA		MSL0104RGBU1
R26	1	Resistor	0 Ω	RC0603JR−070RL
R32	1	Resistor	2.2 kΩ	RC0603FR−072K2L
R9, R23	2	Resistor	10 kΩ	RC0603FR−0710KL
R24	1	Resistor	10 kΩ NTC	ERT−J1VG103FA
R7, R18	2	Resistor	15 Ω	ASC0603−15RFT5
R33	1	Resistor	27 Ω/1 W	352027RJT
R31	1	Resistor	27 kΩ, 1 %	CR0603−FX−2702ELF
C3	1	T598, Tantalum, Polymer Tantalum, 33 μF, 20 %, 35 VDC, SMD, Polymer, Molded, Low ESR, AEC−Q200, 65 mΩ, 7343, HeightMax = 3.1 mm	33 μF, 35 V	T598D336M035ATE065
R6, R17	2	Resistor	68 Ω	MCWR06X68R0FTL
R1, R2, R3, R4, R12, R13, R14, R15	8	Resistor	100 kΩ	RC0603FR−07100KL
C2	1	Capacitor	100 nF/16 V	GCM188R71C104KA37D
C4	1	Capacitor	220 nF/50 V	C1608X7R1H224K080AB
R5, R16	2	Resistor	240 Ω	MCWR06X2400FTL
U1	1	KNX System−in−Package (SiP)	NCN5140S	NCN5140S
J1	1	Male connector for WAGO 243−211	Pitch: 5.75 mm/diameter: 1 mm/100 V/6 A	13.14.125