TI Designs

KNX Thermostat TI Design Guide

TI Designs

TI Designs provide the foundation that you need including methodology, testing and design files to quickly evaluate and customize the system. TI Designs help *you* accelerate your time to market.

Design Resources

TIDM-KNXTHERMOSTAT	TI Design Files

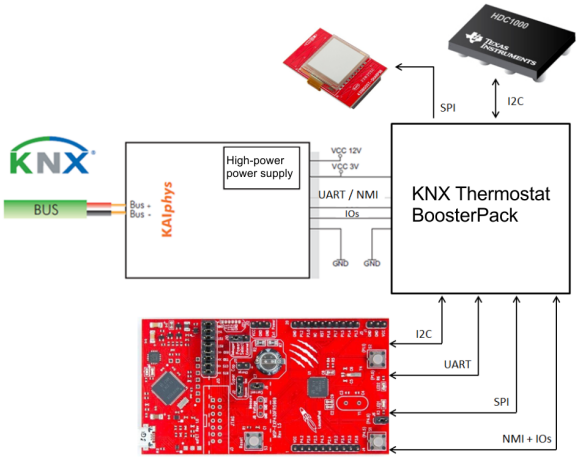
TI E2E™ Community — ASK Our E2E Experts

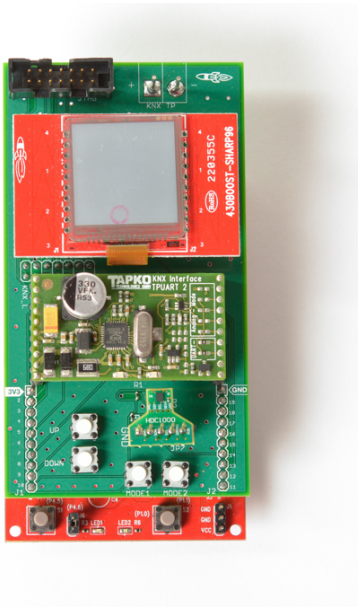
Design Features

- Fully-Certified Twisted Pair KNX Transceiver
- KNX Stack Support for TP-UART and Bit-Based TP
- Interoperability With Any Certified KNX Device
- Multi-Sensor, Multi-Actuator KNX Network Support
- KAIstack Licensing Available Through Tapko Technologies
- Thermostat With Setting Retention Upon Power Down

Featured Applications

- Thermostat





An IMPORTANT NOTICE at the end of this TI reference design addresses authorized use, intellectual property matters and other important disclaimers and information.

1 Before Getting Started

Table 1 and Table 2 shows the hardware and software tools required for this design.

Table 1. Hardware Requirements for the KNX Thermostat

Tool	Link
MSP-FET[1]	http://www.ti.com/tool/msp-fet
USB-to-KNX interface	http://www.tapko.de/en/uim-knx42
PHY module	http://www.tapko.de/en/kaistack – info@tapko.de
MSP-EXP430FR5969	http://www.ti.com/tool/msp-exp430fr5969
KNX thermostat BoosterPack™	http://www.ti.com/tool/TIDM-KNXTHERMOSTAT
KNX power supply IPS640	http://www.tapko.de/en/ips640– info@tapko.de
HDC1000 or HDC1000EVM	http://www.ti.com/product/hdc1000
	http://www.ti.com/tool/HDC1000EVM
430BOOST-SHARP96	http://www.ti.com/tool/430BOOST-SHARP96
PC Running Windows	N/A

(1) MSP-FET required only for TP-KAIphys (bit-based) PHY module.

Table 2. Software Requirements for the KNX Thermostat

Tool	Link
Application software	http://www.ti.com/tool/TIDM-KNXTHERMOSTAT
KAIstack_for_TI_demo.zip	http://www.tapko.de/en/kaistack-for-ti
IAR Embedded Workbench for MSP[1]	http://supp.iar.com/Download/SW/?item=EW430-EVAL
KNX ETS Tool	http://wbt5.knx.org

(1) Binary files of TIDM-KNXTHERMOSTAT are available in the TI design page for users who do not have access to IAR Embedded Workbench. Debugging is not possible with the binary files.

TI recommends the user of this design to review the resources in Table 3.

Table 3. Important Resources for the KNX Thermostat

Resource	Link
Using MSP on KNX Systems	*Using MSP on KNX Systems* application report (SWRA497)
KNX Organization	http://www.knx.org

2 System Description

KNX is a worldwide communication standard for home and building automation. The KNX Association owns the KNX standard. In 1999, members from the European Installation Bus Association (EIBA), the European Home Systems Association (EHSA), and BatiBUS Club International (BCI) founded the KNX Association. KNX is approved as an international standard (ISO/IEC 14543-3), European standard (CENELEC EN 50090 and CEN EN 13321-1), and Chinese standard (GB/T 20965). KNX not only defines the communication language, but also provides the set of tools and certification to ensure seamless interoperability.

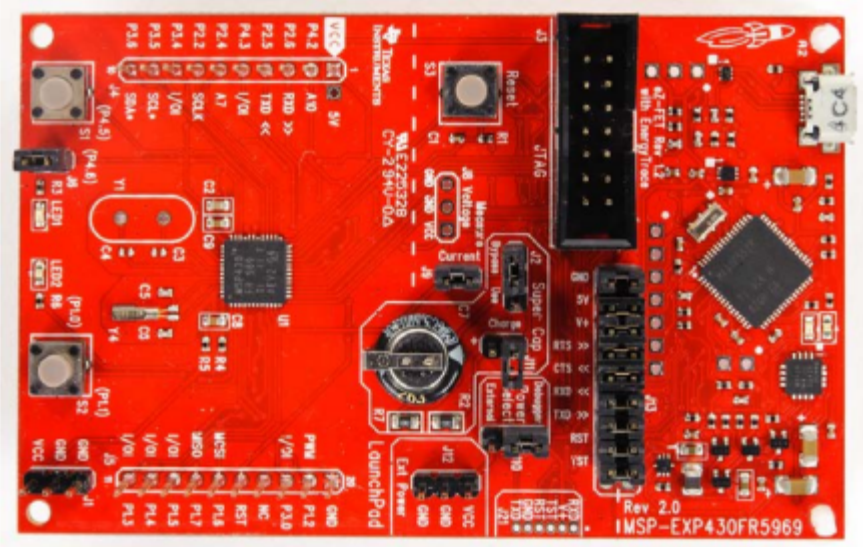
This design serves as a demonstration platform for the KNX software and hardware offering for MSP microcontrollers (MCUs) from TI and Tapko Technologies. The KNX thermostat takes advantage of the FRAM technology of the MSP430FR5969 to efficiently achieve a thermostat with persistent user settings upon power cycles. The real-time clock (RTC) of the MSP device enables the creation of temperature profiles for automatic temperature adjustment. Temperature and humidity measurements are collected by TI's HDC1000 sensor and sent to the MSP over an I^2C communication line. The 430BOOST-SHARP96 BoosterPack and four switches are used to create a user interface. The KNX thermostat BoosterPack is the hardware required to connect the different boards together to create the designed application.

The KNX thermostat supports bit-based and TP-UART KNX physical layers (PHYs). For detailed information of the KNX specifications and TI's KNX offering, see *Using MSP on KNX Systems* application report (SWRA497). Because this design presents an example application for the content in *Using MSP on KNX Systems* application report (SWRA497), TI recommends reviewing it before experimenting with the KNX thermostat in this document.

2.1 MSP-EXP430FR5969

The MSP430FR5969 LaunchPad™ serves as a simple development platform and an easy form factor to highlight applications requiring KNX protocol. The LaunchPad ecosystem provides a standard pin-out for prototyping and designing BoosterPacks for different sensors and applications. The LaunchPad also provides an easy method for debugging and communicating to the application device. TI built and tested this design Revision 2.0 of the MSP-EXP430FR5969. Any revision of the MSP-EXP430FR5969 equal to or more recent than this revision is compatible with this design. Although TI selected the LaunchPad ecosystem for this design, stand-alone MSP430FR5xx devices may also be used for a similar application. For more information, see Figure 1.

Figure 1. MSP-EXP430FR5969



The MSP430FR5969 on board the LaunchPad is the application MCU. This MCU uses the eUSCI modules to communicate to the HDC1000 sensor through I^2C, to the 430BOOST-SHARP96 display BoosterPack through SPI, and to the KNX PHY module through UART (TP-UART) or I/O lines (bit-based). The MSP430™ also uses its RTC module in combination with the low-power and fast FRAM read and write capabilities to enable the generation of power-cycle persistent temperature profiles. The MSP430FR5969 manages the KNX communication stack (KAIstack) from Tapko Technologies. The KNX stack manipulates the data for successful KNX communication. The application accesses the KAIstack through an application interface (API). This API contains numerous possibilities. Through the API, you can access the communications object, interface objects, timer, and so forth.

2.2 HDC1000

In this TI Design, TI chose a digital humidity sensor with an integrated temperature sensor to demonstrate the ultralow-power, duty-cycling power scheme. Humidity and temperature are both common measurements required for many systems in industrial and building automation applications. For example, home heating and cooling systems of the future include humidity and temperature measurements in each room. With the wireless functionality of the system, this environmental information is sent to a smart thermostat, which controls the various air ducts connecting to each room. The smart thermostat provides an intelligent home environment by providing individual comfort settings and increasing energy savings. With a relative humidity accuracy of ±3% and a temperature accuracy of ±0.2°C, the HDC1000 device from TI accurately senses environmental information. The power consumption of the HDC1000 is low, averaging 1.2 µA at a one sample per second measurement rate. Interfacing to the device is easy with any MCU platform using the I^2C communication protocol. This TI Design also includes provisions for installing the HDC1000EVM for quick prototyping and testing. TI intends only one HDC device to be populated at a time.

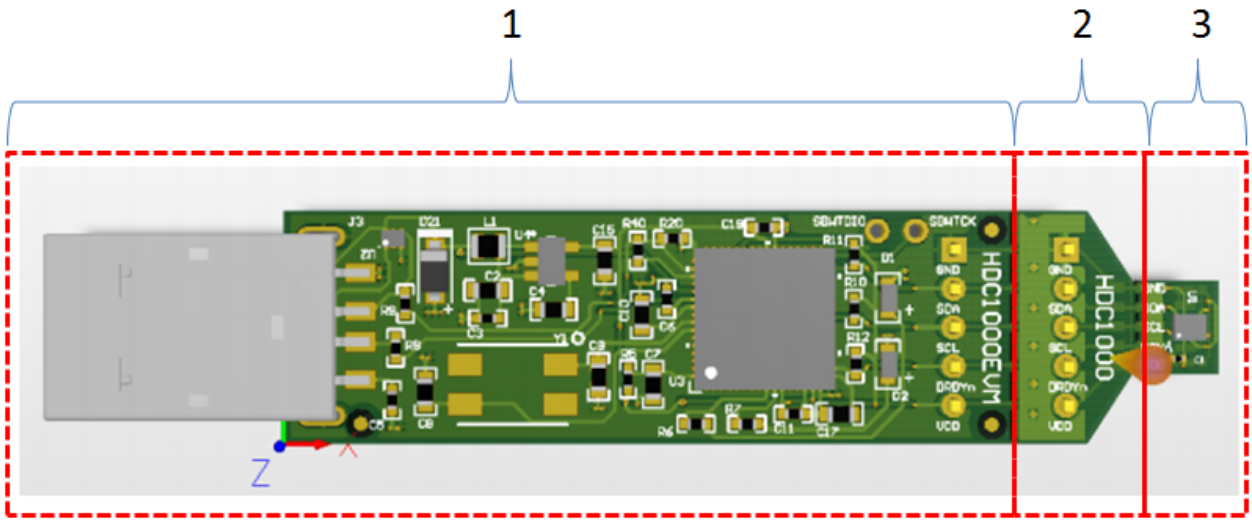
2.3 HDC1000EVM

The HDC1000EVM evaluation kit in Figure 2 is a plug-and-play system to test and evaluate the HDC1000 humidity and temperature sensor. The EVM is a breakable PCB which consists of the following three sections:

1. A USB-to-I^2C converter based on MSP430F5528 MCU
2. A conversion board (WCSP to SIL 100-mil pitch) with the HDC1000
3. A narrow 5-mm × 5.5-mm PCB with the HDC1000 (WCSP to SIL 50-mil pitch) that allows reduction of the thermal mass of the system (sensor and PCB)

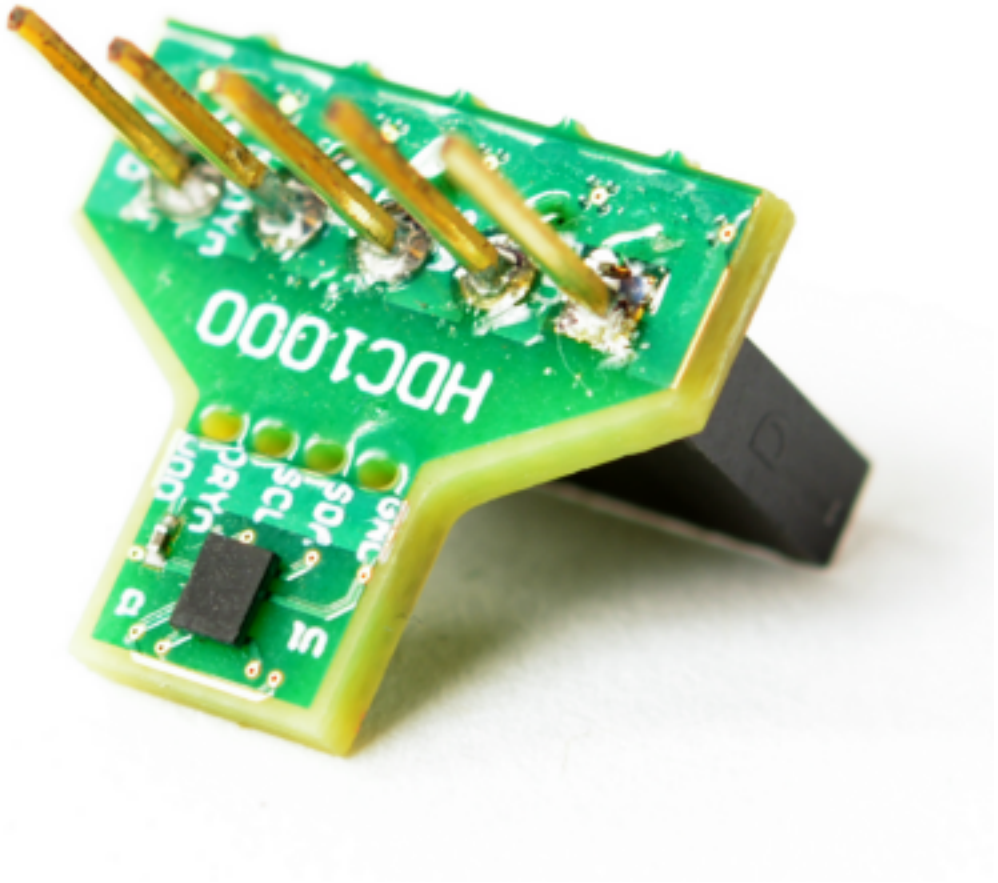
The conversion board and narrow PCB with HDC1000 can be used for remote measurements.

Figure 2. HDC1000EVM



The KNX thermostat BoosterPack is compatible with the sections 2 and 3 of Figure 2. For more information, see Figure 3 .

Figure 3. HDC1000EVM Section 2 and 3



2.4 430BOOST-SHARP96

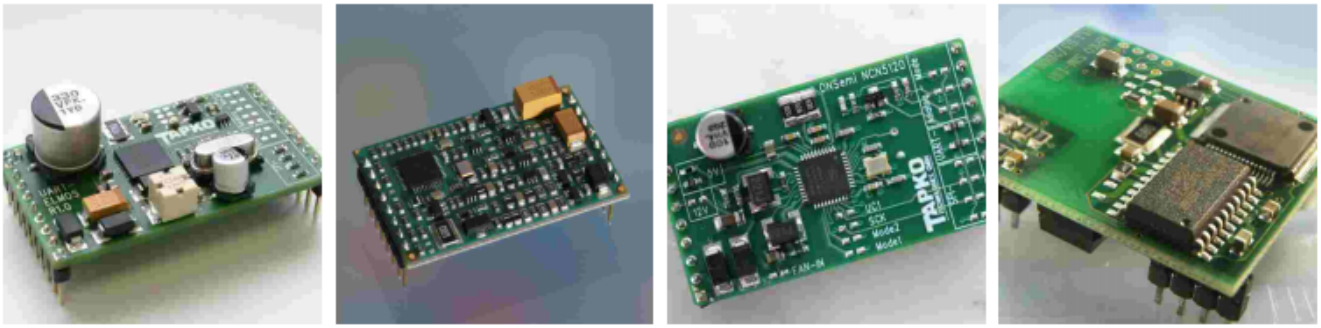
The Sharp® Memory LCD BoosterPack plug-in module is based on the LS013B4DN04 display from Sharp and features capacitive touch sliders. MCU LaunchPad evaluation kit developers can use this BoosterPack to display sensor readings, time, and other information using the 96 × 96 pixels of the display and can also provide touch-based input. See Figure 4 for more information.

Figure 4. 430BOOST-SHARP96

2.5 KNX PHY Module

A physical layer media module is required for interfacing the MSP430FR5969 to the KNX twisted pair bus. Tapko offers several media modules to satisfy UART-based and bit-based communication as in Figure 5.

Figure 5. KNX PHY Modules from Tapko Technology

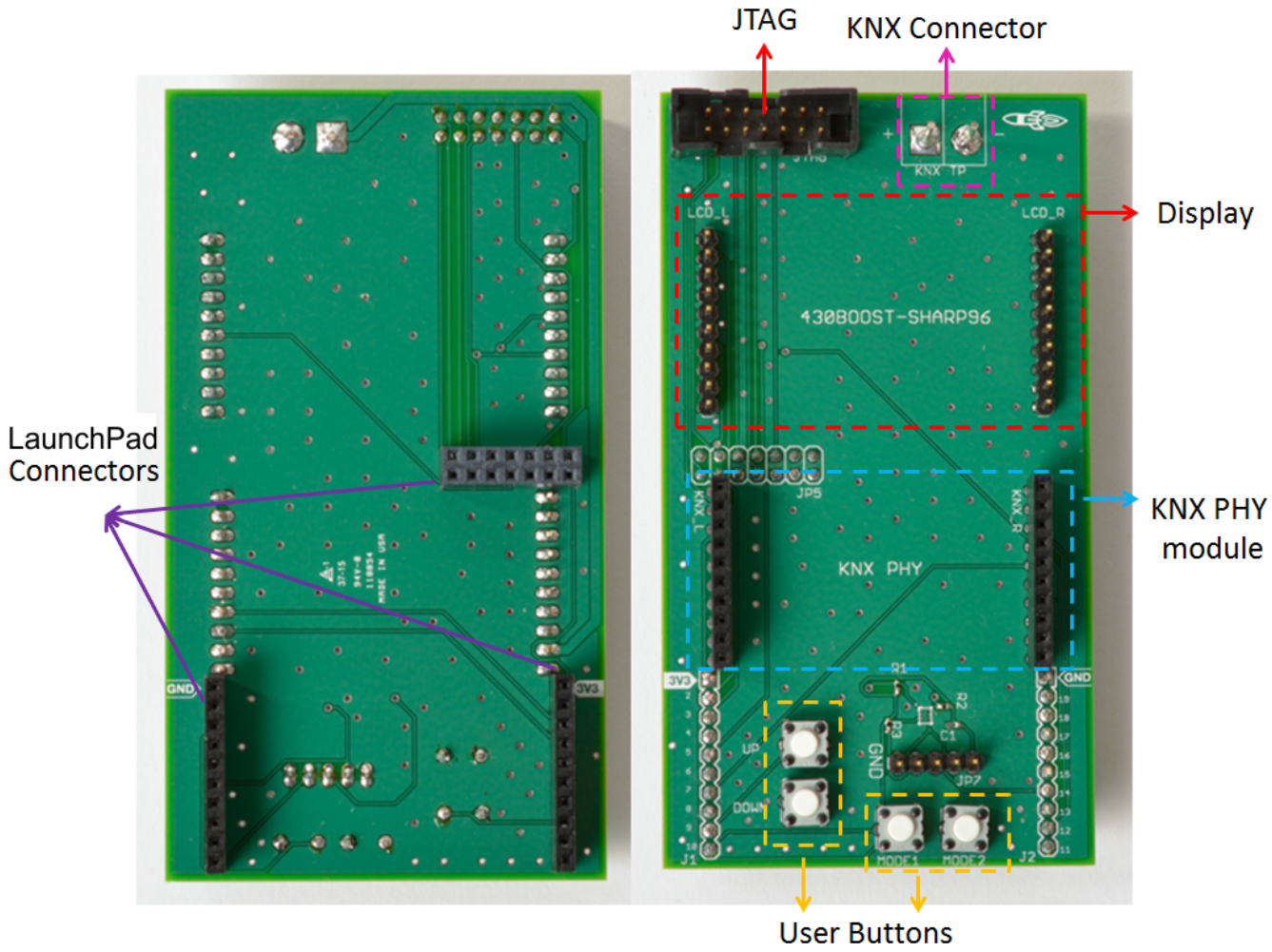


These modules contain self-sufficient hardware and complete software for a generic application. Solutions based on modules containing hardware and software enable developers to concentrate on solving the requirements of their customers. TI recommends using one of these modules for generating KNX-enabled devices in minimum time.

2.6 KNX Thermostat BoosterPack

The KNX thermostat BoosterPack is an adapter printed-circuit board (PCB) that enables the correct routing of signals to the hardware required for the application. The BoosterPack connects to the MSP-EXP430FR5969 headers and to the JTAG male connector, and enables the connection of the KNX PHY module, the HCD1000, the HDC1000EVM, the 430BOOT-SHARP96 BoosterPack, the KNX twisted-pair bus, and the MSP-FET430UIF JTAG standard connector. Four user buttons are available for application control. For more information, see Figure 6.

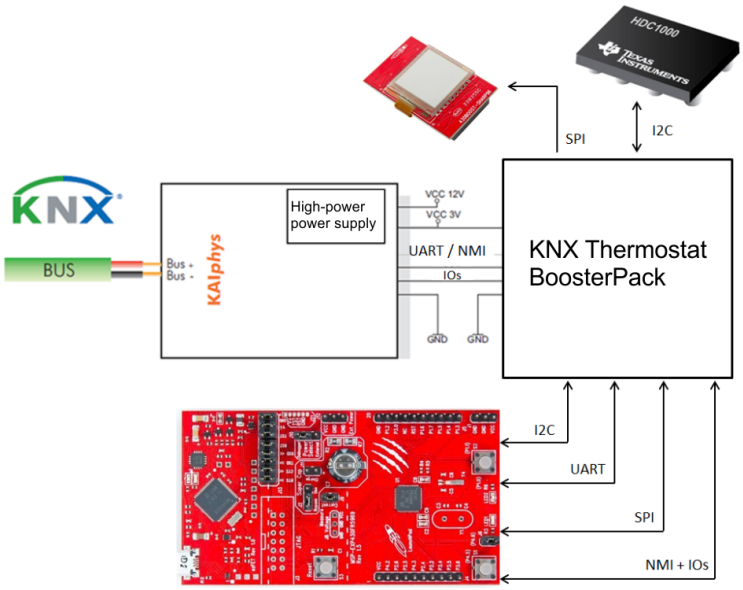
Figure 6. KNX Thermostat BoosterPack



3 Block Diagrams

Figure 7 shows the KNX thermostat sensor block diagram.

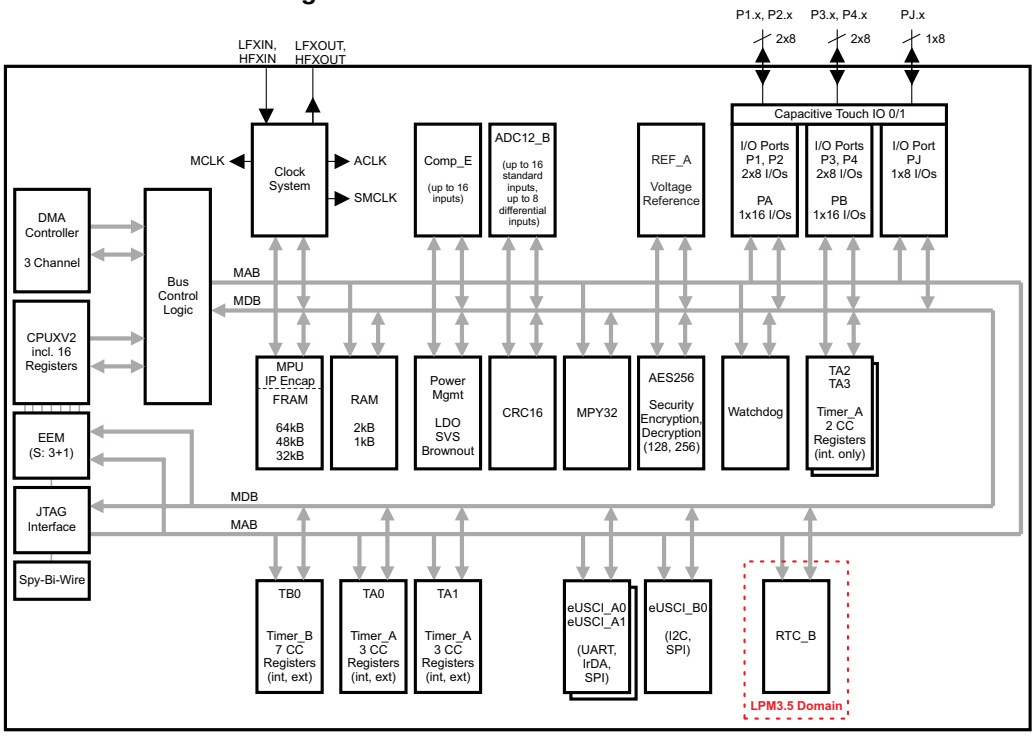
Figure 7. KNX Thermostat Sensor Block Diagram



3.1 *MSP-EXP430FR5969 Functional Diagrams*

Figure 8 shows the MSP430FR5969 block diagram.

Figure 8. MSP430FR5969 Block Diagram



3.1.1 Product Highlights

The following are product highlights of the MSP430FR5969 device:

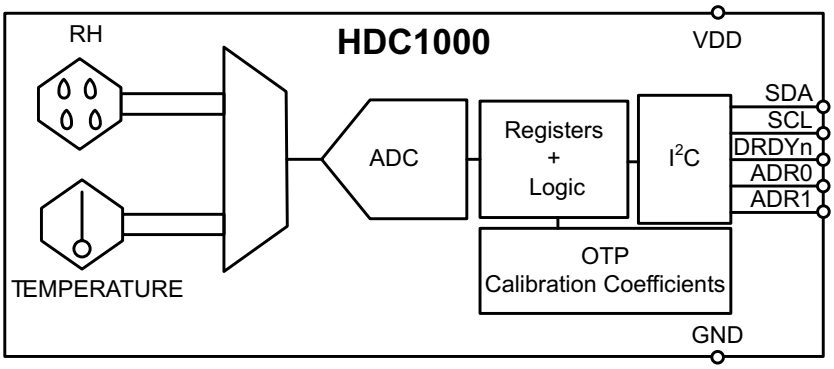
- MSP430 ULP FRAM technology-based MSP430FR5969 16-bit MCU
  - 64KB of FRAM/2KB of SRAM
  - 16-bit RISC architecture up to 8-MHz FRAM access and 16-MHz system clock speed
  - 5 timer blocks
  - Analog: 16-channel 12-bit differential ADC, 16-channel comparator
  - Digital: AES256, CRC, DMA, HW MPY32
- EnergyTrace™ + [CPU state] + [Peripheral state] available for ultralow-power debugging
- 20-pin LaunchPad standard leveraging the BoosterPack ecosystem
- 0.1-F SuperCap for stand-alone power
- Onboard eZ-FET emulation
- Two buttons and two LEDs for user interaction
- Back-channel UART through USB to PC
- Available in the TI eStore

For more information on each of these devices, see the respective product folders at www.TI.com.

3.2 HDC1000 Functional Block Diagram

Figure 9 shows the HDC1000 functional block diagram.

Figure 9. HDC1000 Functional Block Diagram



3.2.1 Product Highlights

The following are product highlights of the HCD1000 device:

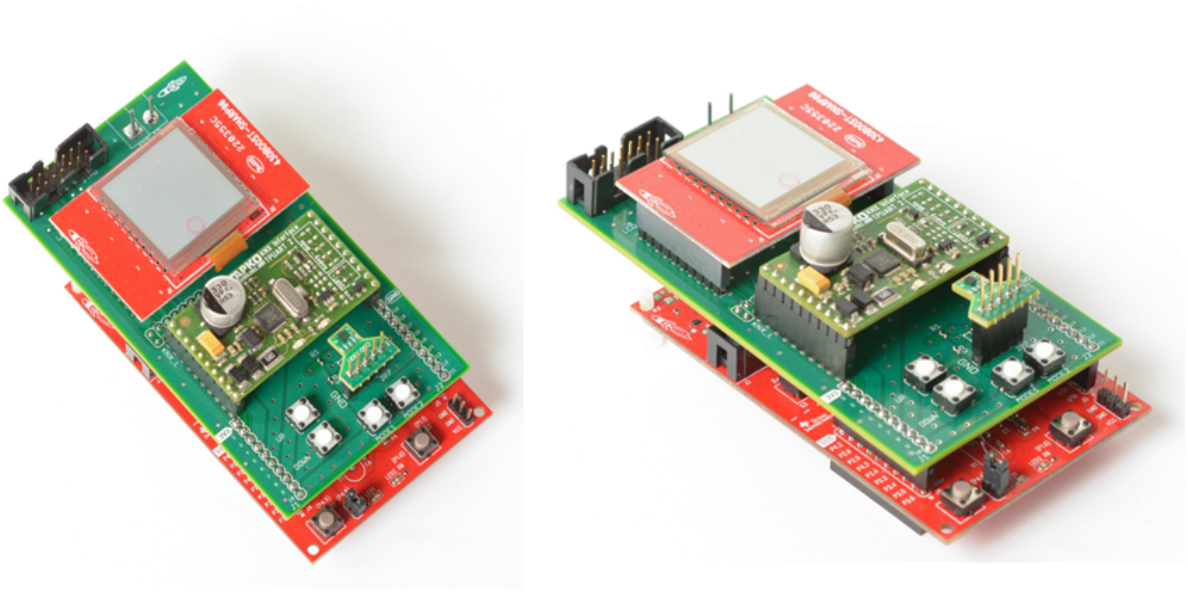
- Relative humidity (RH) operating range from 0% to 100%
- 14-bit measurement resolution
- Relative humidity accuracy ±3%
- Temperature accuracy ±0.2°C
- 200-nA sleep mode current
- Average supply current:
  - 820 nA at 1 sps, 11-bit RH measurement
  - 1.2 µA at 1 sps, 11-bit RH and temperature measurement
- Supply voltage from 3 to 5 V
- Tiny 2-mm × 1.6-mm device footprint
- I^2C interface

4 Getting Started

4.1 Hardware Overview

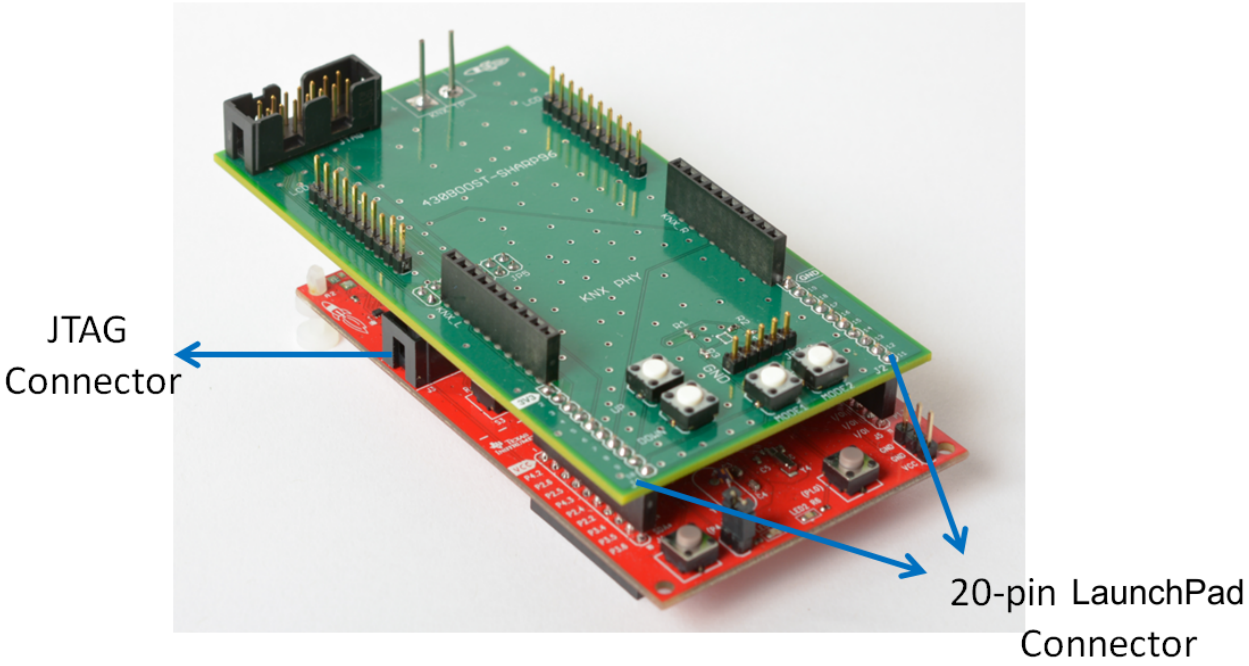
Figure 10 shows the KNX thermostat TI design.

Figure 10. KNX Thermostat Hardware



The KNX thermostat BoosterPack connects to the MSP-EXP430FR5969 as in Figure 11. This BoosterPack connects to the JTAG header as well as the 20-pin typical LaunchPad connector.

Figure 11. KNX Thermostat BoosterPack

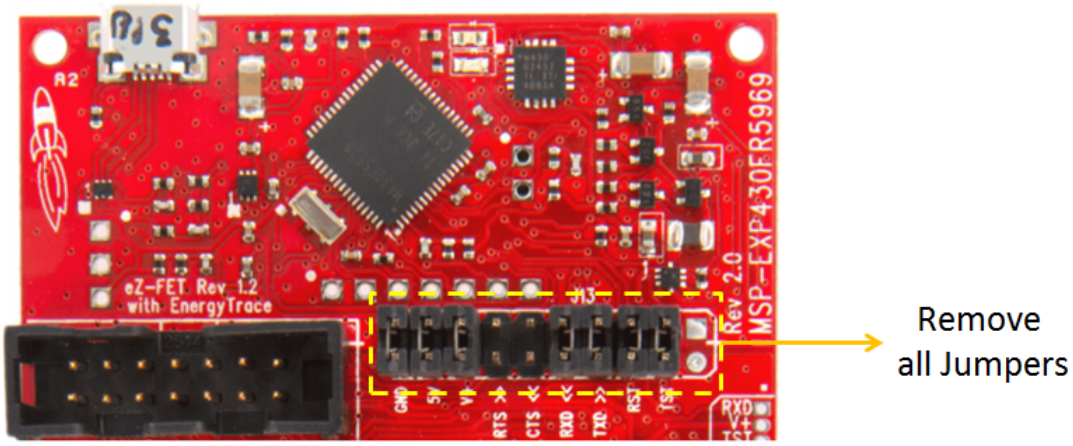


The eZ-FET onboard emulator of the MSP-EXP430FR5969 in Figure 12 is not used for this application when the TP-KAIphys (bit-based) PHY is chosen. Remove all jumpers from the J13 header.

NOTE: If the TP-UART PHY is the chosen PHY module, you can program and debug the thermostat using the eZ-FET onboard emulator. For this configuration, remove all jumpers from the J13 header except for RST and TST.

NOTE: Even when programming and debugging the device, the MSP430FR5969 must be powered externally from the KNX bus. For proper functionality when running the thermostat without the eZ-FET onboard emulator connected to a PC, remove the RST and TST jumpers.

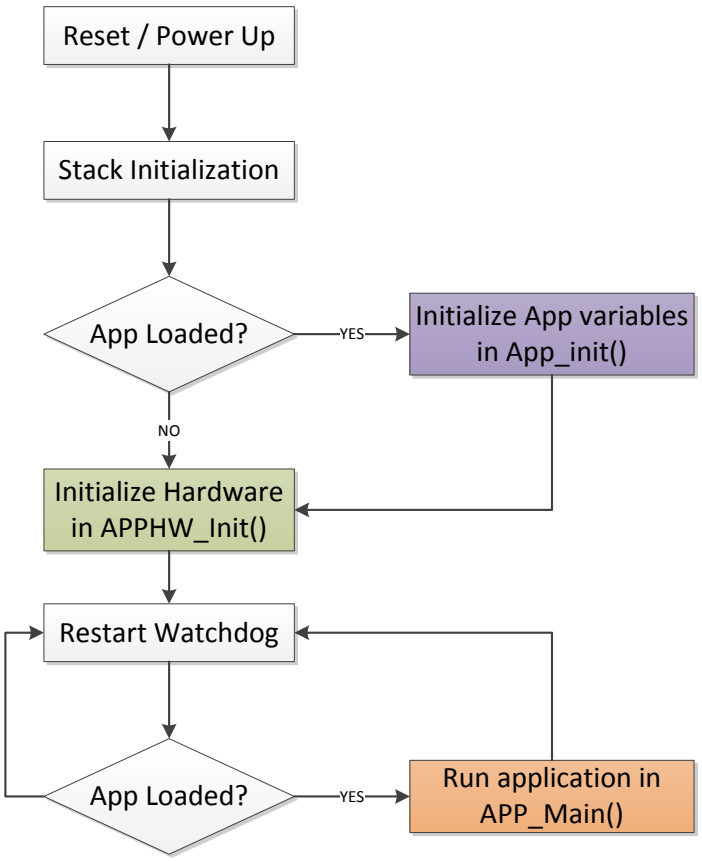
Figure 12. MSP-EXP430FR5969 Onboard Emulator



4.2 Software Overview

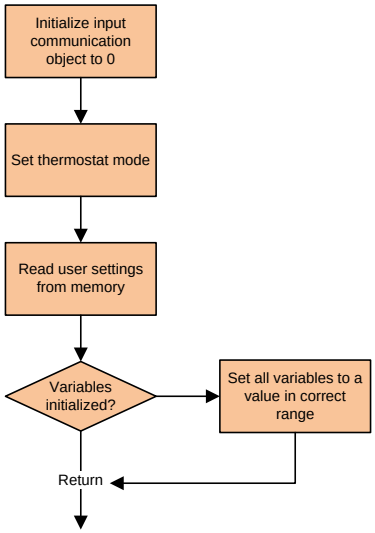
Figure 13 shows the software flow of the KNX thermostat. White blocks are KAIstack-specific blocks. Application software is colored. When a device running KAIstack is powered up, the stack initializes. This initialization includes all low-level drivers and KAIlink. If an application is loaded into the device, the APP_Init() function is called.

Figure 13. Software Flow Chart



In App_init(), the input communication object is initialized to zero. Application variables are read from memory. Application variables include the time and temperature settings of the profile and the day and time before the last power down. If all variables in memory are 0xFF, they have never been set. The initialization code sets all variables to values in the correct range. For example, the hour field cannot be 255 so it is set to a value between 0 and 23. For more information, see Figure 14.

Figure 14. Variable Initialization Routine Flow Chart



APPHW_Init() is called. In APPHW_Init(), a precompile statement checks whether a bit-based interface (KAIphys) is in use. If yes, then P2.2 is set low to enable the NMI interrupts. All GPIOs, the RTC, the display, and the HDC1000 are initialized. The first measurement request is sent to the HDC1000 through the I^2C bus. For more information, see Figure 15 and Figure 16.

Figure 15. Hardware Initialization Routine Flow Chart

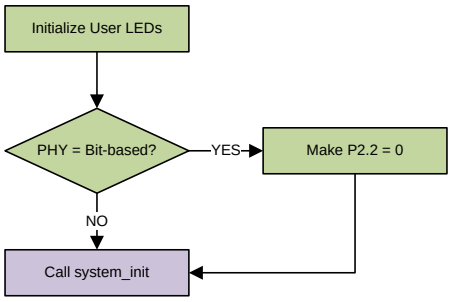
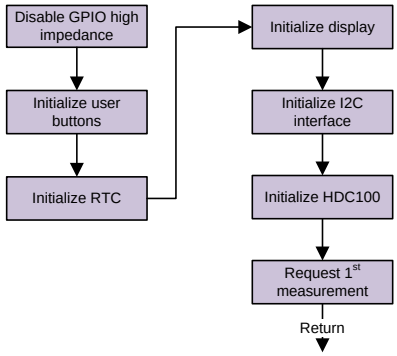


Figure 16. system_init Function Flow Chart

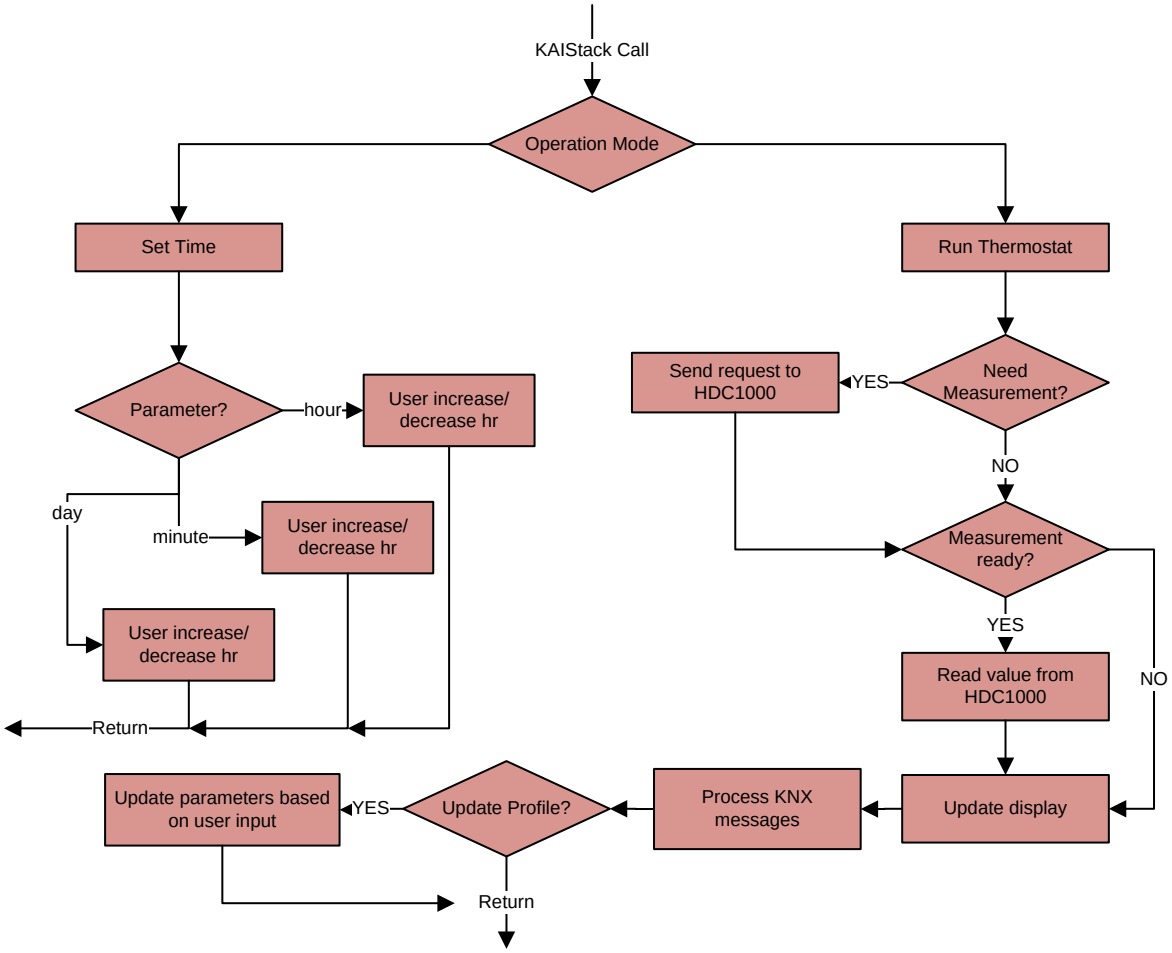


After initialization is complete, the software scheduler calls APP_Main() cyclically. The operation mode is checked in APP_Main. Two operation modes are as follows: set time mode and run thermostat mode (when the thermostat is running).

In run thermostat mode, measurements are taken every second. If a new measurement is available from the HDC1000, the value is read. The display is updated with the time, set temperature, and measured environmental conditions. Incoming messages are processed and outgoing messages are sent on the KNX bus if a change on the HVAC system is required. In this mode, the user can press the MODE2 button to update the secondary profile. The user can update the profile in this mode by continuously pressing MODE2 and using the UP and DOWN buttons.

When the user is updating the profile, the thermostat runs (measurements are taken, the display is updated, and so forth). While in the thermostat view, pressing the UP or DOWN buttons adjusts the current set temperature. If the user presses the MODE1 button, the operation mode enters *Set Time*. In this mode, measurements are not requested to the HCD1000. The user can update the current time and day in this mode by continuously pressing MODE1 and using the UP and DOWN buttons. For more information, see Figure 17.

Figure 17. Cyclic Application Flowchart



The user can press the MODE1 button to set the current time and day. The first time MODE1 is pressed, the hour on the display starts blinking. By using the UP and DOWN buttons on the BoosterPack, the user can increase or decrease the value of the hour. Pressing MODE1 again lets the user set the minute field. Pressing the MODE1 again lets the user set the day field. After setting the day, pressing MODE1 causes the application to run the thermostat normally.

The profile screen has similar navigation but uses MODE2 instead of using MODE1. For this design, P4.5 puts the device in KNX *programming mode* and the LED on P4.6 (LED1) turns on when entering this mode. For more information, see Table 4.

Table 4. Application IOs

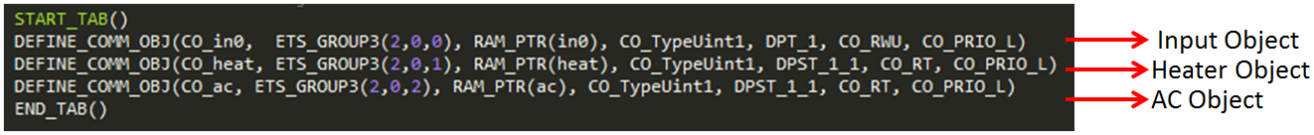
User IO	Function	Location
MODE1	Set current time and day	BoosterPack
MODE2	Set user profile	BoosterPack
UP	Increase current highlighted value	BoosterPack
DOWN	Decrease current highlighted value	BoosterPack
P4.5	Enter or exit programming mode	LaunchPad
P4.6 (LED1)	Programming mode LED	LaunchPad
P1.0 (LED2)	ETS-controlled LED	LaunchPad

4.2.1 Application Communication Objects

Two output communication objects and one input communication object are used in the KNX thermostat. One output object indicates the state of the AC, and the other output object indicates the state of the heater. The input object receives 1-bit messages from ETS (the configuration tool from the KNX association).

The input communication object has a group address of 2/0/0. The output for the heater has an address of 2/0/1 and for the AC the address is 2/0/2. These communication objects are defined in cotab.h. For more information, see Figure 18.

Figure 18. KNX Thermostat Communication Objects



4.3 Required Tools

This section describes the hardware and the software to develop with the KNX thermostat.

4.3.1 Hardware

Figure 19 shows the minimum hardware for KNX development.

Figure 19. Hardware Required for KNX Thermostat System

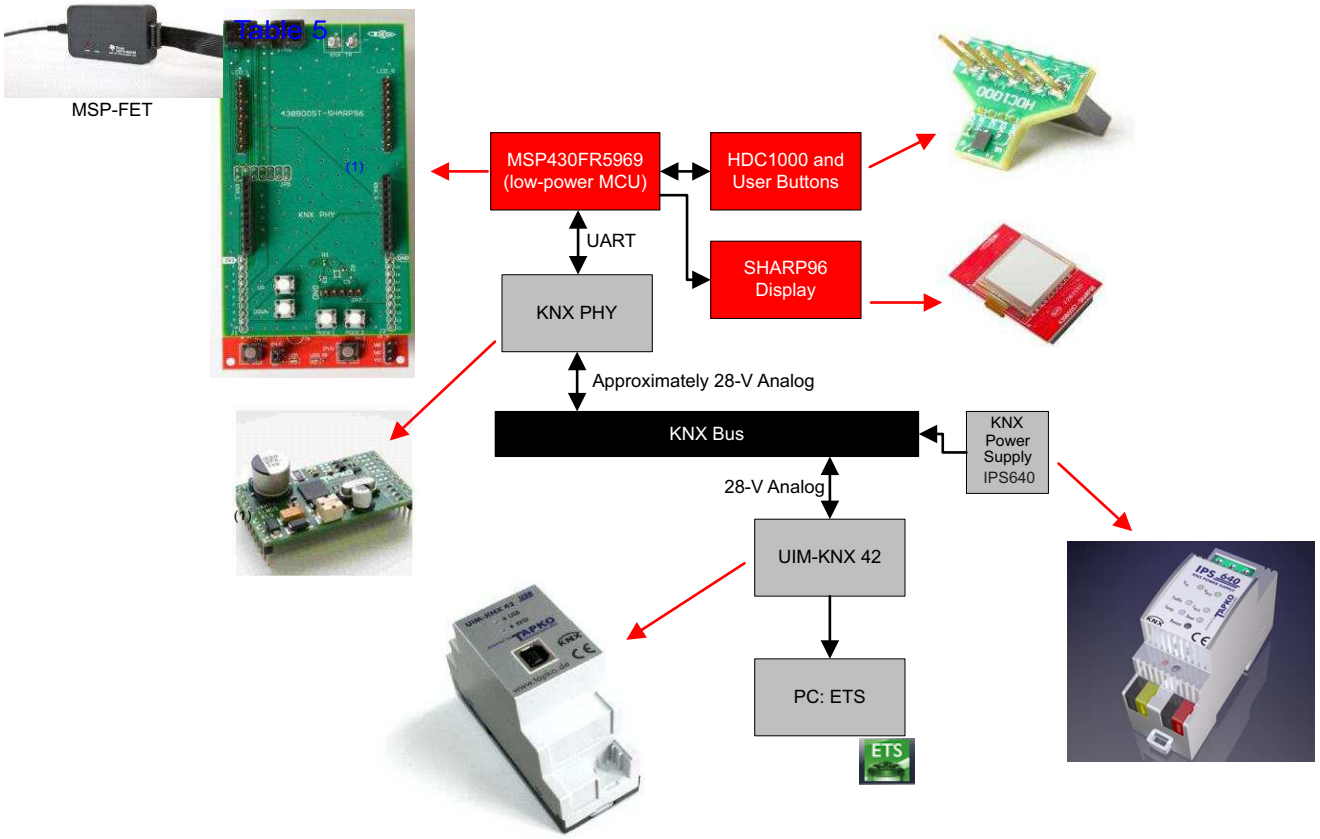


Table 5 lists where to go for more information on each component shown in Figure 19.

Table 5. Hardware Component Resources

Tool	Link
MSP-FET[1]	http://www.ti.com/tool/msp-fet
USB-to-KNX interface	http://www.tapko.de/en/uim-knx42
PHY module	http://www.tapko.de/en/kaistack – info@tapko.de
MSP-EXP430FR5969	http://www.ti.com/tool/msp-exp430fr5969
KNX thermostat BoosterPack	http://www.ti.com/tool/TIDM-KNXTHERMOSTAT
KNX power supply IPS640	http://www.tapko.de/en/ips640 – info@tapko.de
HDC1000 or HDC1000EVM	http://www.ti.com/product/hdc1000
	http://www.ti.com/tool/HDC1000EVM
430BOOST-SHARP96	http://www.ti.com/tool/430BOOST-SHARP96
PC Running Windows	N/A

(1) MSP-FET required only for TP-KAIphys (bit-based) PHY module.

4.3.2 Software

For the application software for the KNX thermostat, see http://www.ti.com/tool/TIDM-KNXTHERMOSTAT. The software package consists of the following three directories:

- \scr: contains source and header files
- \workspace\gmake: contains the app_make.gmic for the compiler
- \TIDM-KNXTHERMOSTAT_binaries: contains one binary image for TP-UART and one for TP-KAIphys for programming the device without IAR

To evaluate, Tapko Technologies provides a free version of KAIstack and KAIlink with the following limitations:

- 16 group addresses, 16 associations, and 16 communication objects
- No interface objects
- Network layer Rout-Count is set to 1
- No transport layer repetitions
- ETS is unable to change physical address
- Support for only one device derivative

Download the KAIstack software installer at http://www.tapko.de/en/kaistack-for-ti.

4.3.3 Development Tools

In addition to the KAIstack demonstration, a valid license of IAR Embedded Workbench for MSP is required. For a time-limited evaluation license of IAR, see http://supp.iar.com/Download/SW/?item=EW430-EVAL.

NOTE: Due to code size, the full version of IAR Embedded Workbench for MSP is required.

NOTE: You can program your device with the binary files provided by TI even if you do not have the full version of IAR Embedded Workbench for MSP. No debugging is possible when using this method. TI recommends using the FET-Pro430 LITE Flash programmer tool from Elprotronic (https://www.elprotronic.com/productdata) to download the binary file to memory.

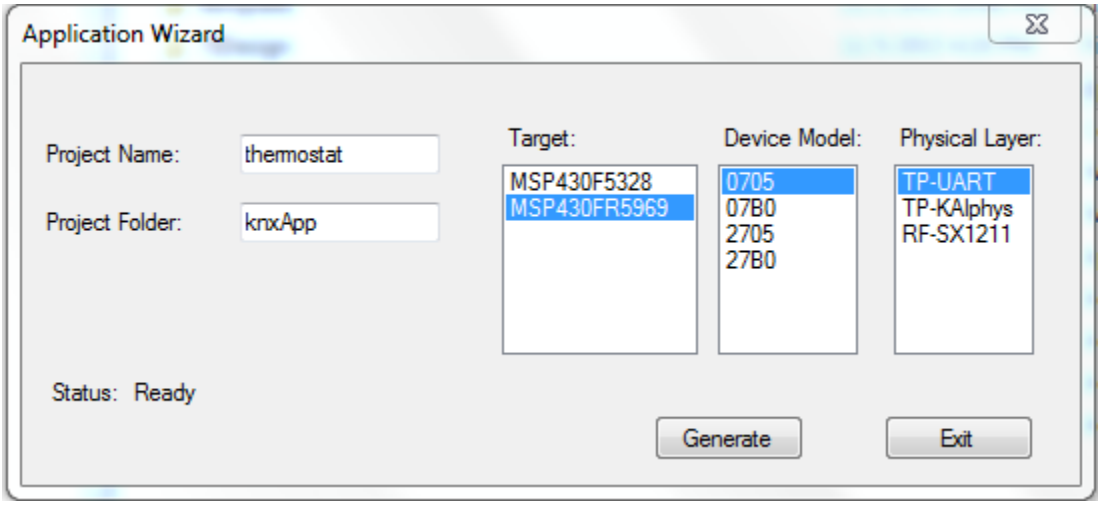
ETS is required. For an evaluation version of ETS and ETS tutorials, see the ETS learning campus at http://wbt5.knx.org.

4.3.4 System Setup

The following steps are required to set up the KNX thermostat TI Design.

1. Create a new project with AppWizard.
2. Name it *thermostat* as in Figure 20.

Figure 20. Create Thermostat Project With AppWizard



NOTE: For information on how to use the AppWizard, see *KNX on MSP* Application Report (SWRA497).

3. Replace the content of the \scr directory generated by the AppWizard with the content of the \scr directory that accompanies this TI Design.
4. To select the correct PHY module in software, open the project.h file in /src.
5. Find the #define for KNX_TARGET.
6. For TP-UART, ensure KNX_TARGET is defined as in Figure 21.

Figure 21. KNX_TARGET for TP-UART

```
#define KNX_TARGET  MSP430_IAR6_TF
```

7. For TP-KAlphys (bit-based), ensure KNX_TARGET is defined as in Figure 22.

Figure 22. KNX_TARGET for TP_KAlphys

```
#define KNX_TARGET  MSP430_IAR6_DF
```

8. Replace the file app_make.gmic generated by the AppWizard in \workspace\gmake with the .gmic file in the application software of this design.
9. Run the *rebuild* program in the project directory (this program generates a debug file in \output).
10. Open the IAR workspace file located in \workspace\iar.
11. Add the debug file to the IAR workspace.
12. Use the MSP-FET to program the MSP430FR5969 using the JTAG connector on the KNX thermostat BoosterPack.

NOTE: If using the TP-UART interface, you can connect to the MSP430FR5969 through the eZ-FET on-board emulator instead of using the MSP-FET.

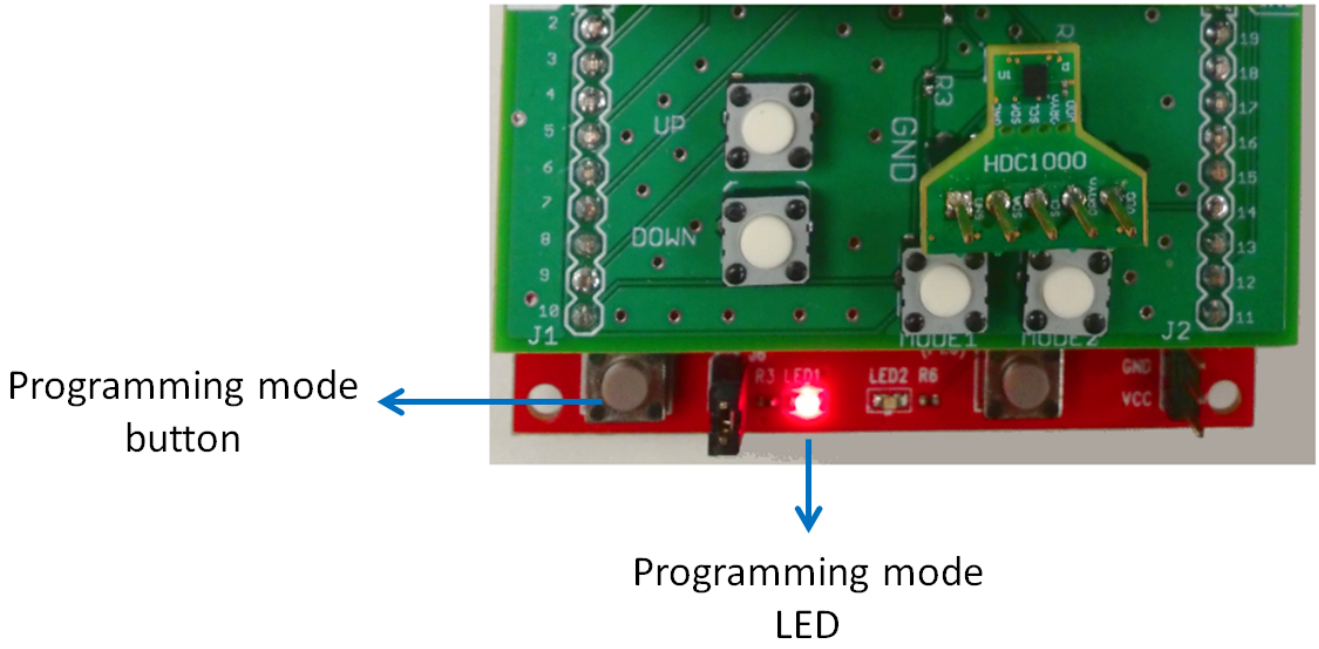
5 Test Data

ETS ensures that the KNX communication is working correctly.

5.1 Programming Mode

The KNX programming mode of the design was tested. The P4.5 button on the LaunchPad was pressed to put the MSP430FR5969 in programming mode. If the KNX is running correctly, LED1 turns on as in Figure 23.

Figure 23. Device in KNX Programming Mode



The diagnostic tool of ETS is put in programming mode. Under normal operation, ETS requests the individual addresses from all devices on the bus and each device responds with the value declared in cotab.h. See Figure 24, Figure 25, and Figure 26 for more information.

Figure 24. Successful Request and Response of Individual Address in Programming Mode

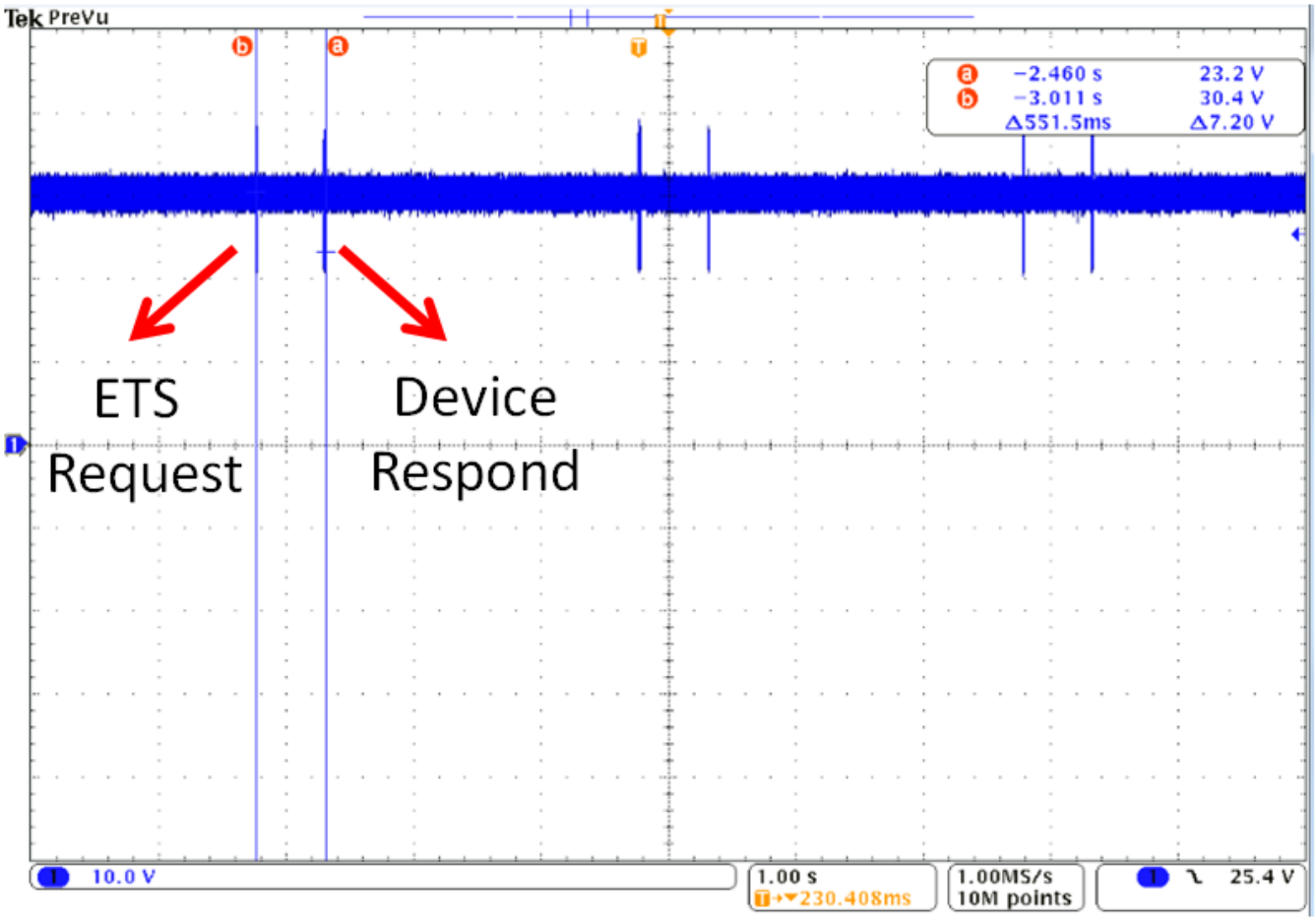
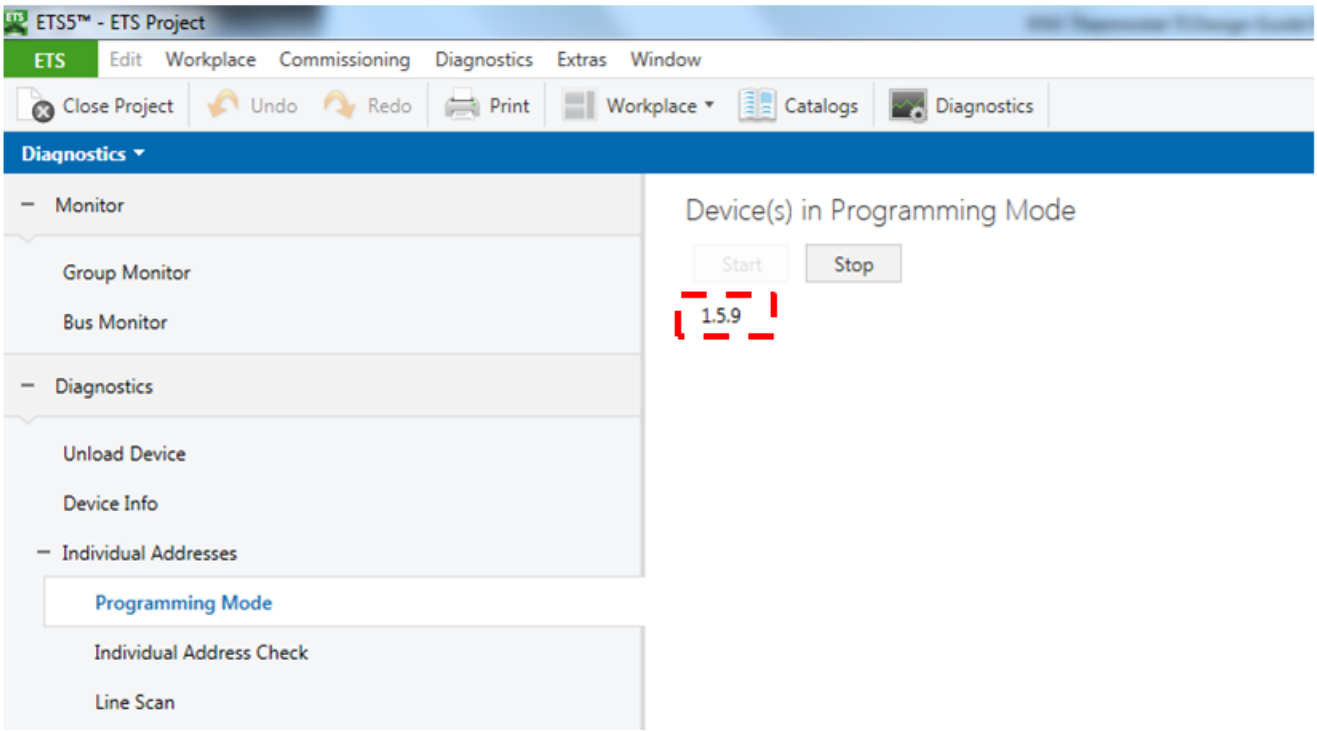


Figure 25. Individual Address Declaration in cotab.h

```
// Defines the individual address of the KNX device
#define KNX_DEF_INDIVIDUAL_ADDR EIB_ADDR(0x1509)
```

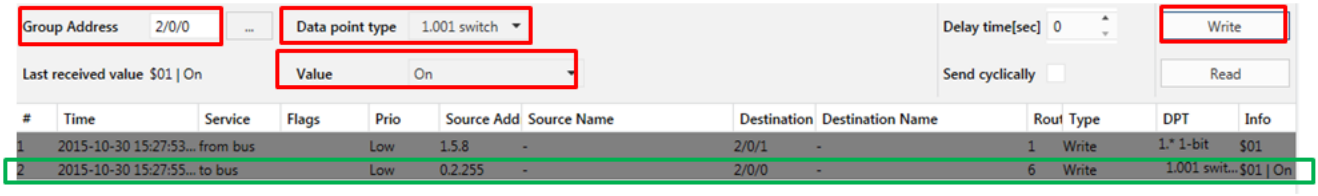
Figure 26. ETS Displaying Received Individual Address From KNX Thermostat



5.2 Sending a Telegram from a PC to a Device

KNX messages are sent from ETS to the thermostat input communication object. The input has a group address of 2/0/0 as declared in cotab.h. The input in this design is of data point type CO_TypeUint1, which is a 1-bit value, so the *1.001 switch* option is selected. A 0 (Off) or a 1 (On) value is selected in the *value* field. For more information, see Figure 27.

Figure 27. ETS Configuration to Send a 1 to the KNX Thermostat



The value of the message received is reflected by the state of LED2 on the MSP-EXP430FR5969 board. For more information, see Figure 28.

Figure 28. LED2 Turns on as a Result of the Message From ETS

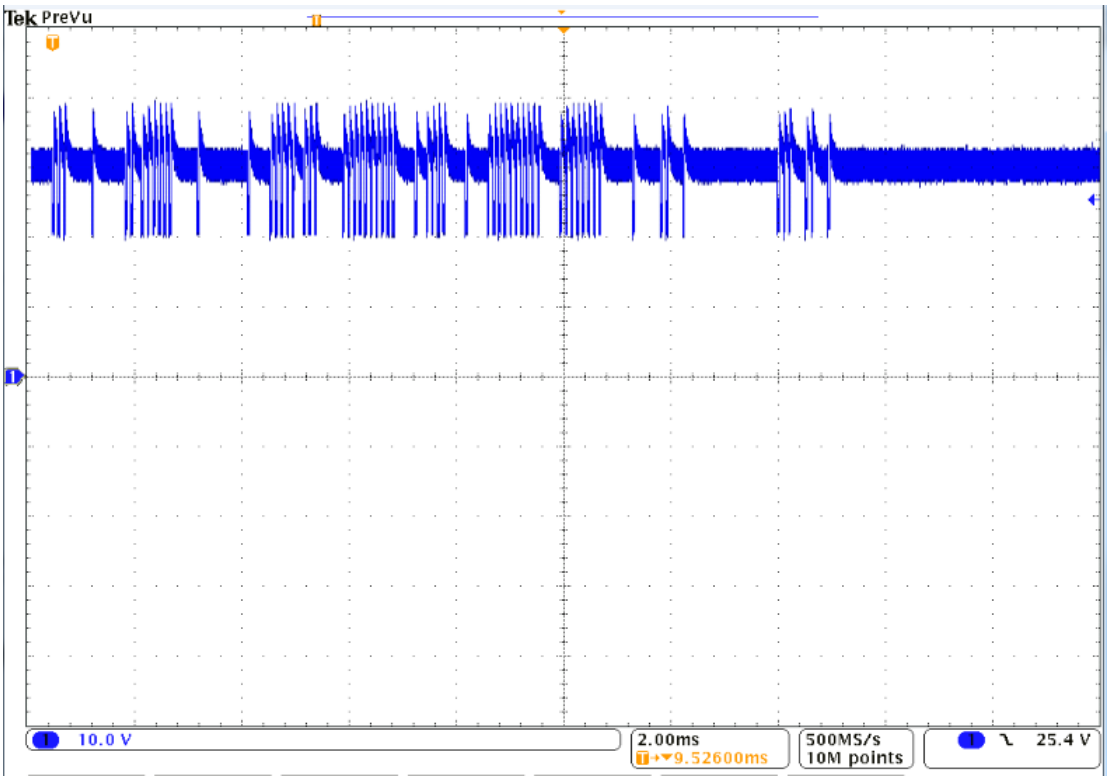
5.3 Sending a Telegram From a Device to a PC

Whenever the thermostat detects that the HVAC system must change state based on temperature, the thermostat sends a message on the bus to indicate that an action is required. Table 6 shows the expected messages based on a temperature reading (Tm) as opposed to the user temperature setup (Tt). For more information, see Figure 29.

Table 6. System Status (Tt = Target Temperature Based on User Configuration, Tm = Measured Temperature by HDC1000)

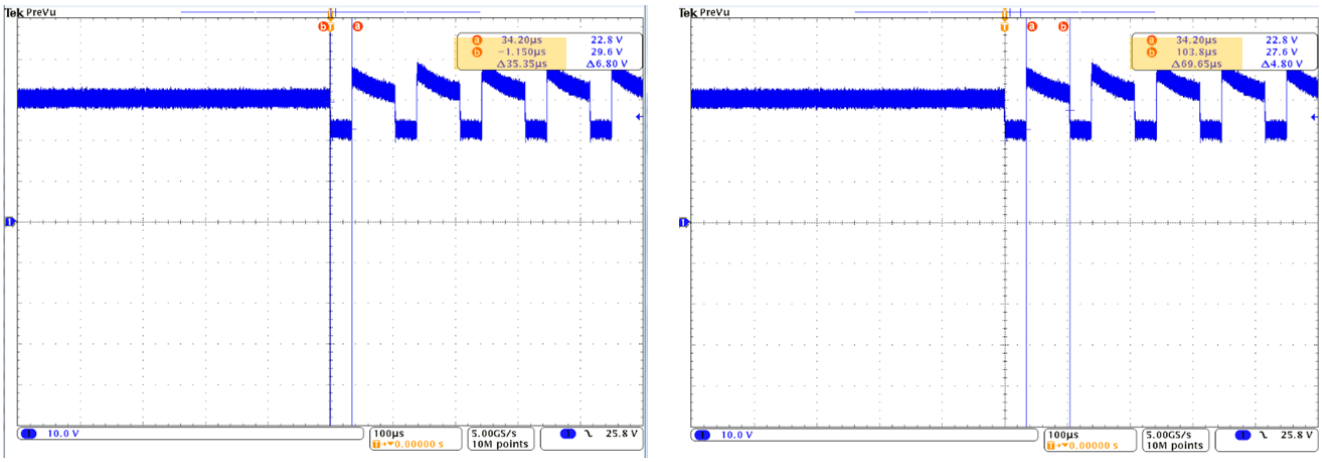
Condition	Heater	AC
Tt = Tm	Off	Off
Tt > Tm	On	Off
Tt < Tm	Off	On

Figure 29. Thermostat KNX Message on the Bus



Data is encoded on the bus based on the KNX specifications. The encoding of the logic 0 can be divided into two stages. The first stage is called an active pulse; it is a voltage drop of about 6 to 9 V in the bus voltage. This voltage dip lasts 35 µs and is generated by the transmitter sinking current from the differential lines. Each active pulse is followed by an abrupt jump of the bus voltage greater than the DC level followed by an exponential decay to the DC level with duration of 69 µs. The signal on the bus was measured to ensure KNX specifications are met. For more information See Figure 30.

Figure 30. Timing Measurement of KNX Message



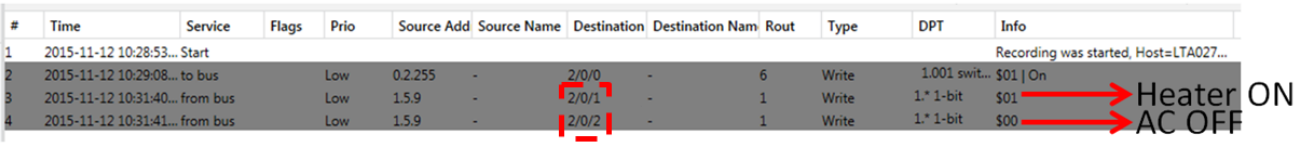
5.3.1 Turning On the Heater and Turning Off the AC

When the target temperature is greater than the measured temperature, the thermostat sends a message through the output communication objects to turn on the heater and turn off the AC. Recall that the group address of the heater is 2/0/1 and the AC is 2/0/2. For more information, see Figure 31 and Figure 32.

NOTE: The message to change the HVAC state is sent only once (the first time the state change is required). Sending the same message constantly would occupy the bus inefficiently.

Figure 31. Target Temperature (49°C) is Greater than the Measured Temperature (23°C)

Figure 32. Messages Received on ETS to Turn On the Heater On and Turn Off the AC



#	Time	Service	Flags	Prio	Source Add	Source Name	Destination	Destination Nam	Rout	Type	DPT	Info
1	2015-11-12 10:28:53...	Start										Recording was started, Host=LTA027...
2	2015-11-12 10:29:08...	to bus		Low	0.2.255	-	2/0/0	-	6	Write	1.001 swit...	$01 | On
3	2015-11-12 10:31:40...	from bus		Low	1.5.9	-	2/0/1	-	1	Write	1.* 1-bit	$01
4	2015-11-12 10:31:41...	from bus		Low	1.5.9	-	2/0/2	-	1	Write	1.* 1-bit	$00

5.3.2 Turning the AC On and the Heater Off

When the target temperature is less than the measured temperature, the thermostat sends a message through the output communication objects to turn off the heater and turn on the AC. The group address of the heater is 2/0/1 and the address of the AC is 2/0/2. For more information, see Figure 34 and Figure 33.

Figure 33. Target Temperature (0°C) is Less Than the Measured Temperature (23°C)

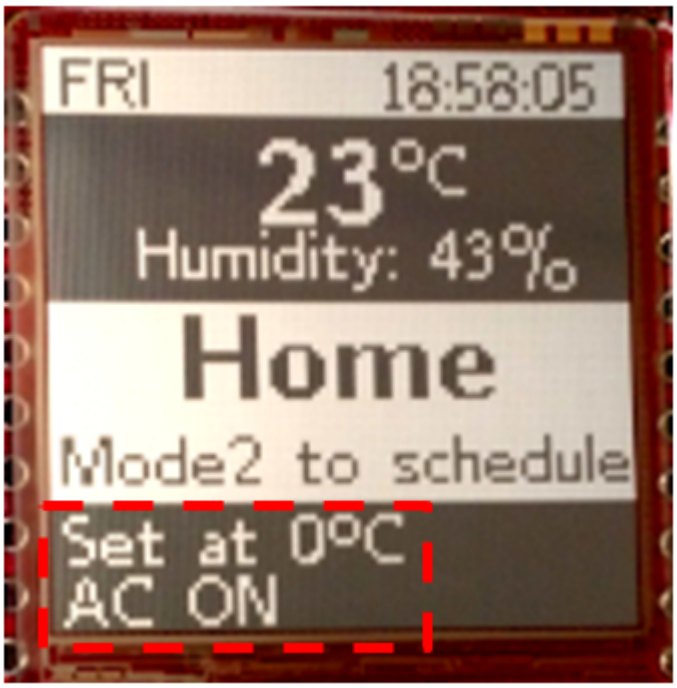


Figure 34. Messages Received on ETS to Turn Off the Heater and Turn On the AC

#	Time	Service	Flags	Prio	Source Add	Source Name	Destination	Destination Nam	Rout	Type	DPT	Info
1	2015-11-12 10:28:53...	Start										Recording was started, Host=LTA027...
2	2015-11-12 10:29:08...	to bus		Low	0.2.255	-	2/0/0	-	6	Write	1.001 swit...	$01 | On
3	2015-11-12 10:31:40...	from bus		Low	1.5.9	-	2/0/1	-	1	Write	1.* 1-bit	$01
4	2015-11-12 10:31:41...	from bus		Low	1.5.9	-	2/0/2	-	1	Write	1.* 1-bit	$00
5	2015-11-12 10:33:25...	from bus		Low	1.5.9	-	2/0/1	-	1	Write	1.* 1-bit	$00
6	2015-11-12 10:33:26...	from bus		Low	1.5.9	-	2/0/2	-	1	Write	1.* 1-bit	$01

Heater OFF
AC ON

5.3.3 Turning Off the AC and the Heater

When the target temperature is the same as measured temperature, the thermostat sends a message through the output communication objects to turn the heater and the AC off. The group address of the heater is 2/0/1 and the AC is 2/0/2. For more information, see Figure 35 and Figure 36.

Figure 35. Target Temperature (24°C) is Equal to the Measured Temperature (24°C)

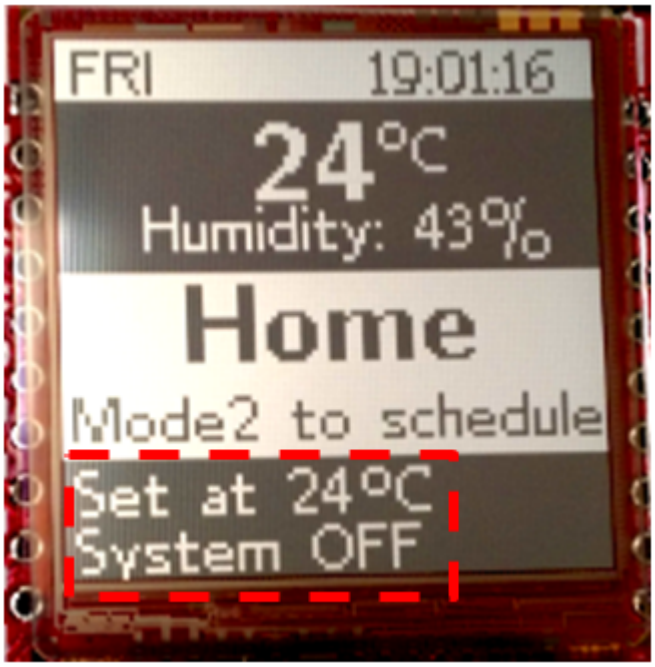


Figure 36. Messages Received on ETS to Turn Off the AC and the Heater

#	Time	Service	Flags	Prio	Source Add	Source Name	Destination	Destination Nam	Rout	Type	DPT	Info
1	2015-11-12 10:28:53...	Start										Recording was started, Host=LTA027...
2	2015-11-12 10:29:08...	to bus		Low	0.2.255	-	2/0/0	-	6	Write	1.001 swit...	$01 | On
3	2015-11-12 10:31:40...	from bus		Low	1.5.9	-	2/0/1	-	1	Write	1.* 1-bit	$01
4	2015-11-12 10:31:41...	from bus		Low	1.5.9	-	2/0/2	-	1	Write	1.* 1-bit	$00
5	2015-11-12 10:33:25...	from bus		Low	1.5.9	-	2/0/1	-	1	Write	1.* 1-bit	$00
6	2015-11-12 10:33:26...	from bus		Low	1.5.9	-	2/0/2	-	1	Write	1.* 1-bit	$01
7	2015-11-12 10:34:46...	from bus		Low	1.5.9	-	2/0/1	-	1	Write	1.* 1-bit	$00 → Heater OFF
8	2015-11-12 10:34:47...	from bus		Low	1.5.9	-	2/0/2	-	1	Write	1.* 1-bit	$00 → AC OFF

5.4 Memory Footprint

The memory requirements depend on the chosen configurations of the stack. For TP-UART, an additional approximate 0.5KB of FRAM is required for this application. See Table 7 for more information.

Table 7. Memory Footprint for Different KAIstack Configurations

PHY	FRAM (Bytes)	RAM (Bytes)
TP-UART	34360	1444
TP-Phys	33964	1444

5.5 Design Resources

The application and KNX stack use hardware resources. Table 8 summarizes the availability of the device peripherals.

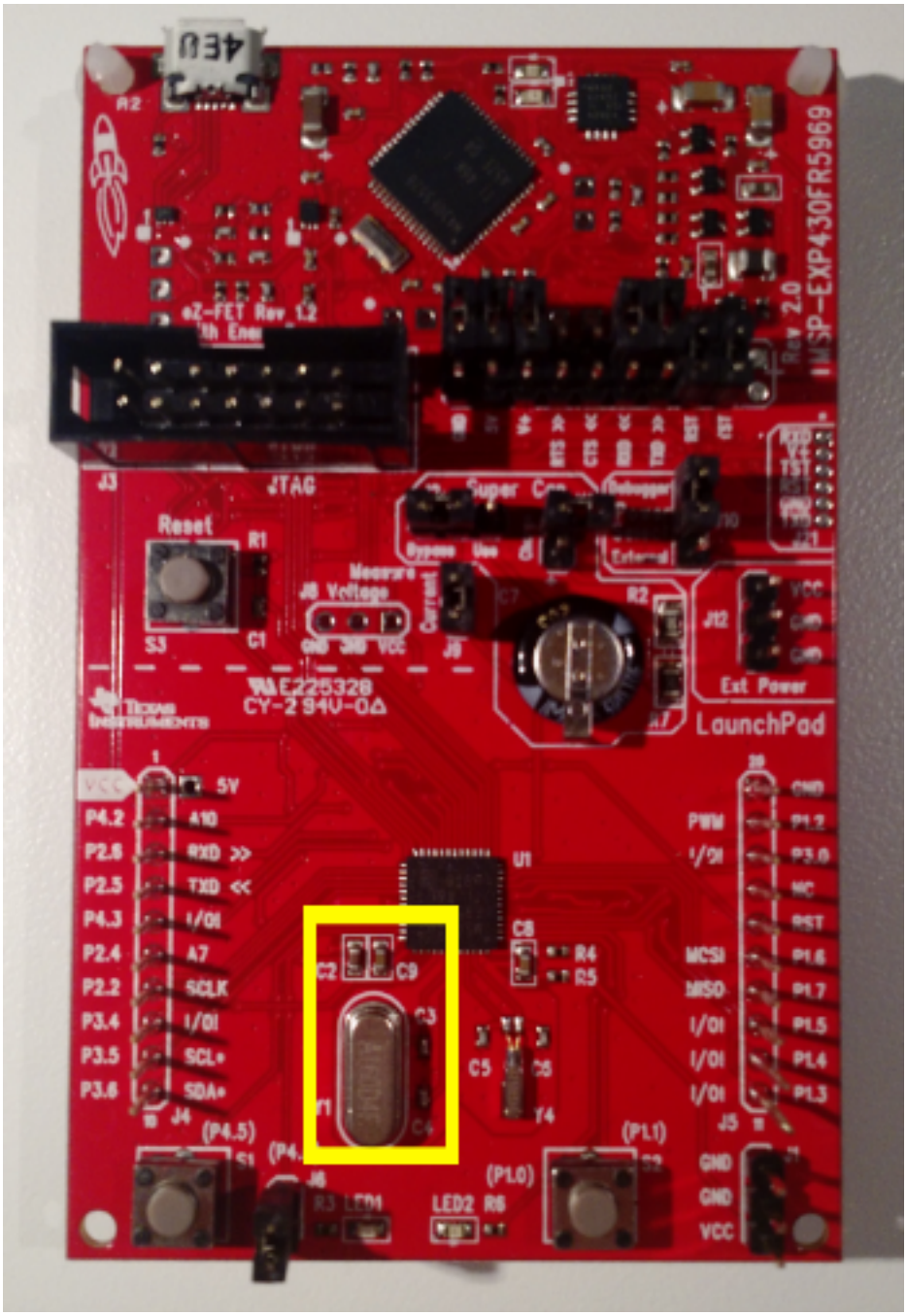
Table 8. MSP430FR5969 Peripherals

MSP430FR5969 Peripheral	Condition
Timer_B (TB0)	in use by application
Timer_A (TA0)	in use by stack
Timer_A (TA1)	in use by stack
eUSCI_A0	in use by application
eUSCI_A1	in use by application
eUSCI_B0	in use by application
RTC	in use by application
CRC16	not in use
MPY32	not in use
AES256	not in use
Watchdog	in use by stack
Comp_E	not in use
ADC12_B	not in use
REF_A	not in use

NOTE: When using a TP-KAIphys (bit-based) PHY, the NMI/RST is used for communication. As a result, the hardware reset line is not available with the TP-KAIphys PHY.

NOTE: When using a TP-KAIphys (bit-based) PHY, an external 16-MHz crystal is required on the MSP430FR5969 LaunchPad as in Figure 37 for proper KNX communication.

Figure 37. MSP-EXP430FR5969 With 16-MHz Crystal on Y1 for TP-KAIphys PHY



5.6 Stack and Application Timings

Timing delays were measured for the initialization and execution of the TIDM-KNXTHERMOSTAT as in Table 9.

Table 9. Timing Information for Initialization and Execution

Function	Time
Initialization of stack and application	1.42 ms
Stack Initialization	1.15 ms
APP_Init	16.9 µs
APPHW_Init	254.1 µs
APP_Main	97.7 ms
APPHW_Cycle	Not used
APP_Save	Not supported in TIDM-KNXTHERMOST

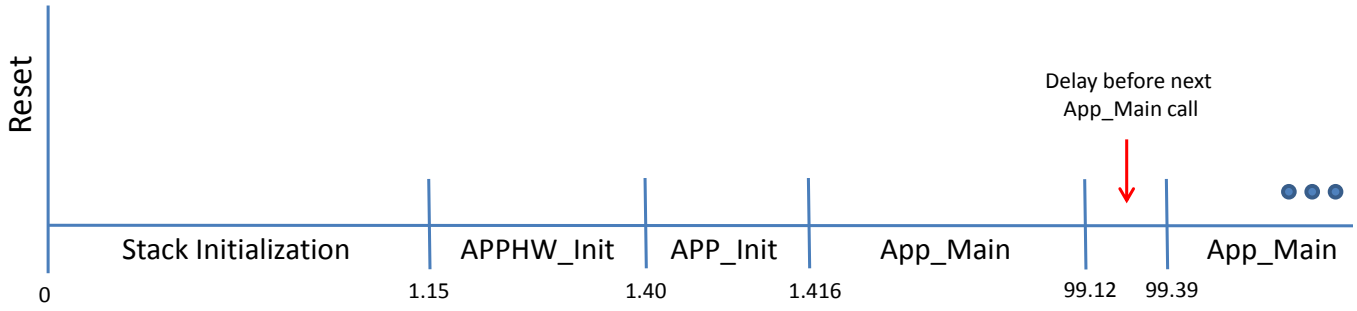
NOTE: Timing values were taken with MCU running at 16 MHz. The clock frequency can be changed in system_15\targets\TI\msp430_common\KAIstackEval_MSP430FR59xx in the selected KNX KAIstack installation directory.

NOTE: The initialization of stack and application timing was recorded from a reset to the first call of APP_Main.

NOTE: APP_main is called by the scheduler approximately 273 µs after the return from the previous APP_main execution. The delay on APP_main callback depends on the bus activity.

An execution timeline from reset to cyclic operation is shown in Figure 38.

Figure 38. KNX Thermostat Initialization and Execution Timeline

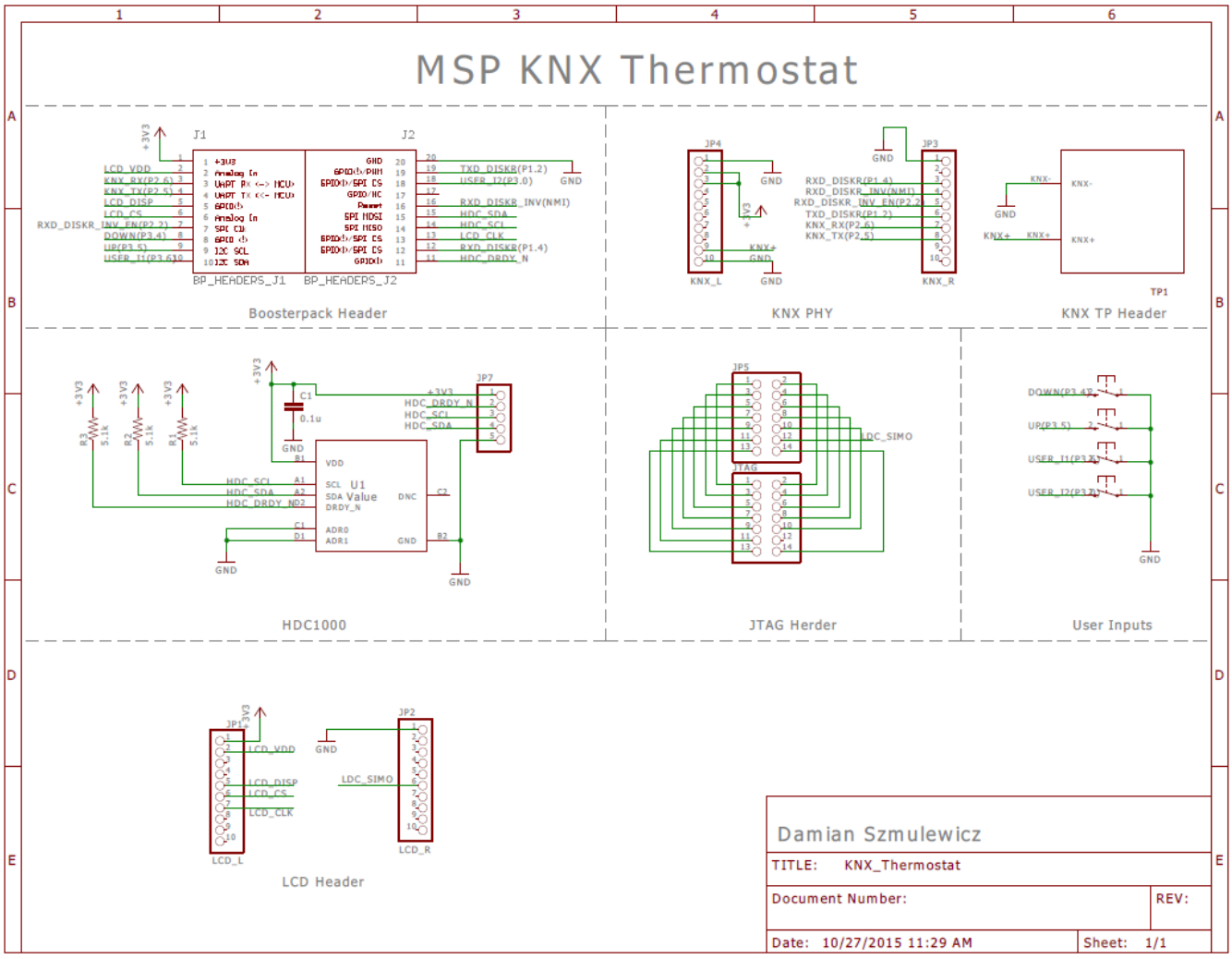


Timings not drawn to scale.

6 Design Files

6.1 Schematic

To download the schematics, see the design files at http://www.ti.com/tool/TIDM-KNXTHERMOSTAT.



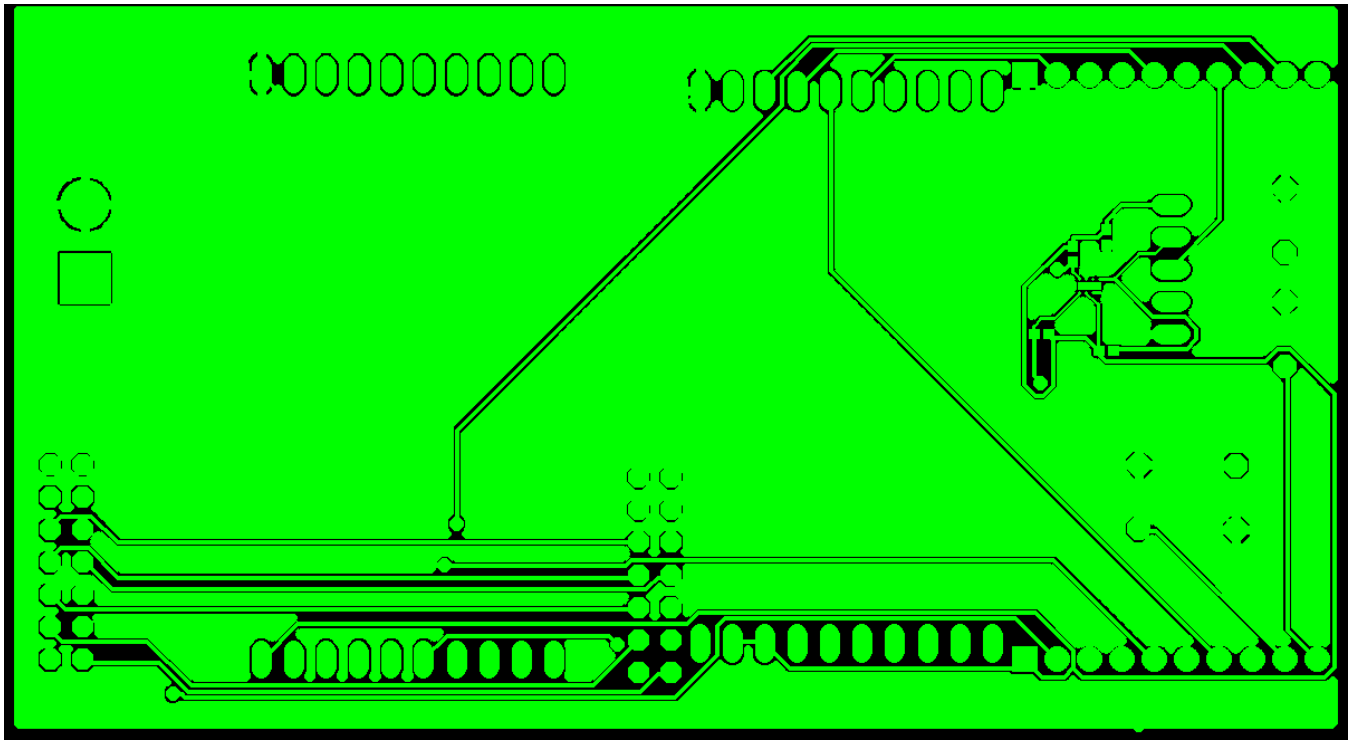
6.2 Bill of Materials

To download the bill of materials (BOM), see the design files at http://www.ti.com/tool/TIDM-KNXTHERMOSTAT.

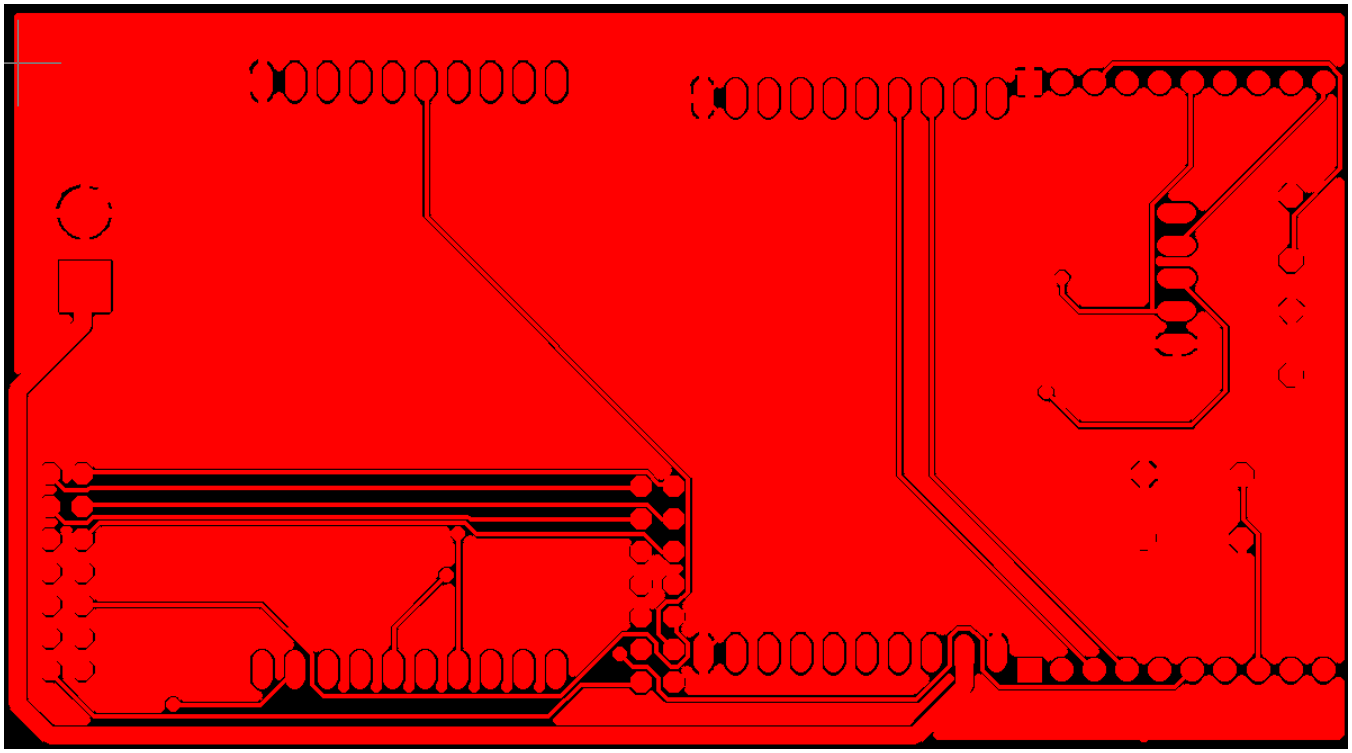
6.3 Layer Plots

To download the layer plots, see the design files at http://www.ti.com/tool/TIDM-KNXTHERMOSTAT.

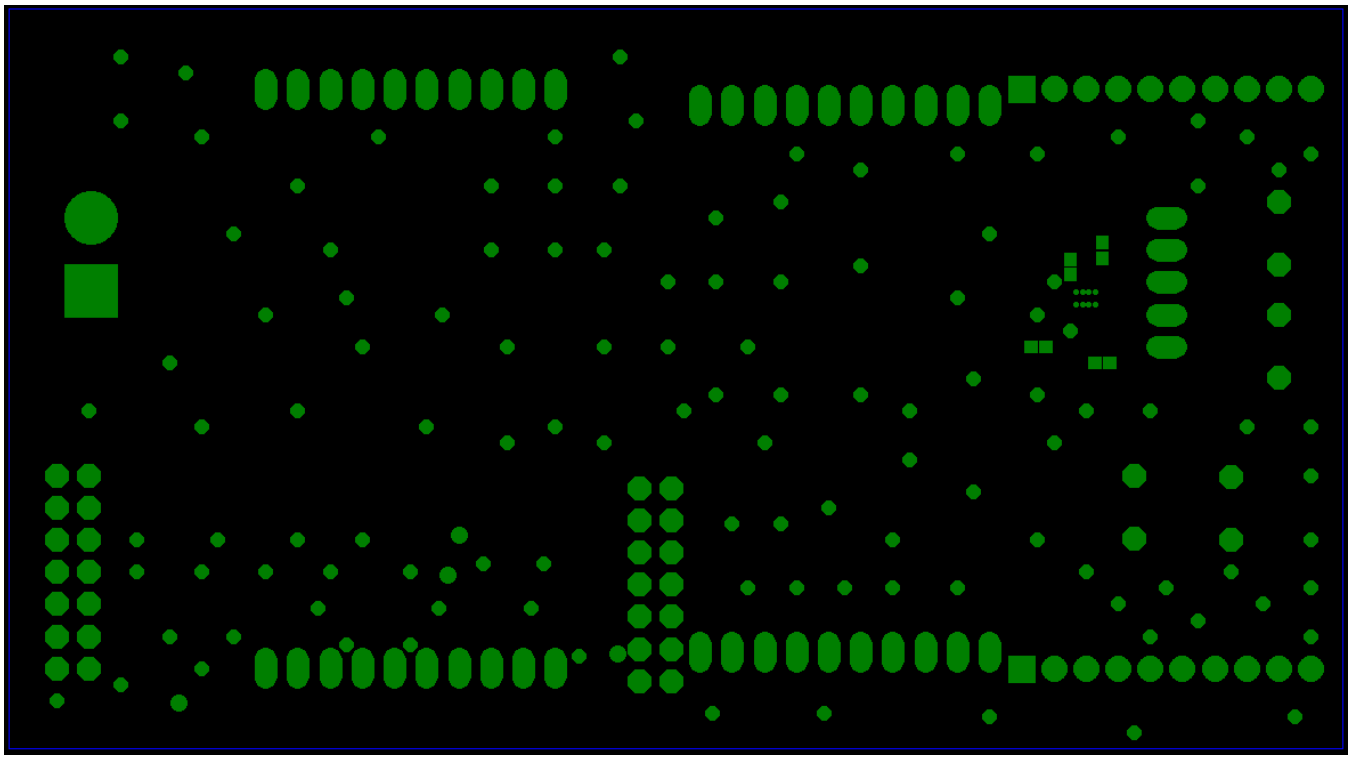
Copper Top



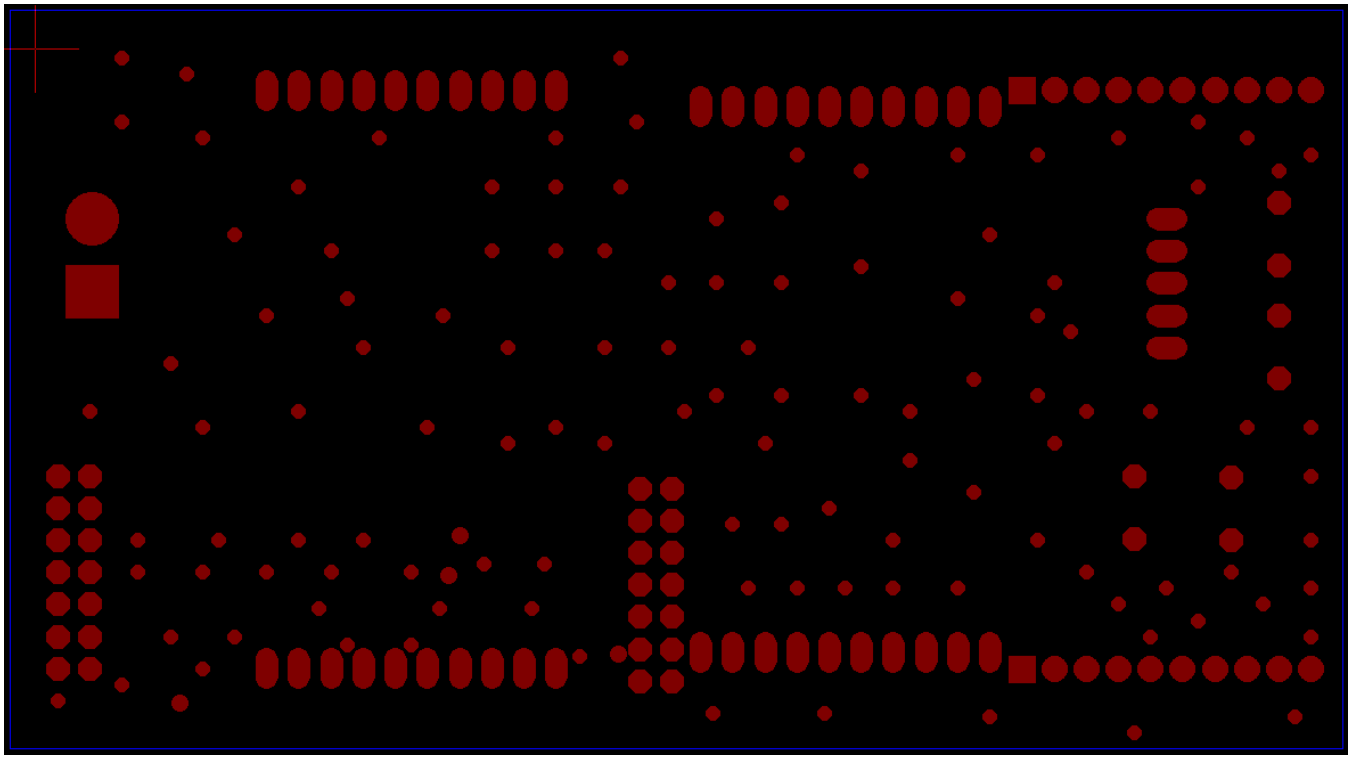
6.3.1 Copper Bottom



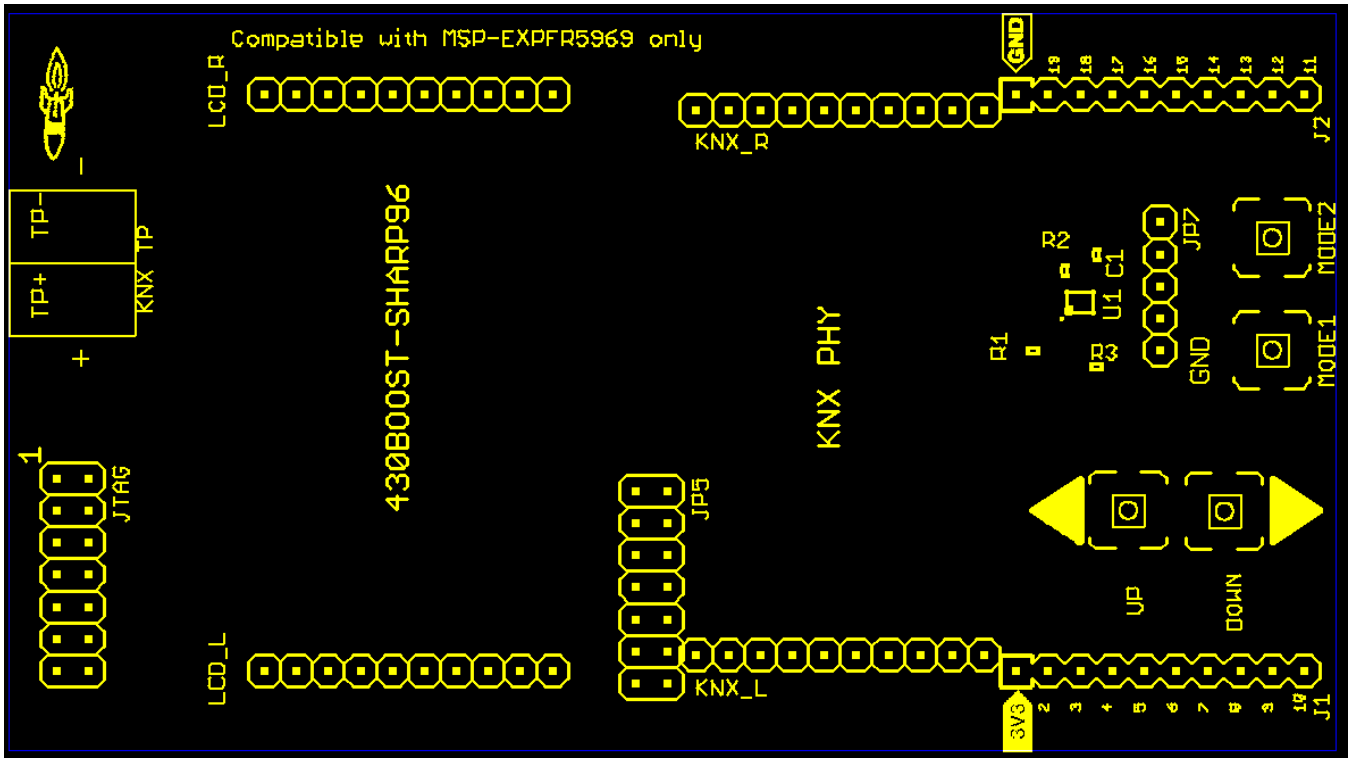
6.3.2 Soldermask Top



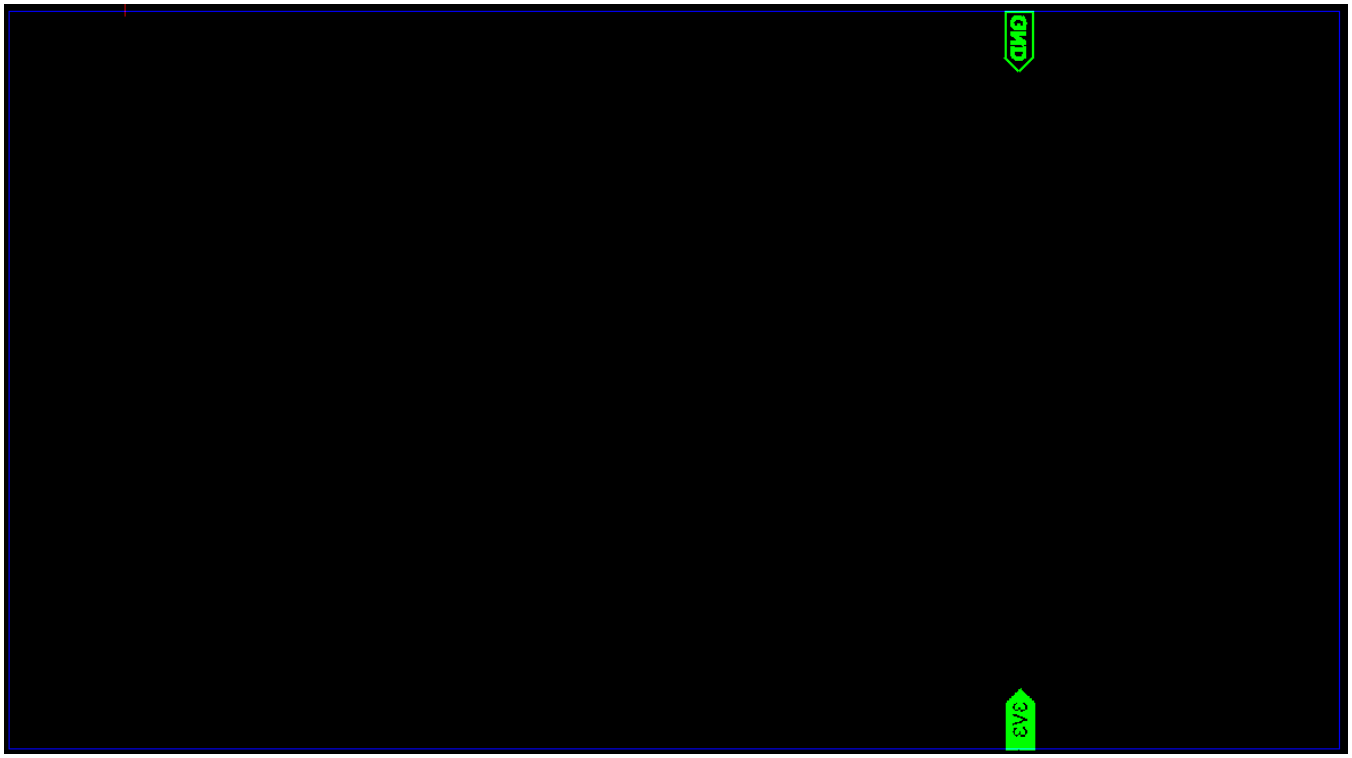
6.3.3 Soldermask Bottom



6.3.4 Silkscreen Top

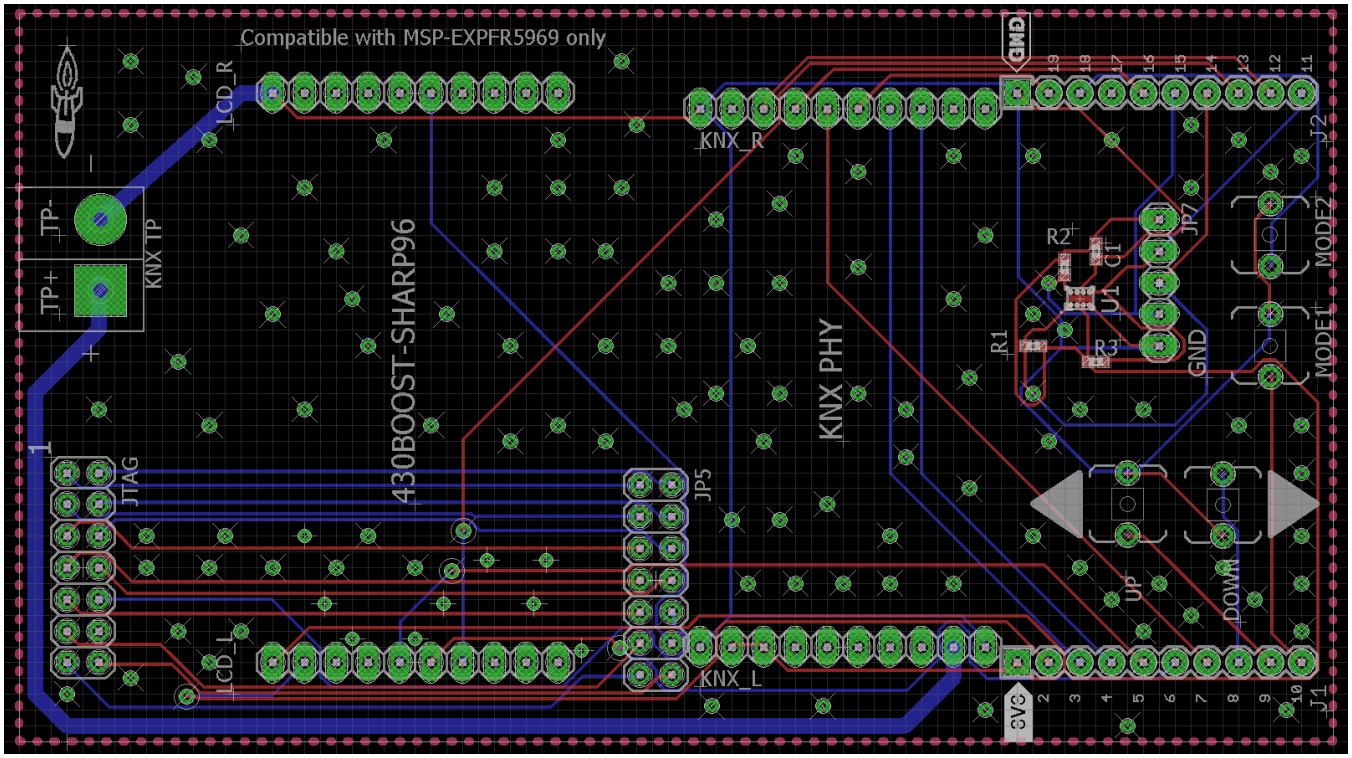


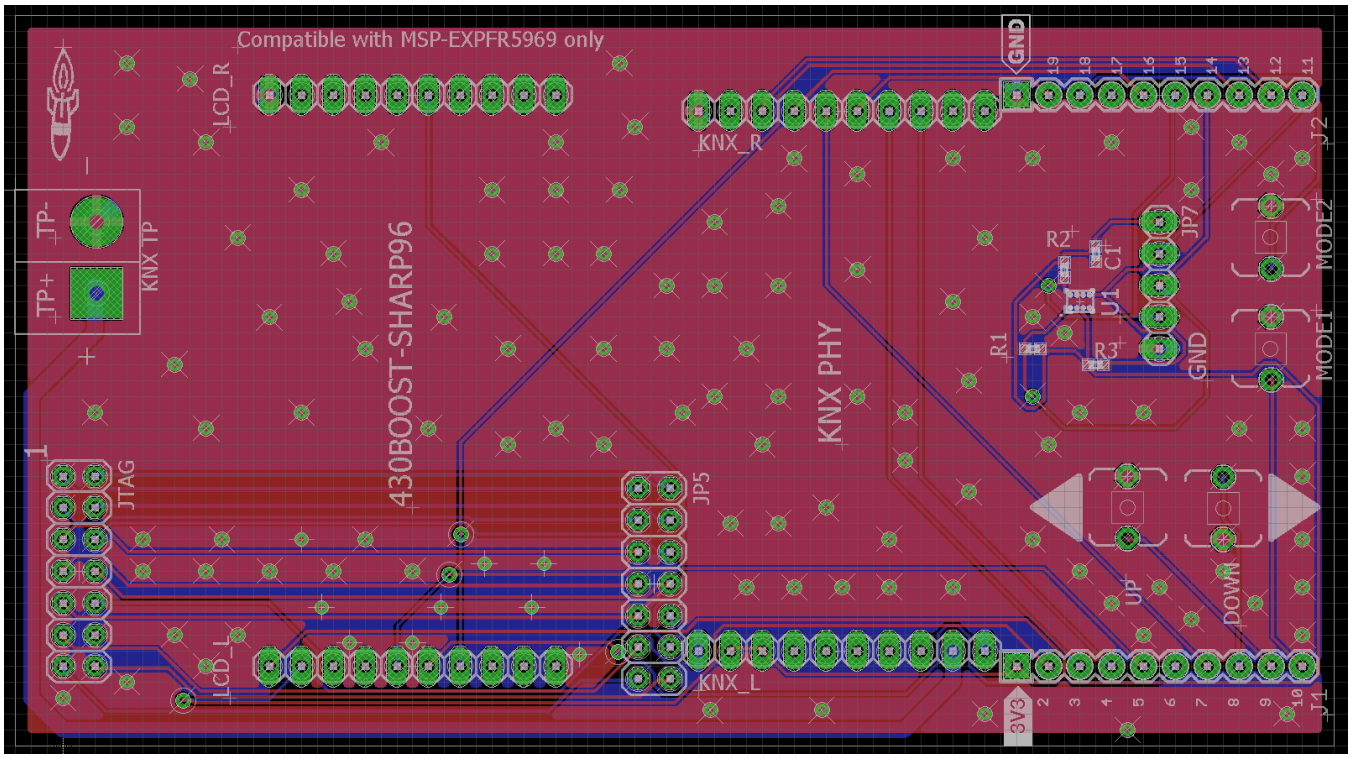
6.3.5 Silkscreen Bottom



6.4 Eagle Layout

To download the Eagle layout, see the design files at http://www.ti.com/tool/TIDM-KNXTHERMOSTAT.





7 References

- *Using MSP on KNX Systems* application report (SWRA497)

8 About the Author

DAMIAN SZMULEWICZ is a systems applications engineer at TI, where he develops reference design solutions for the industrial segment and supports building automation customers.

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Original (January 2016) to A Revision **Page**

- Updated links in Hardware Requirements for the KNX Thermostat Table. 2
- Updated links in Software Requirements for the KNX Thermostat Table. 2
- Added "when the TP-KAIphys (bit-based) PHY is chosen." after "is not used for this application" 12
- Added two notes after "Remove all jumpers from the J13 header."\ 12
- Updated Hardware Component Resources. 18
- Added note "if using the TP-UART interface,..." 20

IMPORTANT NOTICE FOR TI REFERENCE DESIGNS

Texas Instruments Incorporated ('TI") reference designs are solely intended to assist designers ("Designer(s)") who are developing systems that incorporate TI products. TI has not conducted any testing other than that specifically described in the published documentation for a particular reference design.

TI's provision of reference designs and any other technical, applications or design advice, quality characterization, reliability data or other information or services does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such reference designs or other items.

TI reserves the right to make corrections, enhancements, improvements and other changes to its reference designs and other items.

Designer understands and agrees that Designer remains responsible for using its independent analysis, evaluation and judgment in designing Designer's systems and products, and has full and exclusive responsibility to assure the safety of its products and compliance of its products (and of all TI products used in or for such Designer's products) with all applicable regulations, laws and other applicable requirements. Designer represents that, with respect to its applications, it has all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. Designer agrees that prior to using or distributing any systems that include TI products, Designer will thoroughly test such systems and the functionality of such TI products as used in such systems. Designer may not use any TI products in life-critical medical equipment unless authorized officers of the parties have executed a special contract specifically governing such use. Life-critical medical equipment is medical equipment where failure of such equipment would cause serious bodily injury or death (e.g., life support, pacemakers, defibrillators, heart pumps, neurostimulators, and implantables). Such equipment includes, without limitation, all medical devices identified by the U.S. Food and Drug Administration as Class III devices and equivalent classifications outside the U.S.

Designers are authorized to use, copy and modify any individual TI reference design only in connection with the development of end products that include the TI product(s) identified in that reference design. HOWEVER, NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of the reference design or other items described above may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI REFERENCE DESIGNS AND OTHER ITEMS DESCRIBED ABOVE ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING THE REFERENCE DESIGNS OR USE OF THE REFERENCE DESIGNS, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY DESIGNERS AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS AS DESCRIBED IN A TI REFERENCE DESIGN OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF THE REFERENCE DESIGNS OR USE OF THE REFERENCE DESIGNS, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

TI's standard terms of sale for semiconductor products (http://www.ti.com/sc/docs/stdterms.htm) apply to the sale of packaged integrated circuit products. Additional terms may apply to the use or sale of other types of TI products and services.

Designer will fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of Designer's non-compliance with the terms and provisions of this Notice.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265