Technical Article

How to Build a Smart Thermostat Using an MCU – 7 Steps to Make it Happen!

Britta Ruelander

Co-authored by Bhargavi Nisarga

How It All Started

When starting in the ultra-low-power microcontroller (MCU) business at TI, just after finishing university, I found that I had a lot of questions regarding the features and functionality of a microcontroller. I started having regular discussions with more experienced colleagues to better understand the MCU functionality and its typical usage. After a while, I realized I could share what I was learning so other MCU newbies may benefit from the information I gathered. And, here I am, joining forces with my colleague to write this blog series!

Why Another Smart Thermostat Blog?

We wanted to focus on an end-application most people were familiar with, but would want a better understanding of how to implement using an MCU. We picked the smart thermostat end-application as it touches on different levels of detail and complexity, while still addressing basic MCU features and functionality. The basic building blocks of a smart thermostat application include sensing, processing, user-interface and wireless connectivity, and designing these components requires interaction between multiple MCU features Figure 1. The subsequent blog posts as part of this series will walk you through different implementation steps to help you better understand MCU features and how their functionality enables the creation of a smart thermostat.

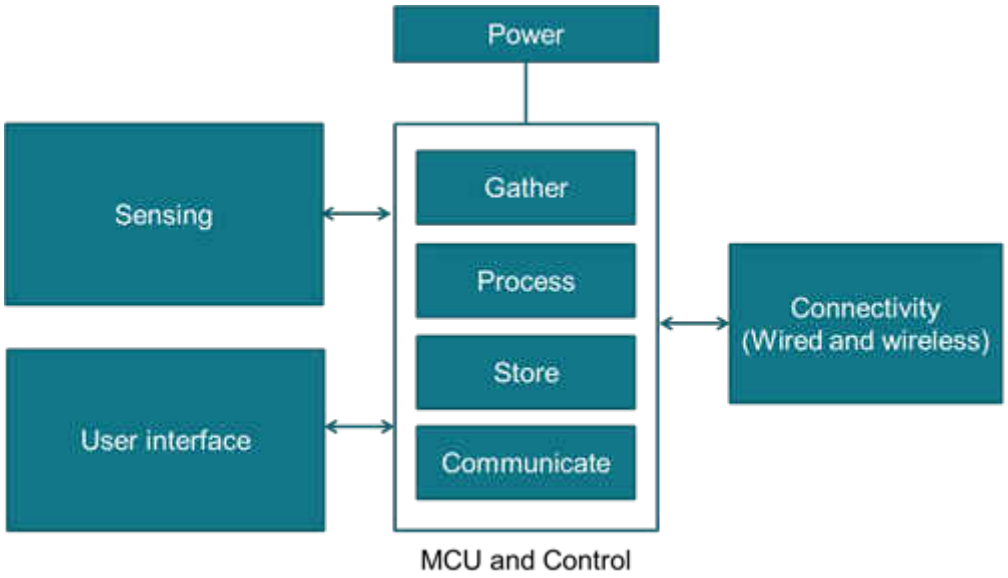

Figure 1. Building Blocks of a Smart Thermostat

Why Make a Thermostat Smart?

The thermostat is a home or building automation application that automatically regulates temperature, or activates a device when the temperature reaches a certain point. With the emergence of the Internet of Things (IoT) and network connected devices, a whole new market has opened for smart thermostats. Smart thermostats not only enable remote management and programming via mobile app or web browser, but they also learn and adapt to users' daily routine, and provide users with energy consumption data to see usage patterns and make adjustments to significantly save on energy bills.

What to Expect from the Blog Series

In order to implement a smart thermostat, you will need some background on the functionality and options an MCU offers. This series of blog posts will help you get started by covering the important implementation steps on your way to a working smart thermostat.

- The **first** blog post (this one) gives an introduction to the topic and provides you with an overview to see the whole picture before getting your hands dirty.
- In the heart of a smart thermostat is the sensing and data processing. Can you imagine a thermostat which can't sense temperature? The **second** blog post will show you some basics on sensing, realized through external sensors, analog-to-digital converters (ADC), operational amplifiers and timers.
- The **third** blog post will discuss data processing and help you take a look at some helpful MCU tools like the Direct Memory Access (DMA) Controller and hardware multiplier.
- The **fourth** blog post is all about human machine interface (HMI). How can you insert user commands to control or program the thermostat? How to interface to displays so you see collected data?
- We called it a "smart" thermostat. But until now we only covered a description of a normal and old-fashioned thermostat. You'll need two more steps for your thermostat to become smart: First adding connectivity, either wired or wireless. The other "smart factor" is that the smart thermostat behaves, works and adapts to changes on its own. Cool, isn't it? You'll learn how to do so in our **fifth** blog post.
- It's nearly time to wrap it up. You're close to your goal. But let's take a look at the underlying MCU systems, the clocking and the energy consumption. Don't you want your smart thermostat to be an ultra-low power application? And it's probably time to talk about some environmental monitoring here. So don't miss our **sixth** blog post.
- Last but not least the **seventh** post you have been longing for. You made it. You collected the information on "How to build a smart thermostat on an MCU." We'll close the loop and summarize for you. But how would you pick the right MCU? Just read the last post to get an idea on how to differentiate one from another.

Who Can Benefit from the Blog Series?

As mentioned before, this blog series is beneficial to MCU newbies, helping them to get started with MCU projects. So, if you are just starting to develop and dig deeper into MCU topics, you're totally in the right place. Even if you already are an advanced user but wondering "How can I add smartness to my application?" or "How to make my application lower power?", then you're in the right place as well. We will focus on connectivity, low power and smart device operation within the course of this blog series.

What to Take with You

As a wrap-up, here are the key takeaways of our **first** blog entry:

- This blog series intends to provide insight into basic MCU features and functionality.
- Smart thermostat is the chosen end-application for the blog series.
- The basic building blocks of a smart thermostat include sensing, user interface, power, control and connectivity blocks.
- The following six blog posts will provide step-by-step guidelines to implement a smart thermostat using an MCU.

And, don't forget to keep reading the rest of our series.

- "Step 2 to build a smart thermostat using an MCU– It's all about the sensing."
- "Step 3 to building a smart thermostat using an MCU – What to do with all the data?"
- "Step 4 to building a smart thermostat using an MCU – adding HMI"
- "Step 5 to build a smart thermostat using an MCU – adding network connectivity"
- "Step 6 to build a smart thermostat using an MCU: energy optimization"
- "Step 7 to build a smart thermostat using an MCU: how to pick the right MCU"

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

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