# Part 1: Explaining Thermostat Support for Peripherals

* **Overview of Peripherals Used:**
  + **Temperature Sensor (I2C):**
    - The thermostat uses an I2C interface to communicate with the temperature sensor, reading the ambient temperature.
    - It ensures compatibility with multiple sensors (e.g., TMP11X, TMP116) using an address-detection mechanism.
  + **LED Indicators (GPIO):**
    - GPIO is configured to drive LEDs that visually indicate whether the heater is ON (red LED) or OFF.
  + **Buttons (GPIO):**
    - GPIO interrupts detect button presses to adjust the desired temperature (set-point).
  + **UART for Cloud Connection:**
    - UART transmits thermostat state (e.g., ambient temperature, set-point, heater status) as a simulation of cloud connectivity.

**Comparison of Hardware Architectures:**

1. **TI Architecture (e.g., CS3220S):**
   * **I2C and GPIO:** Fully supported with dedicated drivers and peripheral libraries, ensuring robust performance.
   * **UART:** Supported with flexible baud rate configurations.
   * **Flash/RAM:** CS3220S typically has an optional 1MB of Flash and 256 KB RAM, sufficient for storing sensor data and running the scheduler.
   * **Resource:** [CC3230S and CC3230SF SimpleLink Wi-Fi 2.4GHz Wireless MCU with Coexistence datasheet (Rev. B)](https://www.ti.com/lit/ds/symlink/cc3230s.pdf?ts=1733782685909&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FCC3230S)
2. **Microchip Architecture (e.g., PIC32):**
   * **I2C and GPIO:** Well-supported, but might require more manual configuration compared to TI.
   * **UART:** Supported through well-documented APIs for transmitting data.
   * **Flash/RAM:** 32-512 KB Flash and 8-32 KB RAM in standard models—can be limiting for complex schedulers but sufficient for this project.
   * **Resource:** [PIC32MX Family Data Sheet](https://ww1.microchip.com/downloads/en/DeviceDoc/PIC32MX_Datasheet_v2_61143B.pdf)
3. **Freescale Architecture (e.g., Kinetis K60):**
   * **I2C and GPIO:** Freescale microcontrollers are known for efficient I2C performance and GPIO configurability.
   * **UART:** Supports high-speed UART with built-in DMA for reliable data transmission.
   * **Flash/RAM:** Typically ranges from 256-512 KB Flash and 64-128 KB RAM, offering a balance of performance for such applications.
   * **Resource:** [Kinetis K6x Fact Sheet](https://www.nxp.com/docs/en/fact-sheet/KINK6XFS.pdf)

# Part 2: Cloud Connectivity via Wi-Fi

* The thermostat connects to the cloud by transmitting data over UART to a Wi-Fi module (e.g., ESP8266 or CC3100). The module handles the actual Wi-Fi communication, abstracting the networking complexity from the thermostat.
* **TI Architecture:**
  + Integrated Wi-Fi solutions like CC3100 make TI highly compatible.
  + Example: UART sends state data to the CC3100 module, which forwards it to a cloud server.
  + **Resource:** [**CC3100 SimpleLink Wi-Fi Network Processor, Internet-of-Things Solution for MCU Applications datasheet (Rev. D)**](https://www.ti.com/lit/ds/symlink/cc3100.pdf?ts=1733783172548&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FCC3100)
* **Microchip Architecture:**
  + Compatible with external Wi-Fi modules (e.g., ATWINC1500) connected via UART.
  + **Resource**: [Atmel ATWINC1500-MR210P Datasheet](https://cdn.sparkfun.com/assets/b/1/5/8/c/ATWINC1500-MR210PB.pdf)
* **Freescale Architecture:**
  + Compatible with a variety of Wi-Fi + Bluetooth chips that NXP offers.
  + **Resource**: [Wi-Fi® + Bluetooth® | NXP Semiconductors](https://www.nxp.com/products/wireless-connectivity/wi-fi-plus-bluetooth-plus-802-15-4:WIFI-BLUETOOTH)

# Part 3: Flash and RAM Discussion

* The thermostat code is lightweight and primarily consists of:
  + Scheduler logic.
  + Drivers for I2C, GPIO, and UART.
  + Simple state machine for heater control.
  + Flash usage includes storing code and static variables, while RAM handles runtime variables like ambient\_temp and set\_point.
* Usage of RAM by Program:
  + 63,089 KB/246,000 KB of SRAM (RAM)
  + 4876 KB/16,384 KB of SRAM2 (Secondary RAM)
* **Requirements Across Architectures:**
  + All architectures except for Microchip provide sufficient Flash and RAM to support the program. TI's CS3220S and Freescale Kinetis offer sufficient memory, with Freescale Kinetis providing a lot more scaling potential.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Architecture | Flash Capacity | RAM Capacity | Key Strengths | Potential Limitations |
| TI (e.g., CS3220S) | Up to 1 MB | 256 KB | Fully supported I2C and GPIO drivers, robust peripheral libraries, and integrated Wi-Fi solutions like CC3100. | Moderate Flash and RAM capacity may limit advanced or highly scalable features. |
| Microchip (e.g., PIC32) | 32–512 KB | 8–32 KB | Well-documented APIs for UART, suitable for basic thermostat functionality. | Limited RAM capacity requires careful memory management for dynamic tasks; less scaling potential. |
| Freescale (e.g., Kinetis K60) | 256–512 KB | 64–128 KB | Efficient I2C performance, high-speed UART with DMA support, and ample memory for scaling advanced features. | Higher power consumption compared to TI or Microchip; might be overkill for simpler projects. |