# **ATTINY85 AND ATMEGA328P**

## **ATTINY85 MICROCONTROLLER**

**OVERVIEW**  
The [ATtiny85](https://www.allelcoelec.com/search.html?QueryField=ATTINY85) microcontroller is designed with a compact structure that combines efficient performance and power-saving capabilities. Known for its small size, this 8-bit microcontroller can manage various functions with minimal components, making it popular for both small-scale hobby projects and more demanding industrial applications. It has a program memory of 8K, allowing you to store applications efficiently while saving space. The ATTINY85 also includes multiple power-saving modes, enabling it to run efficiently on battery-operated devices. Its flexible range of input and output options further supports versatile usage, making it adaptable for a wide array of embedded system tasks. This compact, cost-effective design allows it to deliver solid performance without compromising on essential microcontroller features, offering an easy-to-use solution for developers and experimenters alike.

**DETAILS OF ATTINY85 MICROCONTROLLER**

**1. Compact and Affordable:** The ATTINY85 is known for its small size and affordability, making it a great choice for both hobby projects and professional applications. It provides essential features without taking up much space or stretching your budget.

**2. Adequate Memory:** With 8K of program memory, the ATTINY85 offers enough space for many applications. This allows you to store code and execute a range of tasks without needing additional components or storage.

**3. Power-Saving Modes:** The ATTINY85 includes several power-saving modes, which help reduce power consumption in battery-operated devices. These modes allow it to run efficiently, prolonging battery life for mobile or remote projects. These power-saving modes include:

1. **Idle Mode**: In Idle mode, the CPU stops executing instructions, but all the peripherals like timers, SPI, and ADC can keep running. This mode is useful when the microcontroller is waiting for a peripheral event (like a timer overflow or data reception) but doesn’t need to actively process data. Since the system clock is still running, the ATtiny85 can wake up very quickly from this mode.
2. **ADC Noise Reduction Mode:** This mode is designed to improve the accuracy of analog-to-digital conversions. It shuts down the CPU and most digital peripherals, which reduces electrical noise that can interfere with analog signals. Only the ADC and asynchronous interrupts remain active. It’s ideal for applications requiring precise analog readings without interference from other system components.
3. **Power-down Mode:** Power-down mode is the most energy-efficient sleep state. In this mode, the CPU and all internal peripherals are completely turned off. The only components that can wake the microcontroller are external interrupts (e.g., a button press), pin change interrupts, or the watchdog timer. This mode is perfect for long-term sleeping when the ATtiny85 doesn't need to do anything for extended periods.
4. **Power-save Mode:** This mode is similar to Power-down, but it keeps the asynchronous timer (Timer/Counter2) running. This allows the device to maintain a real-time clock or time-based wake-up functionality without needing an external timer circuit. It is ideal for applications that require periodic activity, such as data logging or timed sensor checks.
5. **Standby Mode:** Standby mode is like Power-down mode but with the main system oscillator kept running. This allows the ATtiny85 to wake up more quickly, as it doesn’t need to wait for the oscillator to start. It’s useful when low power consumption is still important, but a faster wake-up time is also needed, such as in time-sensitive applications that use external crystals.

**4. Flexible I/O Options:** Equipped with multiple input/output options, the ATTINY85 can handle various tasks and interact with different peripherals. This flexibility makes it adaptable for many types of embedded systems.

**5. Built-in Watchdog Timer:** The ATTINY85 comes with a watchdog timer, which helps prevent the microcontroller from hanging or freezing. This feature increases reliability, especially for applications that need consistent and stable performance.

**ATTINY85 FEATURES/ ADVANTAGES**

1. **High Performance and Low Power:** The ATTINY85 operates as a high-performance microcontroller while using low power. Its compact 8-bit AVR design allows it to handle tasks efficiently without draining much energy, making it suitable for various applications.
2. **Advanced RISC Architecture:** With its RISC architecture, the ATTINY85 can execute 120 powerful instructions, most within a single clock cycle. It includes 32 x 8 general-purpose working registers, allowing for quick and smooth operations and fully static operation.
3. **Reliable Memory Options:** The microcontroller offers 8K bytes of in-system programmable flash memory, supporting up to 10,000 write/erase cycles. It also includes 512 bytes of in-system programmable EEPROM (up to 100,000 write/erase cycles) and 512 bytes of internal SRAM, making it suitable for a variety of data storage needs.
4. **Strong Data Protection:** The ATTINY85 includes a programming lock to protect your stored data and code, adding an extra layer of data security in memory.
5. **Versatile Peripheral Features:** Equipped with an 8-bit timer/counter with a prescaler, two PWM channels, and a high-speed timer, the ATTINY85 handles timing and pulse-width modulation tasks smoothly. It also supports a universal serial interface (USI) with a start condition detector and has a 10-bit ADC for handling analog signals.
6. **Enhanced ADC Options:** The 10-bit ADC offers four single-ended channels and two differential pairs with programmable gain, making it suitable for analog signal measurement. Temperature measurement is also possible, adding flexibility to its capabilities.
7. **Built-in Watchdog Timer:** The ATTINY85 includes a watchdog timer with a separate on-chip oscillator to keep it running consistently, helping prevent system errors from causing it to freeze or stall.
8. **On-chip Analog Comparator:** With an on-chip analog comparator, the ATTINY85 can quickly compare analog input signals, which is useful for applications requiring signal analysis or comparisons.
9. **Special Microcontroller Features:** The ATTINY85 offers debug WIRE for easy debugging, in-system programming via SPI port, and both external and internal interrupt sources. It also includes low-power idle modes, ADC noise reduction, and power-down modes to manage power efficiently.
10. **Enhanced Power and Voltage Support:** The enhanced power-on reset circuit and programmable brown-out detection circuit ensure reliability in varying power conditions. An internal calibrated oscillator provides stable operation without needing an external oscillator.
11. **Programmable I/O Lines:** With six programmable input/output lines, the ATTINY85 provides flexibility in how you connect it to other devices or components, allowing more versatile setups.
12. **Wide Operating Voltage Range:** Operating between 2.7 to 5.5V, the ATTINY85 can handle a variety of voltage inputs, making it compatible with different power sources.
13. **Speed and Temperature Compatibility:** The microcontroller performs efficiently across a range of speeds, reaching up to 20 MHz at higher voltages. It also operates within an industrial temperature range, from -40°C to 85°C, making it suitable for diverse environments.
14. **Low Power Consumption:** In active mode, the ATTINY85 draws just 300 µA at 1 MHz, 1.8V. In power-down mode, it reduces consumption to 0.1 µA at 1.8V, making it an excellent choice for battery-powered devices and power-sensitive applications.

**DISADVANTAGES**

1. **Limited I/O Pins:** The ATtiny85 comes in an 8-pin package, out of which only 6 pins are available for general-purpose input/output (GPIO). If you use one of those pins for resetting (which is the default behavior of pin PB5), you’re left with just 5 usable I/O pins. This significantly limits the number of external components (like sensors, LEDs, or actuators) that can be connected directly to the chip. Although creative multiplexing or communication protocols like I2C can help expand its functionality, the limited number of pins remains a significant constraint in more complex designs.
2. **No Hardware UART:** Unlike more advanced microcontrollers (like the ATmega328P used in Arduino Uno), the ATtiny85 does not have a built-in UART (Universal Asynchronous Receiver/Transmitter) for serial communication. While it is possible to implement UART through software, often called “SoftwareSerial”, it’s not as reliable or efficient, especially at higher baud rates. This limitation makes debugging or communicating with other serial devices more challenging and less robust.
3. **No Native USB Support:** The ATtiny85 does not come with built-in USB support. However, versions like Digispark boards use a custom bootloader to emulate USB, allowing you to program or use the microcontroller as a basic USB device. This workaround is functional but limited. It can only simulate certain USB devices and doesn't provide full USB stack support. If your project requires native USB connectivity (like HID, MIDI, or mass storage), the ATtiny85 would not be an ideal choice without significant compromises.
4. **Small Memory Capacity:** The microcontroller has only 8 KB of flash memory for code, 512 bytes of SRAM, and 512 bytes of EEPROM. This is sufficient for simple programs but quickly becomes a limitation as your project grows in complexity. Libraries, especially those with graphical or network functionality, often take up more memory than the ATtiny85 can handle. Developers need to write highly optimized, minimal code and be cautious with data structures and variable usage.
5. **No On-Chip Debugging Tools:** Unlike higher-end microcontrollers that come with dedicated debugging interfaces (such as JTAG or SWD), the ATtiny85 lacks native debugging hardware. This makes it harder to step through code, set breakpoints, or inspect memory during runtime. Debugging is usually done via blinking LEDs, serial prints (if SoftwareSerial is implemented), or simulation tools. For beginners or complex development workflows, this limitation can make troubleshooting time-consuming and less precise.
6. **Limited Library and Peripheral Support:** While the ATtiny85 can be programmed through the Arduino IDE, not all Arduino libraries are compatible due to its smaller memory, lack of certain hardware features (like UART or advanced timers), and architectural differences. Some functions and libraries may need to be rewritten or trimmed to fit within the constraints. This can be an obstacle, especially for users relying on plug-and-play code or advanced peripherals like servo controllers, displays, or networking modules.

**PROGRAMMING ATTINY85 MICROCONTROLLER**

Programming the ATTINY85 involves several simple steps to get your code ready and running on the device. Below is a quick guide on the programming of this microcontroller:

1. First, outline what tasks you want the microcontroller to perform. Listing these tasks will give you a clear direction as you develop your code.
2. Next, write the functions or commands that the microcontroller will need to perform the tasks you listed. Using an Integrated Development Environment (IDE) can make this process easier.
3. After writing the functions, compile your code in the IDE to check for any errors. This step ensures that the code will run smoothly when transferred to the microcontroller.
4. Once compiled, the IDE will produce a HEX file, which is the format needed for the ATTINY85 to understand and execute the code. This file contains the machine instructions that will be stored on the microcontroller.
5. Select a programming device, like an SPI programmer, which will connect your computer to the ATTINY85. This programmer allows you to transfer the HEX file directly to the microcontroller’s memory.
6. Using the programmer’s software, select the HEX file you generated and upload it to the ATTINY85. This step installs your code on the device, preparing it to execute as soon as it’s powered.
7. Finally, disconnect the programmer and connect any peripherals your application needs. Once powered, the ATTINY85 will begin executing the code you've loaded, completing the setup process.

**ATTINY85 APPLICATIONS**

1. **Development Boards and Hobby Projects:** The ATTINY85 is widely used in development boards and DIY electronics projects due to its compact size and flexibility. It’s a great choice for anyone looking to create custom gadgets or automate simple tasks at home.
2. **Industrial Control Systems:** In industrial settings, the ATTINY85 can manage control systems where space and power efficiency are needed. Its stability and reliability make it suitable for basic control tasks in machinery or automated processes.
3. **Power Regulation and SMPS Systems:** The ATTINY85 is commonly used in power regulation and switch-mode power supply (SMPS) systems. Its ability to control and monitor power makes it ideal for devices that require consistent power management.
4. **Analog Signal Measurement and Manipulation:** With its built-in ADC (analog-to-digital converter), the ATTINY85 can measure and manipulate analog signals. This feature is useful in applications that require data collection from sensors or other analog sources.
5. **Embedded Systems in Appliances:** The ATTINY85’s compact design allows it to be embedded in appliances like coffee machines and vending machines, where it can manage basic operations or interface with user controls.
6. **Display Units:** The microcontroller can also be used to drive display units, controlling LEDs or small screens in devices where you need a simple display without complex hardware requirements.
7. **Peripheral Interface Systems:** The ATTINY85 is often used as an interface between different peripherals, allowing various components in a system to communicate effectively. This application is common in both consumer electronics and custom-designed circuits.

# **2. ATMEGA328P MICROCONTROLLER**

**OVERVIEW**

The ATmega328P is a highly popular 8-bit AVR microcontroller manufactured by Microchip Technology. It's widely known as the microcontroller at the heart of the Arduino Uno development board, which has made it a favorite in education, DIY projects, and prototyping. The chip offers a great balance between power, memory, and features, making it suitable for a variety of applications. Whether you're building a simple blinking LED project or a more complex sensor-based system, the ATmega328P provides a reliable and versatile platform for development. With support for digital and analog inputs, communication interfaces, and power-saving modes, it's ideal for both battery-powered and plug-in devices.

**DETAILS OF ATMEGA328P MICROCONTROLLER**

1. **User-Friendly for Beginners and Professionals**: The ATmega328P is easy to use, especially through the Arduino platform. It requires minimal setup and is supported by thousands of tutorials and online examples. Students can start coding and testing their ideas without deep technical knowledge of hardware.
2. **Plenty of Memory for Embedded Projects**: With 32 KB of flash memory for storing program code, 2 KB of SRAM for running variables and buffers, and 1 KB of EEPROM for storing permanent data, the ATmega328P provides enough memory for most small-to-medium embedded applications.
3. **Wide Range of Communication Interfaces**: The ATmega328P supports UART, SPI, and I2C interfaces, allowing it to communicate with various external devices like sensors, displays, memory cards, wireless modules, and other microcontrollers.
4. **Versatile Input and Output Pins**: It offers up to 23 digital I/O pins, 6 analog input channels, and 6 PWM outputs, allowing the chip to interact with LEDs, motors, sensors, and many other components used in real-world embedded systems.
5. **Built-in Timers and PWM Support**: The microcontroller includes two 8-bit timers and one 16-bit timer for precise time-based operations. These are useful for scheduling tasks, generating delays, or creating accurate PWM signals for motor control and dimming LEDs.
6. **Energy-Efficient Operation with Sleep Modes**: The ATmega328P supports multiple sleep modes like Idle, Power-down, Power-save, and ADC Noise Reduction, which make it suitable for low-power and battery-powered applications. It can wake up on interrupts, ensuring efficiency without sacrificing performance.
7. **Reliable in Harsh Environments**: It operates across a wide temperature range (-40°C to 85°C) and voltage range (1.8V to 5.5V), making it robust enough for use in industrial and outdoor applications.

**ATMEGA328P FEATURES / ADVANTAGES**

1. **Beginner-Friendly and Well-Supported**  
   One of the biggest strengths of the ATmega328P is how easy it is to get started with. Through the Arduino IDE, users can program it in simple C++-like code without needing to understand complex register-level programming. The large Arduino community ensures a wealth of libraries, sample codes, and project guides that make development much smoother for students and hobbyists.
2. **Adequate Memory for Practical Projects**  
   With 32 KB of flash memory, most typical embedded programs can be written without running out of space. This includes room for sensor-reading, control logic, and even simple user interfaces. The 2 KB of SRAM allows for enough variables, sensor readings, and buffers for tasks like serial communication, while the 1 KB of EEPROM is perfect for storing settings or logs even when the device loses power.
3. **Multiple Digital and Analog Interfaces**  
   The ATmega328P includes a mix of digital and analog I/O, making it suitable for a wide variety of applications. It can read sensor values using its 10-bit ADC, output analog-like signals with PWM, and control devices such as relays, LEDs, and buzzers. This flexibility allows the microcontroller to be at the core of many different systems, from data loggers to control panels.
4. **Strong Communication Capabilities**  
   With SPI, I2C, and UART support, it is easy to connect the ATmega328P to external memory, displays, sensors, or wireless modules like the HC-05 Bluetooth or ESP8266 Wi-Fi. These communication features make it perfect for applications that require networking or real-time data sharing.
5. **Integrated Timers and PWM Outputs**  
   Timers are essential for creating delays, measuring time, or generating square waves. The ATmega328P has three hardware timers that support multiple features including Pulse Width Modulation (PWM), which is used for applications such as motor control, sound generation, or dimming LEDs.
6. **Power-Efficient and Battery-Friendly**  
   The microcontroller supports various power-saving modes that let it sleep when idle. For example, in Power-down mode, it draws almost no current and wakes up only through external or watchdog interrupts. This is especially useful in battery-operated systems where conserving power is essential, such as remote sensors or wearable devices.
7. **Stable and Durable Operation**  
   The ATmega328P is built to handle a wide range of conditions. It includes built-in safety features like a watchdog timer that resets the system if it hangs, a brown-out detector that prevents errors during voltage drops, and a programmable oscillator for consistent timing.
8. **Cost-Effective and Easily Available**  
   Due to its widespread use, the ATmega328P is both affordable and easy to find. It's included in many beginner kits and development boards, such as the Arduino Uno, which adds USB programming and power regulation features to make it even more accessible.
9. **Highly Documented and Supported**  
   There’s a vast amount of official documentation, user guides, datasheets, and third-party tutorials available. This makes it easy for learners and developers to understand the chip, debug their projects, and extend functionality with confidence.

**DISADVANTAGES**

1. **Limited RAM**  
   With only 2 KB of SRAM, the ATmega328P cannot handle large data buffers or memory-heavy applications. Programs need to be memory-efficient.
2. **No Built-in USB Support**  
   The chip does not have native USB connectivity, unlike the ATmega32U4. To communicate over USB, it requires an external USB-to-serial converter.
3. **No Built-in Wireless Communication**  
   Unlike newer microcontrollers (like the ESP32), it doesn’t have Wi-Fi or Bluetooth. You must connect external modules, which adds complexity and cost.
4. **8-bit Architecture**  
   While it's suitable for many control tasks, it’s not good for high-precision calculations or applications that require floating-point operations or large number processing.
5. **No Hardware Multitasking**  
   It can only execute one instruction at a time in a single-threaded loop. Advanced real-time multitasking is not possible without external software structures or RTOS-like designs.
6. **Slower Compared to Modern Chips**  
   With a top speed of 20 MHz, it is slower than newer 32-bit chips, limiting its use in high-performance embedded systems.

**PROGRAMMING ATMEGA328P MICROCONTROLLER**

Programming the ATmega328P is straightforward, especially using the Arduino platform:

1. **Plan the Application**: List the tasks the microcontroller will perform (e.g., read sensor, blink LED).
2. **Write the Code**: Use the Arduino IDE to write and test the code. Use built-in functions and libraries.
3. **Compile the Code**: The IDE checks for errors and converts the code into a HEX file.
4. **Upload the Code**: Use a USB cable (if using Arduino Uno) or a programmer (like USBasp or Arduino as ISP) to upload the code to the microcontroller.
5. **Run the Program**: Once uploaded and powered, the ATmega328P starts executing the code and interacting with peripherals.

**ATMEGA328P APPLICATIONS**

1. **Arduino Development Boards**: The most common use, powering Arduino Uno and Nano.
2. **Educational Projects**: Used in classrooms to teach programming and electronics.
3. **Automation Systems**: Controls lighting, fans, alarms, or smart devices.
4. **Data Loggers**: Collects data from sensors for analysis or storage.
5. **Robotics**: Manages motors, sensors, and decision-making in robots.
6. **Consumer Electronics**: Embedded in home gadgets, appliances, and toys.
7. **Wearables and Portable Devices**: Used in battery-efficient designs due to its power-saving modes.

# **CHOOSING BETWEEN ATMEGA328P AND ATTINY85 FOR A HOBBY SATELLITE BENCH MODEL**

In designing a **hobby satellite system**, especially a **bench model** that simulates a real satellite’s subsystems, it’s important to select microcontrollers based on the specific roles they will play. A typical satellite system includes modules such as a central flight controller, sensor interfaces, communication links, power management, and possibly actuators (like motors or servos). Both the **ATmega328P** and **ATtiny85** are 8-bit microcontrollers, but they are quite different in terms of capability and are best suited for different tasks within the system.

1. **THE MAIN CONTROL SYSTEM (ATMEGA328P)**

The **ATmega328P** is best suited to act as the **central control unit** or “brain” of the satellite system. This is because it has more memory (32 KB of flash and 2 KB of SRAM), allowing it to handle more complex logic, such as reading sensor data, executing commands, logging information, and communicating with the ground station. It supports **multiple communication protocols**, like **UART, SPI, and I2C**, making it compatible with a wide range of sensors and modules, such as GPS, gyroscopes, IMUs, and radios.

In addition, the ATmega328P offers a greater number of I/O pins (23 in total), which is essential when interfacing with multiple components. It also features built-in **timers and PWM outputs**, which are useful for simulating or controlling actuators like motors or servo-based antenna alignment systems. Because this chip is used in **Arduino Uno** boards, it is supported by the Arduino IDE and thousands of libraries and examples online. This makes development, debugging, and integration much easier, especially for students working on a prototype or learning project.

For example, in our bench model, the ATmega328P can manage the main logic—such as reading telemetry from sensors, making decisions, and simulating a data transmission to a base station via serial or radio. It can also manage time-based tasks like data logging and checking system status periodically.

1. **SUPPORT ROLES (ATTINY85)**

The **ATtiny85**, on the other hand, is a much smaller and more limited microcontroller. It has **only 8 KB of flash** and **512 bytes of SRAM**, with just **6 I/O pins**. While this makes it unsuitable for a central controller role, it is **perfect for handling small, specific tasks** as a **support microcontroller**. Its small size and low power consumption make it ideal for modules that do not require many peripherals or complex logic.

For instance, the ATtiny85 could be used to control a **power management module**, monitoring battery voltage using its ADC, switching between power sources, or turning on indicators (like LEDs) to show power status. Another good use case is as a dedicated **sensor interface** for a single analog sensor, such as a temperature sensor or light sensor, where it simply reads the value and reports it to the main controller. Because the ATtiny85 supports sleep modes and uses very little power in standby, it’s also ideal for **low-power background tasks**.

Even though the ATtiny85 does not have a built-in UART, it can communicate using **SPI or I2C**. In our bench model, we could use I2C to let the ATtiny85 send periodic battery status updates to the ATmega328P, or use its PWM output to drive a buzzer or status LED based on simple thresholds.

**COMPARISON AND RECOMMENDATION**

In summary, the **ATmega328P** is the better choice for the **core control system** of the satellite bench model because it has more memory, more I/O pins, stronger communication options, and greater flexibility for multitasking. It is the ideal candidate for handling sensor data processing, actuator control, data logging, and communication with a PC or base station.

Meanwhile, the **ATtiny85** shines in **supporting roles**, where minimal input/output and low power consumption are important. It is perfect for dedicated monitoring tasks, handling individual components, or acting as a secondary controller for modules like power systems or simple alert mechanisms.

By using both microcontrollers together, we can divide tasks efficiently: let the **ATmega328P focus on system logic and communication**, while the **ATtiny85 manages background tasks and peripheral monitoring**. This separation of roles leads to a more organized, efficient, and modular satellite system design.