Timers in Microcontrollers

Precise timing is crucial for complex microcontroller projects, such as an alarm clock, where events like second counting and LED blinking must occur at specific intervals. Events like moving figures on an LED matrix or generating PWM signals for alarms require timers to ensure accurate timing. Various microcontrollers, including MSP430 and Arduino Uno, feature built-in timers that facilitate the creation of precisely timed events. A simple Arduino sketch demonstrates basic timer functionality by blinking an LED every second. Using delay functions can lead to input commands being ignored due to the microcontroller being stuck in the delay loop. Delay functions may drift over time, causing inaccuracies in timing; thus, utilizing hardware timers is essential.

The 16-bit Timer 1 offers various features and operates independently from the main code loop once set up.

To activate Timer 1's normal mode, specific bits (WGM 13 to 10) are set to zero while configuring the prescaler bit (CS10).

The counter increments with each clock cycle until it reaches its maximum value (65535), after which it overflows back to zero.

An overflow flag is set when the counter overflows, allowing interrupts to be triggered based on this event.

By using different prescalers, such as setting CS12 instead of CS10, one can adjust overflow times for more precise control over timing events.

For example, a prescaler of 256 results in an overflow time of around 1.04 seconds.

To create multiple timed interrupts (e.g., at quarter-second and half-second intervals), CTC mode can be utilized by setting WGM12 bits appropriately.

In CTC mode, compare registers (OCR1A and OCR1B) allow for independent comparison against the counter value to trigger interrupts when they match.