Ćwiczenie 3: Regulacja jasności LED za pomocą timera

Instrukcja laboratorium

Mariusz Chilmon <mariusz.chilmon@ctm.gdynia.pl>

2024-01-08

> Fortune favors the prepared mind.
>
> — *Louis Pasteur*

Cel ćwiczenia

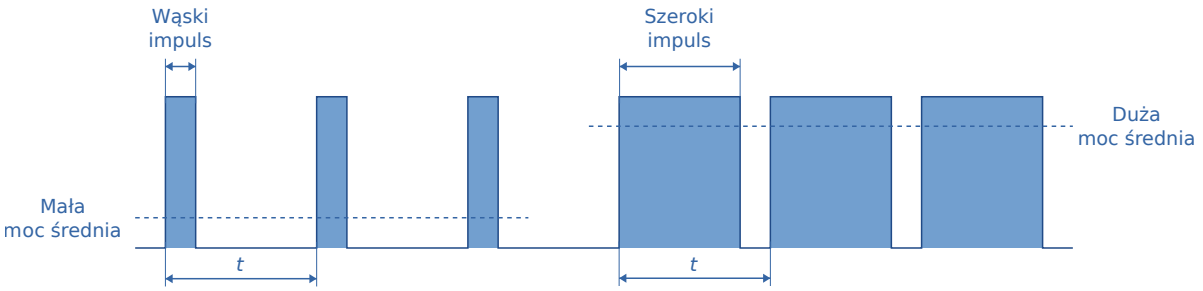
Celem ćwiczenia jest zapoznanie się z:

- regulacją za pomocą modulacji szerokości impulsu,
- wykorzystaniem timera jako źródła sygnału PWM,
- odmierzaniem czasu za pomocą timera.

Uruchomienie programu wyjściowego

1. Podłącz płytkę *WPSH209* do *Arduino Uno*.
2. Zweryfikuj, czy dioda `D1` świeci maksymalną jasnością.
3. Zweryfikuj, czy dioda `D3` mruga słabym światłem.

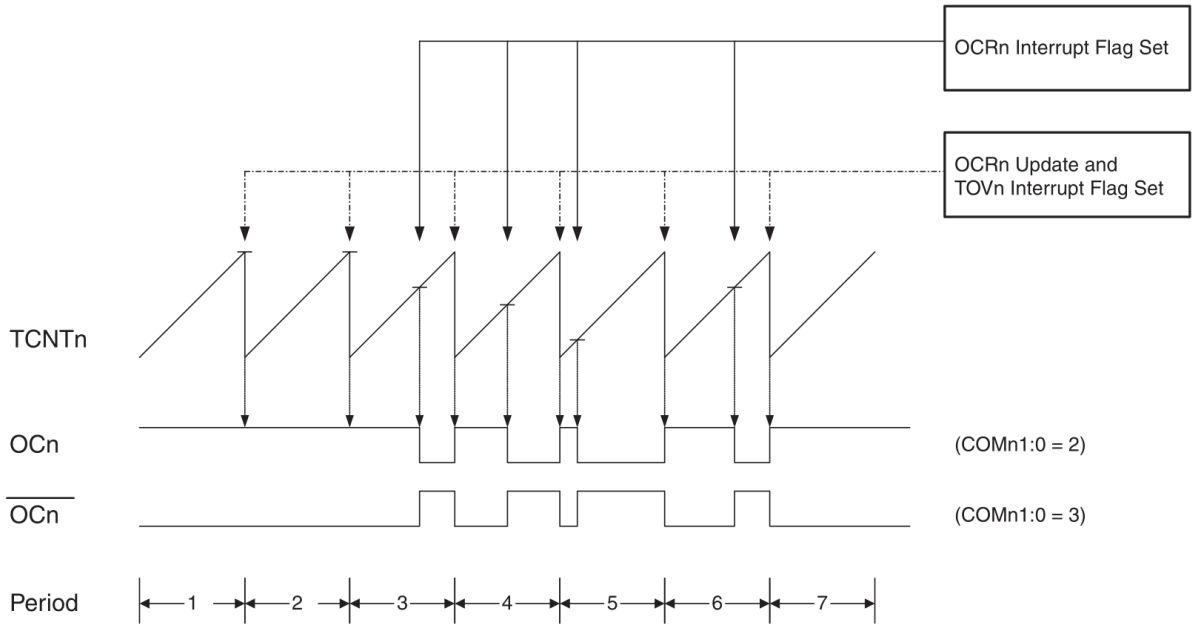
Dioda `D3` podłączona jest do pinu `PB3`, którego alternatywną funkcją jest `OC2A` (*Timer/Counter2 Output Compare Match A Output*), co oznacza, że jest sterowany zdarzeniem generewanym przez kanał *A* peryferiału *Timer/Counter2*. Timer ten został skonfigurowany w trybie *Fast PWM*. Jest to tryb, w którym możliwe jest regulowanie mocy dostarczanej do urządzenia wyjściowego (np. LED) poprzez szybkie włączanie i wyłączanie zasilania. Zmieniając proporcję między czasem włączenia i wyłączenia możemy dostarczyć mniej lub więcej mocy w jednostce czasu. W naszym przypadku wpływa to na jasność diody `D3`.



Rysunek 1: Regulacja mocy za pomocą modulacji szerokości impulsu

Licznik `TCNT` (*Timer/Counter Register*) timera w trybie *Fast PWM* zmienia się od wartości `BOTTOM` (`0x00`) do wartości `TOP` (`0xFF`). Przy wartości `0x00` wyjście `OC2A` załącza sterowane urządzenie,

a przy zrównaniu licznika z zaprogramowaną wartością OCR2A (*Output Compare Register A*) — wyłącza je. Wartość wpisana do rejestru OCR2A pozwala zatem wprost regulować moc dostarczaną do sterowanego urządzenia bez angażowania CPU.



Rysunek 2: Licznik pracujący w trybie *Fast PWM*

Timer/Counter2 może generować dwa sygnały na wyjściach OC2A i OC2B, sterowanych — odpowiednio — wartościami rejestrów OCR2A i OCR2B. W tym zadaniu korzystamy tylko z kanału *A* — wyjścia OC2A i sterującego nim rejestru OCR2A.

Mikrokontrolery AVR obok trybut *Fast PWM* udostępniają także tryb *Phase Correct PWM*, w którym uzyskiwana częstotliwość sygnału jest mniejsza, ale sposób synchronizacji impulsów jest korzystniejszy dla sterowania silników.

Zadanie podstawowe

Wymagania funkcjonalne

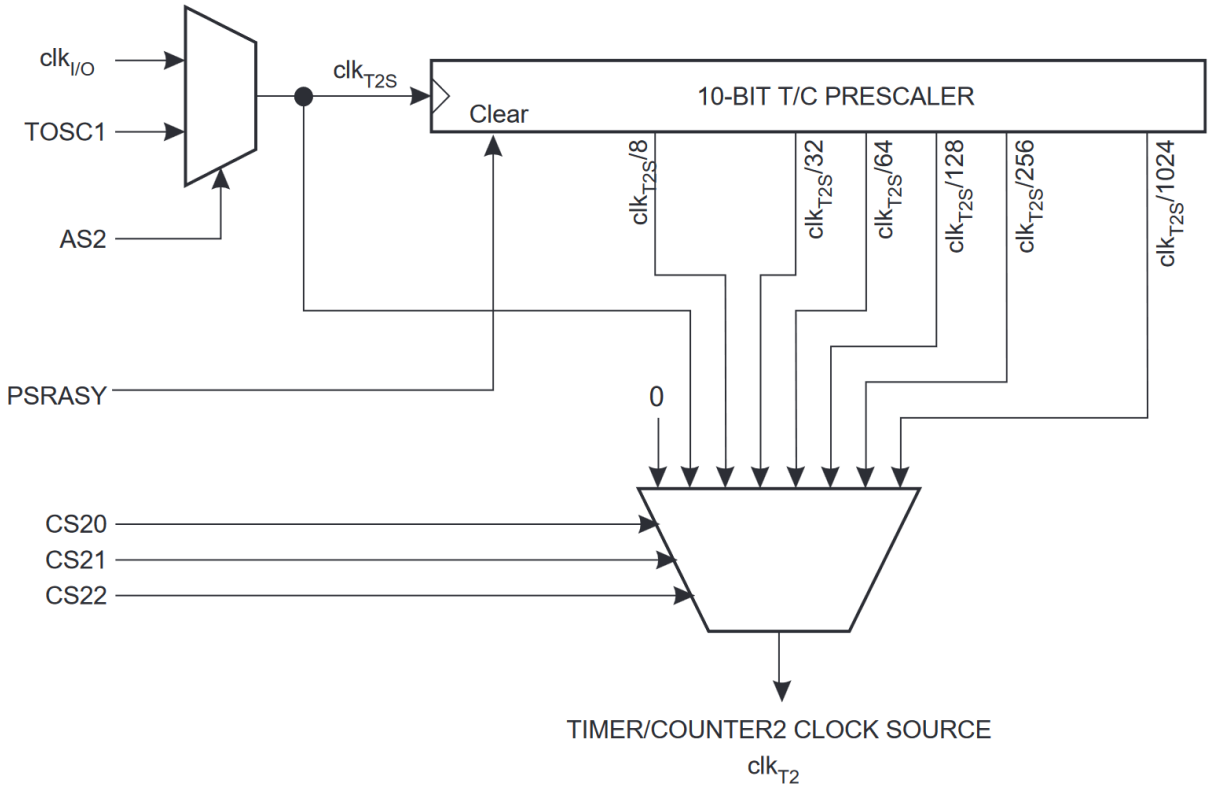
1. Dioda D1 świeci maksymalną jasnością.

2. Dioda D3 świeci połową jasności bez zauważalnego mrugania. Jasność oceń organoleptycznie[1].

Modyfikacja programu

Zmodyfikuj funkcję `pwmInitialize()`:

1. Mruganie diody jest zauważalne, ponieważ timer taktowany jest zbyt wolnym zegarem. Zapoznaj się z bitami `CS20`…`CS22`.
2. Użyj rejestru `OCR2A`, by ustawić jasność diody.



Rysunek 3: Za pomocą bitów `CS20`…`CS22` wybierany jest odpowiednio podzielony zegar

Potrzebne informacje znajdziesz w rozdziale *8-bit Timer/Counter2 with PWM and Asynchronous Operation*, w szczególności w sekcji *Register Description* dokumentacji mikrokontrolera.

[1] Innymi słowy „na oko”.

W tym ćwiczeniu mikrokontroler taktowany jest zegarem $clk_{I/O} = 2$ MHz.

Zadanie rozszerzone

Wymagania funkcjonalne

1. Dioda `D1` świeci maksymalną jasnością.
2. Jasność diody `D3` cyklicznie i płynnie zmienia się od minimalnej do maksymalnej i z powrotem.

Modyfikacja programu

Wykorzystaj przerwanie `TIMER2_OVF` (plik `pwm.cpp`), które wywoływane jest po przepełnieniu timera (na koniec każdego cyklu odliczania).

Przerwanie włącza flaga `TOIE2` oraz funkcja `sei()`.