Zachodniopomorski
Uniwersytet
Technologiczny

WYDZIAŁ ELEKTRYCZNY
Katedra Inżynierii Systemów, Sygnałów i Elektroniki

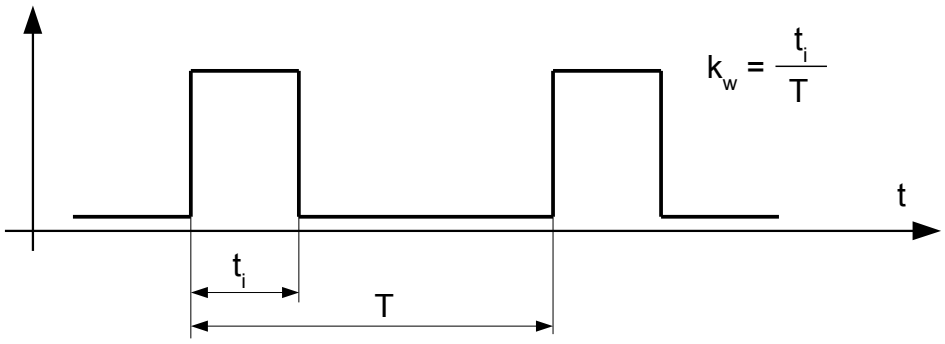
LABORATORIUM

TECHNIKA MIKROPROCESOROWA

Obsługa wyjść PWM w mikrokontrolerach Atmega16 - 32

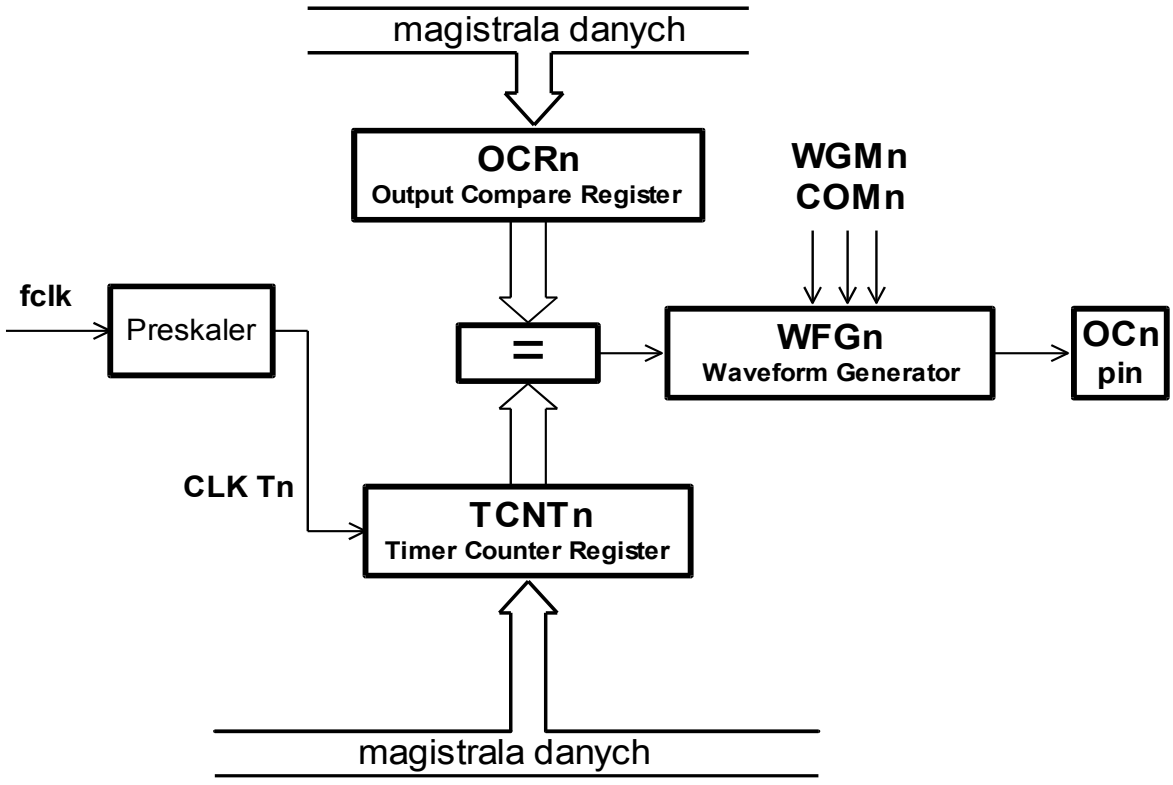
Opracował:
mgr inż. Andrzej Biedka

Sterowanie z wykorzystaniem modulacji PWM (ang. **P**ulse-**w**idth **m**odulation) jest obecnie szeroko stosowaną metodą w wielu dziedzinach techniki regulacji. Elementem regulacyjnym może być klucz pracujący dwustanowo (załącz/wyłącz), dzięki czemu straty w nim są nieznaczne bądź pomijalne. Warunkiem poprawnej regulacji jest zachowanie właściwej relacji okresu kluczowania do stałej czasowej obiektu regulowanego.



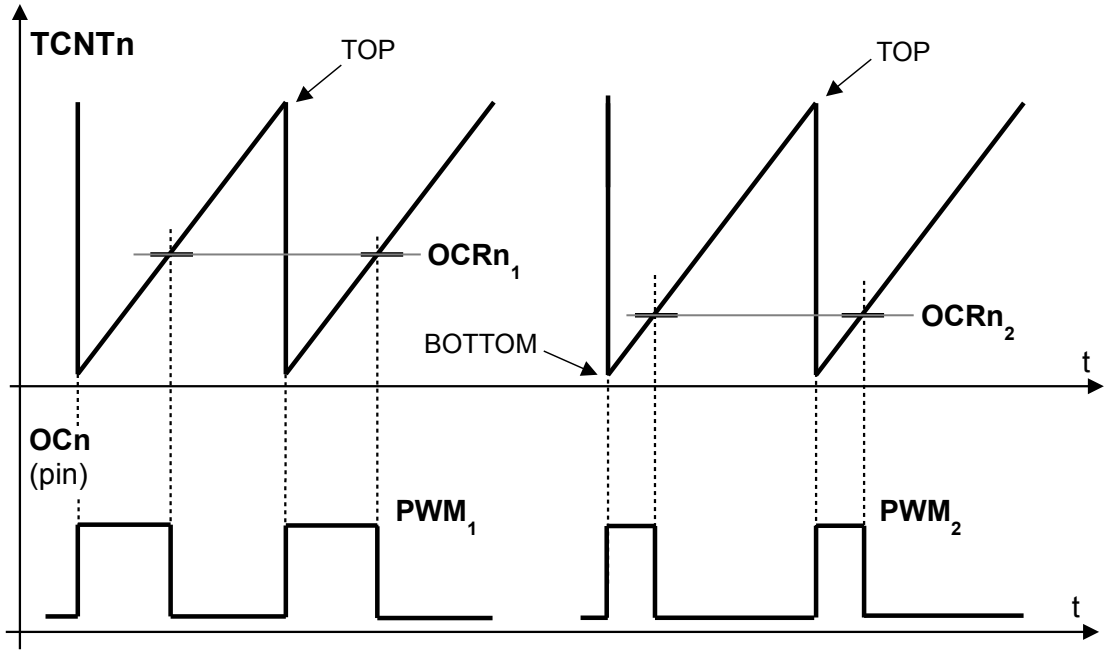
Rys. 9-1. Przebieg czasowy modulacji PWM

Współczesne mikrokontrolery wyposażane są w funkcje sprzętowych generatorów sygnału PWM. Układy te pracują w oparciu o timery mikrokontrolera. Podstawowa konfiguracja przedstawiona jest na rysunku 9-2 i dotyczy timerów 8-bitowych mikrokontrolera rodziny ATmega, czyli Timera0 i Timera2.
Impulsy zegarowe clk Tn są zliczane w głównym liczniku timera (**TCNT**n), który zlicza je w całym zakresie pojemności – dla timera 8-bitowego jest to 256 impulsów. Stan licznika **TCNT**n jest porównywany z zawartością pomocniczego rejestru **OCR**n. W chwili zrównania zawartości rejestrów następuje wygenerowanie sygnału zgodności przekazywanego do bloku kształtowania sygnału wyjściowego – dostępnego na wyprowadzeniu **OC**n.



Rys. 9-2. Uproszczony schemat blokowy generatora PWM w timerach 0 i 1 mikrokontrolera ATmega 16/32

Przebiegi czasowe obrazujące pracę generatora PWM przedstawione są na rysunku 9-3.



TOP – maksymalna zawartość rejestru TCNTn
BOTTOM – minimalna zawartość rejestru TCNTn

Rys. 9-3. Zależność współczynnika wypełnienia PWM od zawartości rejestru OCRn w trybie Fast PWM

Generatory PWM mogą pracować w różnych trybach, które wybierane są ustawieniami rejestrów konfiguracyjnych **TCCR**n. Najprostszym trybem pracy generatora jest tryb szybki PWM.

TCCRn **n** – numer timera (0 lub 2)

FOC	WGMn0	COMn1	COMn0	WGMn1	CSn2	CSn1	CSn0

FOC - wymuszanie stanu wyjścia OCn
WGMn1, WGMn0 - wybór trybu pracy generatora PWM. Patrz tabela 9-1
COMn1, COMn0 - wybór trybu pracy wyjścia OCn. Patrz tabela 9-2
CSn2 – CSn0 - ustawienie podziału preskalera. Patrz tabela 9-3

Tabela 9-1

WGMn1	WGMn0	Opis trybu pracy	Wartość TOP
0	**0**	Normalny (Normal)	0xFF
0	**1**	Symetryczny PWM (PWM, Phase Correct)	0xFF
1	**0**	CTC	OCRn
1	**1**	Szybki PWM (Fast PWM)	0xFF

Tabela 9-2 Tryby pracy wyjścia OCn dla trybu Fast PWM

COMn1	COMn0	Opis trybu pracy
0	0	Pin portu
0	1	Niewykorzystany
1	0	Zeruj stan OCn przy zgodności OCRn i TCNTn ustaw stan OCn przy przepełnieniu TCNTn (BOTTOM) PWM prosty
1	1	Ustaw stan OCn przy zgodności OCRn i TCNTn zeruj stan OCn przy przepełnieniu TCNTn (BOTTOM) PWM zanegowany

Wyboru odpowiedniej wartości okresu przebiegu PWM dokonuje się bitami CSn2 – CSn0 ustalającymi stopień podziału preskalera danego timera.

Tabela 9-3

CSn2	CSn1	CSn0	Współczynnik podziału preskalera
0	0	0	Brak taktowania, timer zatrzymany
0	0	1	**f**clk
0	1	0	**f**clk/8
0	1	1	**f**clk/32
1	0	0	**f**clk/64
1	0	1	**f**clk/128
1	1	0	**f**clk/256
1	1	1	**f**clk/1024

ZADANIA:

1. Dla sygnału taktującego mikrokontroler równego 16 MHz wyznaczyć zakres okresów sygnału PWM dla timerów 0 i 2.
2. .

Przebieg ćwiczenia:

– Utworzyć algorytm, napisać program softwarowego generatora PWM sterującego jasnością czterech diod LED. Pozostałe cztery diody mają być stale załączone. Należy wykorzystać przerwania timerów.

– Utworzyć algorytm i napisać program generatora PWM z wykorzystaniem wyjścia OCn mikrokontrolera. Program powinien cyklicznie zmieniać jasność jednej diody LED.

– Zmodyfikować powyższy program wprowadzając programowe sterowanie czterech sąsiednich diod.

– Zmodyfikować program z pkt. 2 wprowadzając sterowanie zespołu diod LED w funkcji obsługi przerwania od wejścia INT0 lub INT1. Sygnałem wyzwalającym przerwania zewnętrzne ma być wyjście OCn.

Literatura:

[1] Francuz T. Język C dla mikrokontrolerów AVR. Od podstaw do zaawansowanych aplikacji. Helion, Gliwice, 2011
[2] Baranowski R. Mikrokontrolery AVR ATmega w praktyce. BTC, Warszawa, 2005
[3] Kardaś M. Mikrokontrolery AVR. Język C. Podstawy programowania. Atnel, Szczecin, 2011.
[4] Witkowski A. Mikrokontrolery AVR. Programowanie w języku C. Przykłady zastosowań, PKJS, Katowice 2006
[5] Karta katalogowa mikrokontrolera Atmega32 firmy ATMEL.