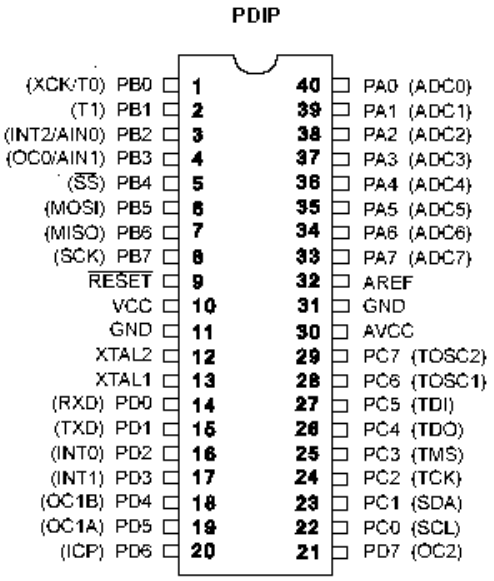
Cechy:

- zaawansowana architektura AVR
  - 131 instrukcji – większość jednocyklowych
  - 32 x 8-bit rejestry ogólnego przeznaczenia
  - możliwość pracy statycznej (0 Hz)
  - do 16 MIPS przy 16 MHZ
  - wbudowany 2-cyklowy układ mnożący
- nieulotne pamięci danych i programu
  - 32K bajtów programowanej w systemie pamięci programu Flash
    - trwałość: 10’000 cykli zapisu / kasowania
  - Obszar Boot Code z Lock Bits
    - Programowanie w systemie przez program w obszarze Boot
    - Operacje Read-While-Write
  - 1024 bajty EEPROM
    - trwałość ponad 100’000 cykli zapisu / kasowania
  - 2K bajtów wewnętrznej pamięci danych SRAM
  - Zabezpieczenie oprogramowania przed odczytem
- Interfejs JTAG
  - Boundary-Scan
  - Funkcja On-chip Debug
  - Programowanie Flash, EEPROM, fuse i lock-bitów przez JTAG
- urządzenia dodatkowe
  - dwa 8-bit liczniki z odrębnymi preskalerami i trybami porównania
  - jeden 16-bit licznik z oddzielnym preskalerem, trybem porównania i przechwytywania
  - licznik czasu rzeczywistego z oddzielnym oscylatorem
  - cztery kanały PWM
  - 8-kanałowy, 10-bit przetwornik analogowo-cyfrowy ADC
    - 8 pojedynczych kanałów
    - 7 kanałów różnicowych (tylko w obudowach TQFP)
    - 2 różnicowe kanały z programowalnym wzmocnieniem 1x, 10x lub 200x
  - interfejs TWI (I^2C)
  - programowany USART
  - Interfejs SPI
  - Programowalny watchdog z oddzielnym oscylatorem
  - Komparator analogowy
- specjalne cechy mikrokontrolera:
  - samoczynny reset po włączeniu zasilania i detektor zapięcia zasilającego
  - przestrajany oscylator RC
  - zewnętrzne i wewnętrzne źródła przerwań
  - 6 trybów obniżonego poboru mocy
- I/O
  - 32 programowalne linie wejścia / wyjścia
  - obudowy: 40-pin DIL; 44 TQFP; 44 MLF
- zakres napięć zasilania:
  - 2,7 – 5,5V dla ATmega32L
  - 4,5 – 5,5V dla ATmega32
- prędkość pracy:
  - 0 – 8MHz dla ATmega32L
  - 0 – 16MHZ dla ATmega32
- zużycie prądu przy 1 MHZ, 3V i 25C dla ATmega32L
  - aktywny – 1,1mA
  - Idle – 0,35mA
  - Power-down - <1uA

Opis:



ATmega32 jest 8-bit mikrokontrolerem o małym poborze mocy, zbudowanym na bazie architektury AVR RISC.
AVR posiada bogatą listę instrukcji z 32 rejestrami ogólnego przeznaczenia. Rejestry te są bezpośrednio połączone z jednostką arytmetyczno – logiczną (ALU) pozwalającą na dostęp do dwóch rejestrów jedną instrukcją w czasie jednego cyklu. Umożliwia to uzyskanie dużej efektywności i prędkości do dziesięciu razy większej od typowych układów CISC.
ATmega32 posiada: 32Kbajty pamięci programu ISP Flash, 1024 bajty EEPROM 2Kb pamięci statycznej RAM, 32 rejestry ogólnego przeznaczenia, 32 linie I/O ogólnego przeznaczenia, interfejs JTAG, On-chip Debugging, trzy liczniki z trybami porównywania, zewnętrzne i wewnętrzne przerwania, USART, TWI, 10-bitowy, 8-kanałowy przetwornik A/C z możliwością ustawienia wejść różnicowych z programowanym wzmocnieniem (tylko w obudowach TQFP), programowalny Watchdog, SPI, 6 wybieranych programowo trybów redukcji poboru mocy. Idle zatrzymuje procesor, ale pozwala pracować układom USART, TWI, AD, SRAM, licznikom, SPI i systemowi przerwań. Tryb Power-down zachowuje rejestry, ale zatrzymuje oscylator i wyłącza wszystkie pozostałe funkcje, aż do wystąpienia zewnętrznego przerwania lub reset. W trybie Power-save licznik w trybie asynchronicznym wybudza procesor. Tryb redukcji zakłóceń ADC zatrzymuje CPU i wszystkie układy we / wy za wyjątkiem licznika asynchronicznego i przetwornika analogowo-cyfrowego w celu zmniejszenia zakłóceń wytwarzanych przez mikrokontroler podczas konwersji wartości analogowej. W trybie Standby działa oscylator, podczas gdy reszta jest w trybie uśpienia. Pozwala to na bardzo szybki start.
Pamięć programu jest nieulotną pamięcią Flash programowaną w systemie (ISP) przez interfejs SPI, równolegle lub przez program znajdujący się w obszarze Boot. Program taki może użyć dowolnego interfejsu do programowania pamięci Flash.

Opis wyprowadzeń:

Vcc — napięcie zasilające część cyfrową

GND — masa

Port A (PA7...PA0) — Port A może służyć jako 8-bit dwukierunkowy port wejścia / wyjścia jeśli przetwornik ADC jest wyłączony. Port posiada wewnętrzne rezystory podciągające pull-up (wybierany dla każdego bitu). Gdy Port A jest ustawiony jako wejście i panuje na nim stan niski, przez końcówki będzie płynął prąd, o ile rezystory podciągające są włączone są włączone. Port ten ustawia wejścia trzystanowe w czasie aktywnego stanu reset, nawet jeśli oscylator nie pracuje.

Port B (PB7...PB0) — Port B jest 8-bit dwukierunkowym portem wejścia / wyjścia z wewnętrznymi rezystorami podciągającymi (wybieranymi dla każdego bitu). Gdy Port B jest ustawiony jako wejście i panuje na nim stan niski, przez końcówki będzie płynął prąd, o ile rezystory podciągające są włączone są włączone. Port ten

	ustawia wejścia trzystanowe w czasie aktywnego stanu reset, nawet jeśli oscylator nie pracuje.
Port C (PC7...PC0)	Port C jest 8-bit dwukierunkowym portem wejścia / wyjścia z wewnętrznymi rezystorami podciągającymi (wybieranymi dla każdego bitu). Gdy Port C jest ustawiony jako wejście i panuje na nim stan niski, przez końcówki będzie płynął prąd, o ile rezystory podciągające są włączone są włączone. Port ten ustawia wejścia trzystanowe w czasie aktywnego stanu reset, nawet jeśli oscylator nie pracuje. Jeżeli uruchomiony jest interfejs JTAG rezystory podciągające są włączone nawet gdy wystąpi sygnał reset. (końcówki PC5-TDI, PC3-TMS, PC2-TCK).
Port D (PC7...PC0)	Port D jest 8-bit dwukierunkowym portem wejścia / wyjścia z wewnętrznymi rezystorami podciągającymi (wybieranymi dla każdego bitu). Gdy Port D jest ustawiony jako wejście i panuje na nim stan niski, przez końcówki będzie płynął prąd, o ile rezystory podciągające są włączone są włączone. Port ten ustawia wejścia trzystanowe w czasie aktywnego stanu reset, nawet jeśli oscylator nie pracuje.
RESET	wejście zerujące. Stan niski na tym wejściu trwający dłużej niż wynosi minimalny okres, spowoduje zrestartowanie procesora, nawet gdy zegar nie pracuje.
XTAL1	wejście wzmacniacza odwracającego i wejście zewnętrznego sygnału zegarowego.
XTAL2	wyjście wzmacniacza
AVCC	napięcie zasilające dla portu A i przetwornika analogowo-cyfrowego. Powinno być z zewnątrz podłączone do VCC nawet jeśli przetwornik ADC nie jest używany. Jeśli ADC jest używany to AVCC powinno być dołączone do VCC przez filtr dolnoprzepustowy (np. L-C).
AREF	źródło napięcia odniesienia dla przetwornika ADC.

Rdzeń AVR:

AVR zbudowany jest w oparciu o strukturę Harwardzką z rozdzielonymi pamięciami i szynami danych i programu. Instrukcje w pamięci programu są wykonywane w trybie 1 – potokowym. W trakcie wykonywania jednej instrukcji, następna jest już pobierana z pamięci. To pozwala na wykonywanie instrukcji w każdym takcie zegara. Pamięć programu jest pamięcią Flash „programowaną w systemie" (ISP).

32 ośmiobitowe rejestry ogólnego przeznaczenia mogą być dostępne w jednym cyklu zegara dzięki czemu operacje ALU (jednostki arytmetyczno – logicznej) na tych rejestrach mogą być wykonywane również w jednym cyklu zegara. W większości operacji, w których argumenty znajdują się w dwóch rejestrach, a wynik operacji zapisywany jest w jednym z nich, ALU wykonuje w jednym takcie.

Sześć z 32 rejestrów może być użyte jako 16 bitowe rejestry wskaźnikowe do pośredniego adresowania pamięci, z możliwością wykonywania dodatkowych operacji obliczeniowych. Jeden z tych wskaźników może być również wykorzystany do odczytywania komórki pamięci programu Flash. Rejestry te określane są jako 16-bit rejestry X, Y i Z.

Alu wykonuje operacje arytmetyczne i logiczne pomiędzy rejestrami lub pomiędzy liczbą a rejestrem. Również mogą być wykonywane operacje na pojedynczym rejestrze. Istnieje również rejestr znaczników (flag, wskaźników) – Status Register, opisujący wynik ostatnio przeprowadzanej operacji arytmetycznej.

Tok programu może być modyfikowany instrukcjami skoków warunkowych, bezwarunkowych i powrotnych w całym obszarze pamięci. Większość instrukcji jest 16-bitowa, niektóre 32 bitowe.

Pamięć programu jest podzielona na dwie części: część ładującą „Boot" i część na programy. Obszar Boot posiada własne bity „Lock Bits" zabezpieczające przed wpisem lub odczytem. Instrukcja SPM wpisująca do obszaru pamięci programu musi być umieszczona w obszarze Boot.

Podczas wyzwalania przerwań i skoków powrotnych (wywołań) rejestr Licznika Programu (PC) jest odkładany na stos. Stos jest ulokowany w pamięci danych SRAM i jego rozmiar jest ograniczony tylko pojemnością tej pamięci. Użytkownik musi zdefiniować położenie stosu zaraz po starcie, zanim wywoływane będą przerwania lub podprogramy. Rejestr wskaźnika stosu znajduje się w przestrzeni I/O i możliwy jest jego zapis i odczyt. Dostęp do pamięci SRAM jest możliwy przez pięć trybów adresowania.

Przestrzeń adresowa pamięci jest liniowa.
Przerwania posiadają swoje rejestry sterujące znajdujące się w przestrzeni I/O i dodatkowy bit w rejestrze stanu kontrolujący wszystkie przerwania. Każde przerwanie posiada oddzielny wektor (adres) w obszarze wektorów przerwań. Posiadają one priorytet ważności: najniższe adresy wektorów przerwań mają najwyższy priorytet; najwyżej położone wektory – najniższy priorytet.
Rejestry I/O zajmują 64 adresy w pamięć. Są przeznaczone dla urządzeń dodatkowych znajdujących się w mikrokontrolerze. Są to m. in. Rejestry sterujące, SPI i pozostałe funkcje wejść/wyjść. Przestrzeń I/O może być adresowana bezpośrednio lub jako przestrzeń obszaru danych; $20-$5F.

ALU – jednostka arytmetyczno – logiczna:

Jednostka ALU jest połączona z rejestrami ogólnego przeznaczenia. Operacje arytmetyczno – logiczne między dwoma rejestrami lub między liczbą a rejestrem są wykonywane w czasie jednego cyklu zegara. Operacje wykonywane przez ALU dzielą się na trzy główne kategorie: arytmetyczne, logiczne i bitowe. Niektóre wersje umożliwiają wykonywanie pełnych operacji mnożenia liczb ze znakiem lub bez.

Rejestr stanu SREG:

Rejestr stanu zawiera informacje o wyniku większości wykonywanych instrukcji arytmetycznych. Mogą być użyte do zmiany toku wykonywanego programu.
Jest uaktualniany po instrukcjach ALU wyznaczonych w liście instrukcji.
Rejestr ten nie jest automatycznie odkładany na stosie po wejściu w obsługę przerwania i nie jest pobierany po wyjściu z przerwania. Trzeba to wykonać programowo.

Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – I: uaktywnienie ogólnego systemu przerwań**

ustawienie bitu I włącza ogólny system przerwań. Indywidualne przerwania ustawiane są w specjalnych, oddzielnych rejestrach. Jeśli bit I jest wyzerowany to żadne z przerwań ustawianych indywidualnie nie zostanie obsłużone. Jest zerowany automatycznie po pojawieniu się przerwania i ustawiany po wyjściu z przerwania instrukcją RETI. Może być ustawiany i kasowany programowo przez użycie instrukcji SEI i CLI.

- **bit 6 – T: bit kopiowania**

Jest używany do operacji bitowych jako bit źródłowy i docelowy za pomocą instrukcji BLD (wpisanie do bitu T) i BST (przepisanie z bitu T). Instrukcja BLD pozwala na skopiowanie jednego bitu rejestru z bloku rejestrów do bitu T, BST na skopiowanie bitu T do rejestru.

- bit 5 – H: przeniesienie połówkowe

Bit ten informuje o zaistnieniu przeniesienia połówkowego w niektórych instrukcjach arytmetycznych. Jest używany w arytmetyce dziesiętnej.

- **bit 4 – S: bit znaku, S=N ⊕ V**

bit znaku jest wynikiem sumy logicznej między flagą N a V.

- **bit 3 – V: bit przeniesienia pomocniczego**

używany jest w niektórych operacjach arytmetycznych

- **bit 2 – N: Flaga wartości ujemnych**

Wskazuje na ujemny wynik arytmetycznych lub logicznych operacji.

- **bit 1 – Z: bit zera**

ustawia się gdy wynikiem operacji arytmetycznych lub logicznych będzie zero.

- **bit 0 – C: bit przeniesienia**

jest ustawiany w wyniku zaistnienia przeniesienia podczas wykonywania operacji arytmetycznych lub logicznych.

Rejestry ogólnego przeznaczenia:

Są zoptymalizowane pod kątem listy instrukcji AVR RISC. Na zbiorze rejestrów ogólnego przeznaczenia możliwe są następujące operacje:

- wyprowadzenie 8-bit operandu i wprowadzenie 8-bit wyniku;
- wyprowadzenie dwóch 8-bit operandów i wprowadzenie 8-bit wyniku;
- wyprowadzenie dwóch 8-bit operandów i wprowadzenie 16-bit wyniku;
- wyprowadzenie 16-bit operandu i wprowadzenie 16-bit wyniku.

	7 0	Addr.	
	R0	$00	
	R1	$01	
	R2	$02	
	...		
	R13	$0D	
General	R14	$0E	
Purpose	R15	$0F	
Working	R16	$10	
Registers	R17	$11	
	...		
	R26	$1A	X-register Low Byte
	R27	$1B	X-register High Byte
	R28	$1C	Y-register Low Byte
	R29	$1D	Y-register High Byte
	R30	$1E	Z-register Low Byte
	R31	$1F	Z-register High Byte

Większość instrukcji na zbiorze tych rejestrów używa bezpośredniego dostępu i większość wykonuje się w jednym cyklu.
Każdy z rejestrów jest ulokowany na początku przestrzeni adresowej pamięci i do każdego z nich możliwy jest dostęp przez pośredni dostęp do pamięci.
Dostęp do rejestrów jest też przez wskaźniki X, Y i Z.

Rejestry X, Y, Z:
Rejestry R26...R31 posiadają dodatkowe funkcje jako 16-bitowe rejestry wskaźnikowe, które umożliwiają pośrednie adresowanie pamięci.

	15 XH 8	7 0	7 XL 0	7 0
Rejestr X		R27 ($1B)		R26 ($1A)
Rejestr Y		R29 ($1D)		R28 ($1C)
Rejestr Z		R31 ($1F)		R30 ($1E)

W różnych rodzajach adresowania rejestry wskaźnikowe posiadają dodatkowe funkcje automatycznego zwiększania i zmniejszania zawartości.

Wskaźnik stosu:
Jest używany głównie do odkładania tymczasowych danych, zmiennych i adresów powrotnych po wejściu w przerwania lub wywołania procedury. Wskazuje zawsze wierzchołek stosu. Stos rozwija się z góry na dół w obszarze pamięci danych.
Wskaźnik stosu musi być zdefiniowany w programie zanim zostaną wykonane instrukcje wywołania czy przerwania. Musi wskazywać na adres wyższy od $60. Jest zmniejszany o jeden gdy odkładane są na stos dane instrukcją PUSH, o dwa gdy odkładany jest adres powrotny z przerwań i instrukcji wywołań. Przy pobieraniu danych ze stosu instrukcją POP jest zwiększany o jeden, adresu o dwa poprzez instrukcje RET lub RETI.
Rejestr ten składa się z dwóch rejestrów 8-bit umieszczonych w obszarze I/O. Obszar pamięci w niektórych mikrokontrolerach jest na tyle mały, że wystarcza tylko rejestr SPL. W tym przypadku rejestr SPH nie jest umieszczony w procesorze.

Bit	15	14	13	12	11	10	9	8	
	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	7	6	5	4	3	2	1	0	
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Okres wykonywania instrukcji:
AVR CPU jest taktowana sygnałem zegarowym clk_{CPU} bezpośrednio z wybranego źródła zegara. Nie ma wewnętrznego podzielnika częstotliwości.
Architektura harwardzka, przetwarzanie potokowe i szybki dostęp ALU do rejestrów umożliwiają uzyskanie prędkości do 1MIPS na 1MHz.

Przerwania:

AVR posiada wiele różnych źródeł przerwań. Przerwania te razem z resetem posiadają oddzielne wektory umieszczone w pamięci programu. Każde z przerwań posiada własny bit uaktywniający, który musi być ustawiony w stan logicznej jedynki razem z bitem I w rejestrze SREG, żeby przerwanie mogło być aktywne. Zależnie od wartości licznika programu, przerwania mogą być automatycznie wyłączone, gdy bity Boot Lock BLB02 lub BLB12 są zaprogramowane.
Najniższy adres w pamięci programu jest zdefiniowany jako Reset i wektor przerwania. Różne przerwania mają swój priorytet ważności. Im niższy adres tym wyższy priorytet. Reset posiada najwyższy priorytet, następny jest INT0 – zewnętrzne przerwanie nr 0. Wektory przerwań mogą być przesunięte na początek obszaru Boot Flasch przez ustawienie bitu IVSEL w głównym rejestrze zarządzającym przerwaniami – GICR. Wektor reset również może być przesunięty do obszaru Boot Flash przez zaprogramowanie fusebitów BOOTRST.
Gdy wystąpi przerwanie bit I w rejestrze SREG zostaje wyzerowany i wszystkie przerwania zostają wyłączone. Programowo ten bit może być ustawiony w 1 w celu umożliwienia zachodzenia na siebie przerwań. Każde aktywne przerwanie może przerwać aktualnie wykonywane. Bit I jest automatycznie ustawiany, gdy przerwanie zostanie opuszczone instrukcją RETI.
Są dwa podstawowe rodzaje przerwań. Pierwszy typ jest wyzwalany w przypadku, gdy bit flagi (znacznika, wskaźnika) przerwania zostaje ustawiony. Licznik rozkazów przyjmuje wtedy wartość zapisaną w wektorze danego przerwania (zaczyna się wykonywać program obsługi przerwania), a bit flagi przerwań automatycznie kasowany. Flaga ta może być też skasowana przez wpisanie do niej jedynki. Jeśli okoliczność przerwania wystąpi podczas gdy bit uaktywniający (odblokowujący) odpowiednie przerwanie jest wyzerowany, bit flagi będzie ustawiony i zapamiętany aż do momentu gdy przerwanie zostanie odblokowane lub flaga zostanie wyzerowana programowo. Gdy jedno lub więcej przerwań wystąpi gdy ogólny bit przerwań I w SREG jest wyzerowany, odpowiednie bity flag przerwań będą ustawione i zapamiętane dopóki bit I nie zostanie ustawiony. Wtedy przerwania zostaną wykonane w kolejności zależnej od ich priorytetów ważności.
Drugi typ przerwań jest wyzwalany tak długo jak przerwanie występuje. Dla tych przerwań nie jest konieczny bit flagi (znacznika) przerwania. Jeśli sygnał przerwania zniknie zanim przerwanie będzie odblokowane, będzie ono zignorowane.
Po zakończeniu przerwania procesor zawsze wraca do głównego programu i wykonuje instrukcję następną po tej gdy wystąpiło przerwanie.
Rejestr statusowy (SREG) nie jest automatycznie zapamiętywany gdy CPU wchodzi w obsługę przerwania i nie jest odnawiany gdy przerwanie zostaje zakończone. Ta czynność musi być wykonana programowo.
Użycie instrukcji CLI natychmiast wyłącza przerwania. Żadne przerwanie nie będzie wykonywane po tej instrukcji, nawet jeśli wystąpi ono równocześnie z tą instrukcją.

Poniższy przykład pokazuje jak ta instrukcja może być użyta gdy wykonywany jest zapis do pamięci EEPROM:

Przykład w asemblerze:

```
in r16,SREG                 ; zapisz wartość SREG
cli                         ; wyłącz przerwania
sbi EECR , EEMWE            ; rozpocznij zapis do EEPROM
sbi EECR , EEWE
out SREG,r16                ; odzyskaj SREG (bit – I)
```

Przykład w C:

```
char cSREG;
cSREG = SREG;               /* zapisz zawartość SREG */
/* wyłącz przerwania podczas zapisu */
_CLI();
EECR | = (1<<EEMWE);        /* rozpocznij zapis do EEPROM */
EECR | = (1<<EEWE);
SREG = cSREG;               /* odzyskaj SREG (bit – I) */
```

Gdy urzyta jest instrukcja SEI, instrukcja następna będzie wykonana zanim rozpocznie się obsługa przerwania:

Przykład w asemblerze:

```
sei                         ; włącza system przerwań
sleep                       ; wprowadza tryb uśpienia aż do wystąpienia przerwania
; zanim wystąpi jakiekolwiek przerwanie
```

Przykład w C:

```
_SEI();                     /* włącza system przerwań */
_SLEEP();                   /* wprowadza tryb uśpienia aż do wystąpienia przerwania */
/* zanim wystąpi jakiekolwiek przerwanie */
```

Czas odpowiedzi na przerwanie:
Odpowiedź na przerwanie dla wszystkich przerwań aktywnych wynosi najmniej cztery cykle. Po tym czasie wykonywany jest program obsługi przerwania. Po okresie 4 cykli licznik programu jest odkładany na stos. Wektor jest tutaj skokiem do programu obsługi i pochłania trzy cykle. Jeśli przerwanie wystąpi podczas wykonywania instrukcji wielocyklowych, instrukcje te zostaną najpierw w całości wykonane, dopiero po tym nastąpi skok do przerwania. Jeśli wystąpi przerwanie gdy MCU znajduje się w trybie Sleep, czas reakcji jest zwiększony o cztery cykle. Jest to związane z czasem wybudzania się układu w z trybu Sleep.
Powrót z programu przerwania trwa cztery cykle. W tym czasie licznik programu (dwubajtowy) jest ściągany ze stosu, wskaźnik stosu jest zwiększany o dwa, a bit I w SREG jest ustawiany.

Pamięci ATmega32

W tym rozdziale opisane są różne rodzaje pamięci. Architektura AVR posiada dwa rodzaje pamięci: pamięć danych i programu. Dodatkowo posiada pamięć nieulotną do przechowywania danych. Wszystkie pamięci są liniowe.

Pamięć programu Flash programowana w systemie:
ATmega32 posiada 32Kbajty wewnętrznej, programowalnej pamięci Flash do przechowywania programu. Ponieważ instrukcje AVR są 16- lub 32-bit, pamięć ta jest zorganizowana jako 16K x 16. Dla bezpieczeństwa programu, obszar pamięci Flash jest podzielony na dwie części: Boot Program i obszar na programy.
Trwałość Flash wynosi powyżej 10’000 cykli zapisu / kasowania. Licznik programu (PC) jest 14-bitowy – może adresować 16k pamięci.
Można w niej umieszczać tablice ze stałymi, odczytywane instrukcją LPM.

Pamięć danych SRAM:
2144 pamięci danych zajmują: blok rejestrów, I/O i wewnętrzna pamięć SRAM. Pierwsze 96 bajtów to blok rejestrów ogólnego przeznaczenia i obszar I/O, następne 2048 bajtów to wewnętrzna pamięć danych SRAM.
Istnieje pięć trybów adresowania pamięci: bezpośrednie, pośrednie z przesunięciem, pośrednie, pośrednie z pre-dekrementacją i pośrednie z post-inkrementacją. W bloku rejestrów ogólnego przeznaczenia, R26 – R31 służą do adresowania pośredniego.
Bezpośrednie adresowanie obejmuje cały obszar pamięci.
Adresowanie pośrednie z przesunięciem obejmuje 63 adresy od wartości wskaźnika Y lub Z.
Adresowanie pośrednie z pre-dekrementacją i post-inkrementacją powodują zmniejszenie lub zwiększenie rejestrów wskaźnikowych X, Y lub Z.
32 rejestry ogólnego przeznaczenia, 64 rejestry specjalne I/O i 2048 bajtów pamięci danych SRAM mogą być adresowane wszystkimi sposobami.

Czas dostępu do pamięci:
Czas dostępu do pamięci SRAM wynosi dwa cykle zegarowe clk_{CPU}.

Pamięć danych EEPROM:
ATmega32 zawiera 1024 bajty pamięci danych EEPROM. Jest zlokalizowana w oddzielnym obszarze danych. Odczytać lub zapisać bajt można tylko pojedynczo. Posiada trwałość ponad 100’000 cykli zapisu / kasowania. Dostęp do tej pamięci odbywa się za pośrednictwem rejestrów specjalnych: rejestru adresowego danych i sterującego.
Zapis / odczyt:
Rejestry dostępu do EEPROM znajdują się w obszarze I/O.
Czas zapisu do EEPROM jest pokazany w tabeli 1.
Zapis następnego bajtu powinien nastąpić po stwierdzeniu dostępu po zapisie poprzedniego bajtu. Podczas zapisu do pamięci muszą być spełnione pewne warunki dotyczące napięcia zasilającego i prędkości pracy układu.
Zapis należy przeprowadzać wg pewnych zasad.
Podczas odczytu z pamięci EEPROM, CPU jest zatrzymywane na okres dwóch cykli zegarowych zanim następna instrukcja będzie wykonana.

Rejestry adresowe EEARH, EEARL:

Bit	15	14	13	12	11	10	9	8	
	-	-	-	-	-	-	EEAR9	EEAR8	EEARH
	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEARL
	7	6	5	4	3	2	1	0	
Odczyt/zapis	O	O	O	O	O	O	Z/O	Z/O	
	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	x	x	
	x	x	x	x	x	x	x	x	

- **bity 15...10 – Res: bity zarezerwowane.**

Bity te są zarezerwowane w ATmega32 i odczytywane są zawsze jako zero.

- **bity 9...0 – EEAR9...0: Adres EEPROM**

Rejestry te (EEARL, EEARH) zawierają numer komórki pamięci EEPROM spośród 1024. obszar adresowania: 0 – 1023. Zawartość rejestrów jest nie określona po włączeniu zasilania. Właściwa wartość musi być wpisana przed dostępem do pamięci.

Rejestr danych EEDR:

Bit	7	6	5	4	3	2	1	0	
	MSB							LSB	EEDR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bity 7...0 – EEDR7.0: dane EEPROM**

Rejestr ten zawiera daną która zostanie zapisana do EEPROM pod adres zapisany w rejestrach adresowych EEAR. Przy operacjach odczytu rejestr EEDR zawiera daną odczytaną z pamięci z adresu zawartego w EEAR.

Rejestr sterujący EEPROM - EECR:

Bit	7	6	5	4	3	2	1	0	
	-	-	-	-	EERIE	EEMWE	EEWE	EERE	EECR
Odczyt/zapis	O	O	O	O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	x	0	

- **bity 7...4 – zarezerwowane**

wartością odczytywaną jest zawsze zero.

- **bit 3 – EERIE: odblokowanie gotowości przerwań:**

Ustawienie bitu EERIE odblokowuje przerwanie gotowości EEPROM, gdy bit I w SREG jest ustawiony.
Wpisanie zera do EERIE zablokowuje. Przerwanie to jest generowane gdy EEWE jest skasowane.

- **bit 2 – EEMWE:**

bit EEMWE decyduje czy ustawienie EEWE powoduje zapis do EEPROM. Gdy EEMWE jest ustawiony, wpis jedynki do EEWE w ciągu czterech cykli zegara spowoduje zapis do EEPROM pod wskazany adres. Gdy EEMWE jest wyzerowany, ustawienie EEWE nie będzie skuteczne.
Gdy EEMWE będzie ustawione programowo, sprzętowe wyzerowanie nastąpi po czterech cyklach.

- **bit 1 – EEWE: uaktywnienie zapisu do EEPROM:**

Gdy wartości adresu i danej są prawidłowo ustawione, bit ten musi być w stanie wysokim aby nastąpił zapis do pamięci. EEMWE musi być ustawione zanim wpisze się jedynkę do EEWE. Procedura zapisu do pamięci (punkty 3 i 4 nie są istotne):

1. czekaj aż w EEWE pojawi się zero
2. czekaj aż SPMEN w SPMCR osiągnie zero.
3. wpisz nowy adres EEPROM do EEAR
4. wpisz nową daną do rejestru EEDR.
5. ustaw bit EEMWE
6. w ciągu czterech cykli od ustawienia EEMWE, wpisz jeden do EEWE

EEPROM nie może być programowany gdy CPU zapisuje do pamięci Flash. Program musi czekać aż programowanie Flash zostanie zakończone. Punkt 2 jest wymagany gdy oprogramowanie zawiera Boot Loader pozwalający CPU programować Flash. Jeżeli CPU nigdy nie będzie wpisywać do Flash, krok 2 może być pominięty.
Zaleca się wyłączenie globalnego systemu przerwań pomiędzy punktami 5 i 6 podczas zapisu do EEPROM ze względu na możliwość przerwania procesu zapisu. Np. gdy procedura przerwania odwołując się do EEPROM zmodyfikuje rejestry EEAR i EEDR.
Jeżeli czas dostępu zapisu jest przekroczony, bit EEWE zostaje samoczynnie wyzerowany.
Gdy EEWE będzie ustawiony, CPU jest wstrzymywany na dwa cykle.

- **bit 0 – EERE: uaktywnienie odczytywania z EEPROM:**

Gdy wartości adresu i danej są prawidłowo ustawione, bit ten musi być ustawiony żeby odczytać pamięć. Do odczytu potrzebna jest jedna instrukcja a wynik otrzymuje się od razu. Gdy EEPROM jest odczytywany, CPU jest zatrzymywany na cztery cykle zegara.
Podczas operacji zapisu nie jest możliwy odczyt z EEPROM ani zmiana EEAR.
Tabela 1. czas programowania EEPROM:

Symbol	Liczba cykli kalibrowanego oscylatora RC (1)	Typowy czas programowania
Zapis EEPROM (przez CPU)	8448	8,5 ms

(1) Użyj 1 MHz niezależnie od ustawień CKSEL w fusebitach.

Poniższy przykład pokazuje sposób zapisu do EEPROM. W przykładzie tym zakłada się, że przerwania są kontrolowane (np przez wyłączenie systemu przerwań) więc przerwania nie powinny wystąpić w trakcie wykonywania tych programów. Dotyczy to również funkcji zapisu do Boot Loader w programie. Jeśli taka występuje zapis do EEPROM może się odbyć dopiero po zakończeniu instrukcji SPM

.

Przykład w asemblerze:

```
EEPROM_write:
        ; czekaj aż zakończony zostanie poprzedni zapis do EEPROM
        sbic EECR , EEWE
        rjmp EEPROM_write
        ; tu adres wpisywany jest do rejestrów r18,r17
        ; przepisz adres z r18:r17 do rejestru adresowego
        out EEARH , r18
        out EEARL , r17
        ; zapisz daną (r16) do rejestru danych
        out EEDR , r16
        ; ustaw EEMWE
        sbi EECR , EEMWE
        ; rozpocznij zapis w EEPROM przez ustawienie EEWE
        sbi EECR , EEWE
        ret
```

Przykład w C:

```
        void EEPROM_write(unsigned int uiAddress, unsigned char ucData)
        {
            /* czekaj aż zakończony zostanie poprzedni zapis do EEPROM  */
        while(EECR & (1<<EEWE))
        ;
              /* tu adres wpisywany jest do rejestrów r18,r17
             przepisz adres z r18:r17 do rejestru adresowego */
        EEAR = uiAddress;
        EEDR = ucData;
            /* ustaw EEMWE  */
        EECR | = (1<<EEMWE);
            /* rozpocznij zapis w EEPROM przez ustawienie EEWE */
        EECR | = (1<<EEWE);
        }
```

Poniższy przykład pokazuje sposób odczytu z pamięci EEPROM. Stan przerwań jak wyżej.

Przykład w asemblerze:

```
    EEPROM_read:
        ; czekaj aż zakończony zostanie poprzedni zapis do EEPROM
    sbic EECR,EEWE
    rjmp EEPROM_read
            ; tu adres wpisywany jest do rejestrów r18,r17
            ; przepisz adres z r18:r17 do rejestru adresowego
    out EEARH, r18
    out EEARL, r17
        ; rozpocznij odczyt z EEPROM przez ustawienie EERE
    sbi EECR,EERE
        ; odczytaj daną z rejestru danych
    in r16,EEDR
    ret
```

Przykład w C:

```
unsigned char EEPROM_read(unsigned int uiAddress)
{
/* czekaj aż zakończony zostanie poprzedni zapis do EEPROM  */
while(EECR & (1<<EEWE))
;
        /* tu adres wpisywany jest do rejestrów r18,r17
         przepisz adres z r18:r17 do rejestru adresowego r */
EEAR = uiAddress;
/* rozpocznij odczyt z EEPROM przez ustawienie EERE */
EECR |= (1<<EERE);
/* powróć z funkcj z odczytaną daną */
return EEDR;
}
```

Zapis do EEPROM w trybie Power-down Sleep:
W sytuacji gdy wprowadzenie trybu Power-down Sleep nastąpi podczas zapisu do pamięci EEPROM, zapis będzie kontynuowany i dokonany bez przekroczenia czasu. Jednak gdy operacja zapisu będzie wykonana, oscylator będzie ciągle pracował i układ nie przejdzie w tryb obniżonego poboru mocy. Dlatego powinno się sprawdzić przed wprowadzeniem w tryb Power-down czy zapis do pamięci został zakończony.

Zapobieganie uszkodzeniom EEPROM:
Gdy napięcie zasilające Vcc jest zbyt niskie, dane w EEPROM mogą być uszkodzone

Uszkodzenie danych w EEPROM może być spowodowane przez dwie sytuacje:
- zapis do pamięci EEPROM wymaga pewnego minimalnego napięcia,
- CPU sam może wykonać instrukcję niepoprawnie gdy napięcie jest zbyt niskie.

Środki zapobiegawcze:
Utrzymywać stan RESET (poziom niski) podczas nieprawidłowego napięcia. Można to uzyskać uaktywniając wewnętrzny detektor niskiego napięcia – Brown-out Detector (BOD). Gdy napięcie zasilające będzie zbyt niskie zadziała układ Reset. Jeżeli reset wystąpi podczas operacji zapisu, operacja ta zostanie wykonana do końca pod warunkiem, że napięcie zasilające będzie właściwe.

Obszar I/O:
Wszystkie urządzenia w które wyposażona jest ATmega32 są położone w obszarze I/O. Dostęp do niego jest możliwy dzięki instrukcjom IN i OUT. Instrukcje te przesyłają dane między 32 rejestrami ogólnego przeznaczenia o obszarem I/O. W obszarze od $00 do $1F rejestry I/O mogą być adresowane bezpośrednio bitowo (możliwość ustawienia pojedynczych bitów) dzięki instrukcjom SBI i CBI. Wartość każdego bitu w tych rejestrach może być sprawdzony instrukcjami SBIS i SBIC. Jeżeli obszar I/O jest adresowany instrukcjami IN i OUT adresy przez nie obejmowane wynoszą od $00 do $3F. Jeśli instrukcjami adresującymi są LD i ST, do tych samych adresów należy dodać $20 (czyli $20 - $5F). Dla instrukcji LD i ST poniżej $20 leży blok rejestrów ogólnego przeznaczenia (r0 – r31), powyżej $5F pamięć SRAM.
Ze względy na zgodność z przyszłymi modelami bity zarezerwowane powinny być ustawione na zero jeśli program się do nich odwołuje, zarezerwowane adresy komórek pamięci nigdy nie powinny być zapisywane.
Niektóre bity flag mogą przyjmować zero logiczne po wpisaniu do nich jedynki. Instrukcje CBI i SBI działają na wszystkich bitach w rejestrach I/O, wpisując o jeden wstecz do flagi odczytywanej jako ustawiona, a więc kasuje te flagi (?). CBI i SBI działają tylko w obszarze $00 - $1F.

Zegar systemowy

Zegar systemowy:
Wszystkie źródła zegara nie muszą być aktywne w danym czasie. W celu redukcji poboru mocy, źródła sygnału zegarowego urządzeń nie używanych mogą być zatrzymane przez różne tryby uśpienia Sleep.

Zegar CPU – clk_{CPU}:
Jest używany do napędzania różnych części rdzenia AVR.

I/O – $clk_{I/O}$:
Jest używany przez większość urządzeń I/O jak liczniki, SPI, USART, także przez układy zewnętrznych przerwań. Niektóre z nich mogą wykrywać sygnały asynchroniczne, nawet jeśli zegar I/O jest wyłączony.

clk_{FLASH}:
używany w operacjach interfejsu Flash. Zazwyczaj jest taki sam jak clk_{CPU}.
Zegar asynchroniczny clk_{ASY}:
Taktuje licznik asynchroniczny (TC2) bezpośrednio z zegarkowego rezonatora kwarcowego 32,768kHz. Pozwala na pracę licznika jako zegara czasu rzeczywistego (również gdy mikrokontroler jest w stanie uśpienia).
Zegar przetwornika A/C – clk_{ADC}:
Pozwala zatrzymać CPU w trakcie konwersji, dzięki czemu unika się zakłóceń od cyfrowych obwodów. Pozwala to na podniesienia dokładności przetwarzania.

Źródła sygnałów zegarowych:
Źródło sygnału zegarowego wybierane jest przez Flash Fuse:

Rodzaj zegara	CKSEL3...0
Zewnętrzny rezonator kwarcowy / ceramiczny	1111 – 1010
Zewnętrzny rezonator o małej częstotliwości	1001
Zewnętrzny oscylator RC	1000 – 0101
Wewnętrzny, regulowany generator RC	0100 – 0001
Zewnętrzny sygnał zegarowy	0000

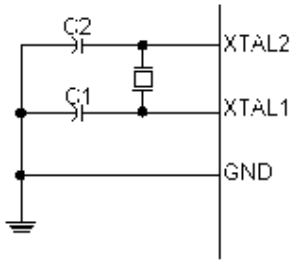
„1" oznacza bit niezaprogramowany, „0" zaprogramowany.

Gdy CPU zostaje wybudzony z trybu Power-Down lub Power-Save, wybrany sygnał zegarowy jest użyty w trakcie startu, zapewniając ustabilizowanie się pracy oscylatora zanim CPU przystąpi do wykonywania instrukcji. Jeżeli CPU rozpoczyna pracę po Reset, pojawia się dodatkowe opóźnienie. Watchdog służy do określenia czasu pracy zanim jednostka zostanie zresetowana.

Typ. Time-out (Vcc=5,0V)	Typ. Time-out (Vcc=3,0V)	Liczba cykli
4,1 ms	4,3 ms	4 K (4096)
65 ms	69 ms	64 K (65536)

Standardowe ustawienia CKSEL = "0001" , SUT = "10", zegar ustawiony jest wewnętrzny generator RC 1MHz z długim czasem uruchamiania. Zmiany mogą być wprowadzone przez programator ISP lub równoległy.

Oscylator kwarcowy:
XTAL1 i XTAL2 są wejściem i wyjściem odwracającego wzmacniacza. Użyty może być dowolny rezonator kwarcowy. Fuse-bity CKOPT służą do wybierania między dwoma trybami pracy wzmacniacza Oscylatora. Gdy CKOPT jest zaprogramowane, wyjście będzie oscylować w pełnym zakresie poziomu sygnału. Jest to używane gdy rezonator kwarcowy pracuje w otoczeniu z dużą ilością zakłóceń, lub gdy wyjście XTAL2 zasila inny układ. Ten tryb daje szeroki zakres częstotliwości pracy. Gdy CKOPT jest niezaprogramowane, poziom na wyjściu oscylatora jest niewielki. Zmniejsza się dzięki temu pobór prądu, ale zawęża się zakres częstotliwości i sygnał z wyjścia oscylatora (XTAL2) nie może być używany do niczego innego.



Dla rezonatorów, maksymalna częstotliwość wynosi 8 MHz gdy CKOPT nie jest zaprogramowane i 16 MHz gdy jest zaprogramowane. C1 i C2 zawsze powinny być równe. Wartości tych kondensatorów zależą od typu rezonatora, tolerancji i zakłóceń elektromagnetycznych. W przypadku użycia rezonatora ceramicznego należy dać kondensatory zalecane przez producenta.

Oscylator może pracować w trzech różnych trybach, każdy zoptymalizowany dla danego zakresu częstotliwości. Tryby te można wybrać przez fuse-bity CKSEL3...1:

CKOPT	CKSEL3...1	Zakres częstotliwości [MHz]	Wartości kondensatorów C1, C2 [pF]
1	101(1)	0,4 – 0,9	-
1	110	0,9 – 3,0	12 – 22
1	111	3,0 – 8,0	12 – 22
0	101, 110, 111	$\geq$ 1,0	12 – 22

(1) – ta opcja powinna być wybrana tylko dla rezonatorów ceramicznych.

Bity Fuse CKSEL0 razem z SUT1...0 służą do wyboru czasu uruchamiania:

CKSEL0	SUT1...0	Uruchomienie z trybów Power-down i P-save	Dodatkowe opóźnienie po reset (Vcc=5V)	Zalecane użycie
0	00	258 CK(1)	4,1 ms	Rezonator ceramiczny, szybkie uruchomienie
0	01	258 CK(1)	65 ms	Rezonator ceramiczny, powolne uruchomienie
0	10	1K CK(2)	-	Rezonator ceramiczny, włączone BOD
0	11	1K CK(2)	4,1 ms	Rezonator ceramiczny, szybkie uruchomienie
1	00	1K CK(2)	65 ms	Rezonator ceramiczny, powolne uruchomienie
1	01	16K CK	-	Rezonator kwarcowy, włączone BOD
1	10	16K CK	4,1 ms	Rezonator kwarcowy, szybkie uruchomienie
1	11	16K CK	65 ms	Rezonator kwarcowy, powolne uruchomienie

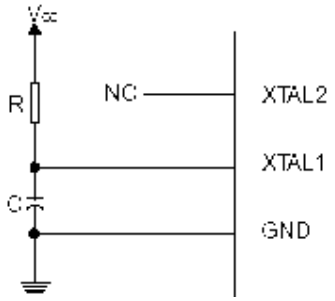
(1) – Ten tryb nie jest właściwy dla rezonatorów kwarcowych.
(2) – może być użyte z rezonatorami kwarcowymi.

Rezonator kwarcowy o małej częstotliwości:
W celu użycia do taktowania układu rezonatora kwarcowego 32,768kHz należy ustawić Fuse-bity CKSEL na „1001". Można też wybrać dołączenie do końcówek XTAL1 i XTAL2 wewnętrznych kondensatorów 36pF. Wybranie tego rezonatora powoduje, że czas rozpoczęcia pracy Start-up zależy od ustawienia fuse-bitów SUT:

SUT1...0	Uruchomienie z trybów Power-down i P-save	Dodatkowe opóźnienie po reset (Vcc=5V)	Zalecane użycie
00	1K CK(1)	4,1 ms	Szybkie uruchomienie, lub włączone BOD
01	1K CK(1)	65 ms	Powolne uruchomienie
10	32K CK	65 ms	Stała częstotliwość przy starcie
11	zarezerwowane		

(1) – ta opcja może być wybrana tylko przy stabilnej częstotliwości podczas uruchamiania układu.

Zewnętrzny generator RC:
Sposób podłączenia:



Częstotliwość określona jest wzorem: f=1/(3RC). Kondensator powinien mieć co najmniej 22pF. Przez odpowiednie ustawienie fuse-bitów CKOPT można włączyć wewnętrzny kondensator 36pF między XTAL1 a GND, co wyeliminuje konieczność użycia zewnętrznego kondensatora.
Generator ten może pracować w czterech różnych trybach, każdy dobrany do pewnego zakresu częstotliwości. Wybór następuje przez CKSEL3...0:

CKSEL3...0	Przedział częstotliwości [MHz]
0101	≤ 0,9
0110	0,9 – 3,0
0111	3,0 – 8,0
1000	8,0 – 12,0

Po wybraniu tego generatora czas uruchomienia układu zależy od ustawień SUT:

SUT1...0	Uruchomienie z trybów Power-down i P-save	Dodatkowe opóźnienie po reset (Vcc=5V)	Zalecane użycie
00	18 CK	-	Włączone BOD
01	18 CK	4,1 ms	Szybkie uruchomienie
10	18 CK	65 ms	Powolne uruchomienie
11	6 $CK_{(1)}$	4,1 ms	Szybkie uruchomienie, włączone BOD

(1) – może być użyty przy częstotliwości dochodzącej do maksimum.

Regulowany wewnętrzny generator RC:
Generator ten może pracować z częstotliwościami 1 , 2 , 4 , 8 MHz. Częstotliwości te są dla 25°C i 5V. Wybór następuje przez ustawienie fuse-bitów CKSEL. Układ pracuje wtedy bez zewnętrznych elementów. CKOPT powinny być niezaprogramowane, gdy używany jest wewnętrzny generator.

CKSEL3...0	Częstotliwość znamionowa (MHz)
0001(1)	1,0
0010	2,0
0011	4,0
0100	8,0

(1) – układy dostarczane są z tym ustawieniem.

Podczas pracy z wewnętrznym generatorem czas uruchamiania zależy od ustawień SUT. Końcówki XTAL1 i XTAL2 powinny pozostać niepodłączone.

SUT1...0	Uruchomienie z trybów Power-down i P-save	Dodatkowe opóźnienie po reset (Vcc=5V)	Zalecane użycie
00	6 CK	-	Włączone BOD
01	6 CK	4,1 ms	Szybkie uruchomienie
10(1)	6 CK	65 ms	Powolne uruchomienie
11	zarezerwowane		

(1) – układy dostarczane są z tym ustawieniem.

Rejestr regulacji generatora – OSCCAL:

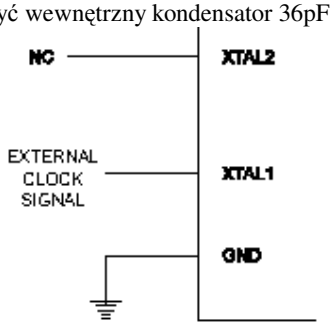
Bit	7	6	5	4	3	2	1	0	
	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	OSCCAL
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	Wartość kalibracji								

- **bity 7...0 – CAL7...0: wartość kalibracji generatora:**

Wpis do tego rejestru powoduje zmianę częstotliwości generatora wewnętrznego. Podczas resetu, 1MHz wartość kalibracji jest automatycznie wpisywana od rejestru OSCCAL. Jeśli generator pracuje przy innych częstotliwościach, wartość ta musi być wpisana ręcznie. Można to uzyskać np. przez odczytanie wartości kalibracji przez programator i zapisanie go do Flash lub EEPROM. Następnie wartość ta może być odczytana przez program i wpisana do rejestru OSCCAL. Gdy w OSCCAL jest $00 generator pracuje z najniższą częstotliwością. Wpisanie $FF spowoduje pracę przy najwyższej częstotliwości. Jeżeli następuje zapis do Flash lub EEPROM nie można podnosić częstotliwości ponad 10%, ponieważ może to spowodować błędy w zapisie.
Zakres częstotliwości wewnętrznego generatora RC:

Wartość OSCCAL	Wartość minimalna częstotliwości (%)	Wartość maksymalna częstotliwości (%)
$00	50	100
$7F	75	150
$FF	100	200

Zewnętrzny zegar:
Podczas pracy z zewnętrznym sygnałem zegarowym, fuse-bity CKSEL muszą być ustawione na „0000”. Przez zaprogramowanie CKOPT można włączyć wewnętrzny kondensator 36pF między XTAL1 a GND.



Podczas pracy z zewnętrznym sygnałem zegarowym czas uruchamiania zależy od ustawień SUT.

SUT1...0	Uruchomienie z trybów Power-down i P-save	Dodatkowe opóźnienie po reset (Vcc=5V)	Zalecane użycie
00	6 CK	-	Włączone BOD
01	6 CK	4,1 ms	Szybkie uruchomienie
10	6 CK	65 ms	Powolne uruchomienie
11	zarezerwowane		

Zewnętrzny sygnał zegarowy nie powinien zmieniać nagle częstotliwości. Zwiększanie częstotliwości więcej niż 2% na jeden cykl zegara może spowodować nieprzewidywalne zachowanie się CPU. Podczas zmiany częstotliwości powinno się trzymać układ w stanie reset.

Oscylator licznika/timera:
Rezonator ceramiczny włączany jest między wyprowadzenia TOSC1 i TOSC2. Nie są wymagane zewnętrzne kondensatory. Oscylator jest przewidziany do pracy z rezonatorem zegarkowym 32,768kHz. Nie jest zalecane dołączanie zewnętrznego sygnału zegarowego do końcówki TOSC1.

Zarządzanie trybami redukcji poboru mocy:

Tryby sleep umożliwiają wyłączenie nieużywanych modułów w MCU. AVR posiada różne tryby sleep pozwalające użytkownikowi zredukowanie poboru mocy stosownie do wymagań aplikacji.

Wprowadzenie jednostki w jeden z sześciu trybów sleep, odbędzie się przez ustawienie bitu SE w rejestrze MCUCR i wykonanie przez procesor instrukcji SLEEP. Bity SM2, SM1, SM0 służą do określenia który z trybów sleep (Idle, redukcja szumów ADC, Power-down, Power-save, Standby i Extended standby) będzie włączony po wykonaniu instrukcji SLEEP. Jeżeli któreś przerwanie wystąpi podczas gdy MCU jest w trybie sleep, układ zostanie wybudzony. MCU zostanie wstrzymany na cztery cykle (czas uruchomienia), wykona procedurę przerwania i powróci do wykonywania programu zaczynając od instrukcji następnej po SLEEP. Zawartość rejestrów ogólnego przeznaczenia i pamięci RAM nie ulegnie zmianie. Jeżeli w trakcie trybu sleep wystąpi Reset, MCU wybudzi się i wykona program od początku.

Rejestr sterujący MCU – MCUCR

Rejestr ten zawiera bity sterujące zarządzaniem trybami redukcji poboru mocy:

Bit	7	6	5	4	3	2	1	0	
	SE	SM2	SM1	SM0	ISC11	ISC10	ISC01	ISC00	MCUCR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – SE: umożliwienie trybu sleep**

MCU wejdzie w tryb sleep po wykonaniu instrukcji SLEEP, gdy bit SE zostanie ustawiony w stan „1”. Aby uniknąć wprowadzenia w tryb sleep, o ile nie jest to celem programu, należy wpisać jedynkę do SE tuż przed wykonaniem instrukcji SLEEP, i wyzerować go zaraz po wybudzeniu.

- **bity 6...4 – SM2...0: wybór trybu sleep 2,1,0**

wybierają jeden z sześciu trybów redukcji poboru mocy.

SM2	SM1	SM0	Tryb sleep
0	0	0	Idle
0	0	1	ADC z redukcją szumów
0	1	0	Power-down
0	1	1	Power-save
1	0	0	Zarezerwowany
1	0	1	Zarezerwowany
1	1	0	Standby(1)
1	1	1	Extended Standby(1)

(1) - możliwe tylko z zewnętrznym kwarcem lub generatorem kwarcowym.

Idle:

Po stawieniu bitów SM2...0 w 000, instrukcja SLEEP wprowadza MCU w tryb Idle. CPU jest zatrzymywany, natomiast SPI, USART, komparator analogowy, ADC, TWI, liczniki, Watchdog i system przerwań nadal działają. Tryb ten zatrzymuje zegary clk_{CPU}, clk_{FLASH}, podczas gdy pozostałe pracują.

Idle umożliwia wybudzenie MCU przez zewnętrzne przerwania i wewnętrzne takie jak przepełnienie licznika czy USART. Jeśli nie jest wymagane wybudzenie przerwaniem od komparatora analogowego, można komparator wyłączyć ustawiając bit ACD w rejestrze ACSR. Pozwoli to na obniżenie poboru prądu. Gdy przetwornik analogowo-cyfrowy ADC jest włączony, konwersja zostanie uruchomiona automatycznie po wejściu w tryb Idle.

Przetwornik ADC z trybem redukcji szumów:

Po stawieniu bitów SM2...0 w 001, instrukcja SLEEP wprowadza MCU w tryb redukcji szumów ADC. CPU zostaje zatrzymany a ADC, zewnętrzne przerwania, układ czuwania w TWI, liczniki i Watchdog wciąż działają (o ile są włączone). Tryb ten zatrzymuje zegary $clk_{I/O}$, clk_{CPU}, clk_{FLASH}, podczas gdy pozostałe pracują.

Tryb ten umożliwia zwiększenie dokładności pomiaru ADC. Jeżeli przetwornik ADC jest włączony, konwersja zostaje automatycznie uruchomiona po wprowadzeniu układu w ten tryb. MCU może zostać wybudzony przez przerwanie ADC (zakończenie konwersji), zewnętrzny Reset, Watchdog reset, Brown-out reset, przerwanie od układu wykrywającego adres TWI, przerwania z licznika TC2, przerwania zgłoszenia gotowości EEPROM/SPM, zewnętrzne przerwania INT0...2.

Power-down:

Po stawieniu bitów SM2...0 w 010, instrukcja SLEEP wprowadza MCU w tryb Power-down. W trybie tym zewnętrzny oscylator jest zatrzymywany, podczas gdy zewnętrzne przerwania, układ wykrywający adres TWI i Watchdog pracują (o ile są włączone). MCU może zostać wybudzony przez zewnętrzny Reset, Watchdog reset, Brown-out reset, przerwanie od układu wykrywającego adres TWI, zewnętrzne przerwania INT0...2. Zatrzymywane są wszystkie sygnały zegarowe, pracować mogą tylko układy asynchroniczne.

Należy zwrócić uwagę, że gdy przerwanie zewnętrzne wyzwalane poziomem służy do wybudzania układu z trybu Power-down, poziom ten musi trwać pewien okres czasu. (więcej w rozdziale: Zewnętrzne przerwania).

Wybudzenie układu pochłania pewien okres czasu w celu ustabilizowania układu zegarowego.

Power-save:
Po stawieniu bitów SM2...0 w 011, instrukcja SLEEP wprowadza MCU w tryb Power-save. Tryb ten jest identyczny jak Power-down z jednym wyjątkiem:
Jeżeli licznik TC2 jest napędzany asynchronicznie (np. przez ustawienie bitu AS2 w ASSR) licznik będzie nadal działał. MCU może być wybudzony w momencie przepełnienia się licznika lub porównania w liczniku, jeśli odpowiednie bity przerwań licznika TC2 zostaną ustawione (w rejestrze TIMSK) i globalny system przerwań jest włączony (rejestr SREG).
Jeżeli TC2 nie pracuje asynchronicznie, należy używać trybu Power-down zamiast Power-save, ponieważ zawartość rejestrów w liczniku TC2 może się pozmieniać po wybudzeniu z trybu Power-save.
Standby:
Jeżeli bity SM2...0 są ustawione na 110 i jako źródło zegara dla systemu wybrany jest zewnętrzny rezonator kwarcowy / ceramiczny, instrukcja SLEEP wprowadzi MPU w tryb Standby. Tryb ten jest taki sam jak Power-down, z wyjątkiem tego, że oscylator nadal działa. Umożliwia to szybkie wybudzenie układu w ciągu 6 cykli zegara.
Extended Standby:
Jeżeli bity SM2...0 są ustawione na 111 i jako źródło zegara dla systemu wybrany jest zewnętrzny rezonator kwarcowy / ceramiczny, instrukcja SLEEP wprowadzi MPU w tryb Extended Standby. Tryb ten jest taki sam jak Power-save, z wyjątkiem tego, że oscylator nadal działa. Umożliwia to szybkie wybudzenie układu w ciągu 6 cykli zegara.

Stan sygnałów zegarowych w rożnych trybach Sleep:

	Sygnały zegarowe					oscylatory		Źródła wybudzania					
Tryb Sleep	clkCPU	clkFLASH	clkI/0	clkADC	clkASY	Główne źródło sygnału zegatrowego	Oscylator timera	INT2 INT1 INT0	Wykrywanie adresu TWI	Timer2	Gotowość SPM / EEPROM	ADC	Pozostałe I/O
Idle			X	X	X	X	X(2)	X	X	X	X	X	X
redukcja szum. ADC				X	X	X	X(2)	X(3)	X	X	X	X	
Power-down								X(3)	X				
Power-save					X(2)		X(2)	X(3)	X	X(2)			
Standby(1)						X		X(3)	X				
Extended Standby(1)					X(2)	X	X(2)	X(3)	X	X(2)			

(1) – wybrany zewnętrzny kwarc lub generator jako źródło sygnału zegarowego
(2) – gdy AS2 jest ustawiony w rejestrze ASSR
(3) – tylko INT2 lub przerwanie poziomem na INT1 i INT0

Zminimalizowanie poboru mocy:
Istnieje kilka czynników, które należy wziąć pod uwagę podczas redukowania poboru mocy w systemach wykorzystujących AVR. Tryby Sleep powinny być używane tak często jak to możliwe. Wszystkie funkcje, które nie są potrzebne powinny być wyłączone. W szczególności poniższe moduły wymagają specjalnego traktowania:
Przetwornik ADC:
Jeśli jest włączony, ADC będzie aktywny w każdym trybie Sleep. W celu obniżenia mocy przetwornik powinien być wyłączony przed wprowadzeniem w któryś z trybów Sleep. Jeżeli przetwornik będzie wyłączony i ponownie włączony, następna konwersja będzie trwała dłużej. (więcej w rozdziale: przetwornik ADC).
Komparator analogowy:
Przy wprowadzaniu w tryb Idle komparator analogowy powinien być wyłączony jeśli nie jest używany. Podczas wprowadzania w tryb redukcji szumów ADC, komparator należy wyłączyć. W pozostałych trybach Sleep, komparator jest automatycznie wyłączony. (więcej w rozdziale: komparator analogowy).
Brown-out:
Jeżeli detektor Brown-out nie jest potrzebny w aplikacji, powinien być wyłączony. Jeżeli jest włączony przez ustawienie fuse-bitów BODEN, będzie aktywny we wszystkich trybach Sleep, powodując niewielkie zwiększenie poboru prądu.
Wewnętrzne źródło napięcia odniesienia:
Powinno być włączone gdy wymagane jest w układach Brown-out, komparatorze analogowym lub przetworniku ADC. Jeżeli układy te są wyłączone tak jak opisano to powyżej, źródło napięcia odniesienia też będzie wyłączone i nie będzie zużywać energii. Gdy zostanie włączone, powinno być użyte dopiero gdy źródło to się całkowicie uruchomi. Jeżeli źródło odniesienia jest włączone w trybach Sleep, napięcie na końcówce może być nadal używane.

Układ Watchdog:
Jeżeli układ watchdog nie jest potrzebny w aplikacji, powinien być wyłączony. Jeżeli jest włączony, będzie aktywny we wszystkich trybach Sleep.
Końcówki portów:
Powinny być ustawione dla minimalnego poboru mocy. Większy pobór prądu powoduje obciążenie wyjść rezystancją. W trybie Sleep, gdy sygnały zegarowe I/O ($clk_{I/O}$) i ADC (clk_{ADC}) są zatrzymane, bufory wejściowe układu będą wyłączone. Dzięki temu obniża się zużycie mocy przez logikę wejściową gdy nie jest używana. W niektórych przypadkach wyjścia są wykorzystywane do wykrycia sygnału budzącego (wake-up) i wtedy są włączone.
On-Chip Debug:
Gdy system On-Chip Debug (OCD) jest włączony przez fuse-bit OCDEN i układ jest wprowadzony w tryb Sleep, sygnał głównego zegara jest nadal aktywny.
Są trzy metody na wyłączenia OCD:

- wyłączenie fuse-bit OCDEN
- wyłączenie fuse-bitu JTAGEN
- wpisanie jedynki do bitu JTD w rejestrze MCUCSR

Sterowanie i reset systemu:

Resetowanie AVR:
Podczas resetu we wszystkich rejestrach I/O ustawia się wartość początkowa a program zaczyna się wykonywać od adresu wektora reset. Instrukcją umieszczoną w wektorze Reset musi być instrukcja skoku JMP – skok bezwzględny do programu. Jeżeli program nigdy nie będzie używał przerwań, wektory przerwań nie będą wtedy używane i program może być położony w ich obszarze. Jest to możliwe również wtedy, gdy wektor resetu położony jest w obszarze Boot.
Porty I/O są natychmiast ustawiane do wartości początkowej.
Po resecie uaktywniany jest licznik opóźniający uruchomienie układu. Pozwala to na ustabilizowanie się układu przed normalną pracą. Fuse-bity CKSEL decydują o opóźnieniu wywołanym tym licznikiem.
Źródła sygnału reset:
ATmega32 posiada pięć źródeł sygnału reset:

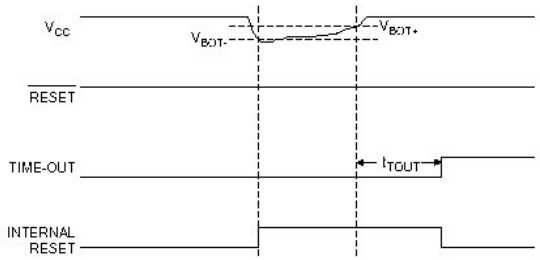
- power-on reset: MCU będzie zresetowany gdy napięcie zasilające obniży się poniżej progu (V_{POT})
- External Reset: MCU zostaje wyzerowany gdy na końcówce RESET pojawi się stan niski (o czasie trwania dłuższym niż wartość minimalna).
- Watchdog-reset: MCU wyzeruje się gdy zostanie przekroczony okres licznika watchdog (gdy został włączony)
- Brown-out Reset: zeruje MCU gdy napięcie zasilające spadnie poniżej poziomu V_{BOT} (gdy detektor ten jest włączony)
- JTAG AVR Reset: MCU jest w stanie reset tak długo jak w resestrze ResetRegister jest logiczna jedynka

Charakterystyka Reset:

symbol	Parametr	warunek	min	typ	max	jednostka
V_{POT}	Power-on Reset Próg narastania napięcia			1,4	2,3	V
	Power-on Reset Próg opadania napięcia(1)			1,3	2,3	V
V_{RST}	Próg napięcia na końcówce RESET		0,2 V_{CC}		0,85 V_{CC}	V
t_{RST}	Minimalny czas trwania sygnału na k. RESET			50		ns
V_{BOT}	Próg napięcia dla Brown-out (2)	BODLEVEL=1	2,5	2,7	3,2	V
		BODLEVEL=0	3,7	4,0	4,2	
t_{BOT}	Minimalny okres stanu niskiego dla Brown-out	BODLEVEL=1		2		us
		BODLEVEL=0		2		us
V_{HYST}	Histereza detektora Brown-out			50		mV

1. Power-on Reset nie będzie działał gdy napięcie zasilające będzie poniżej V_{POT}
2. V_{BOT} może być poniżej minimalnego napięcia zasilania układu.

Power-on reset:
Impuls Power-on Reset (POR) jest generowany przez układ wykrywający poziom napięcia zasilającego.
External reset:
Zewnętrzny Reset jest generowany gdy na końcówce RESET jest stan niski. Sygnał ten dłuższy od minimalnego okresu wygeneruje sygnał Reset nawet jeśli zegar systemowy nie pracuje. Krótsze impulsy nie gwarantują poprawnego wyzerowania procesora.
Detektor Brown-out:
ATmega32 posiada wbudowany detektor Brown-out kontrolujący napięcie zasilające. Poziom wyzwalania tego układu resetującego jest ustalany przez fuse-bity BODLEVEL na 2,7V (BODLEVEL nie zaprogramowane) i 4,0 V (BODLEVEL zaprogramowane). Układ posiada histerezę: $V_{BOT+}=V_{BOT}+V_{HYST/2}$ i $V_{BOT-}=V_{BOT}-V_{HYST/2}$.
Układ BOD może być włączony lub wyłączony przez fuse-bit BODEN. Gdy BOD jest aktywne i V_{CC} opada poniżej progu wyzwalania (V_{BOT-}), układ BOD zareaguje natychmiast zerując mikrokontroler. Gdy napięcie wzrośnie powyżej poziomu wyzwalania (V_{BOT+}) licznik opóźniający uruchamia procesor dopiero po upłynięciu czasu t_{TOUT}. Układ BOD zadziała tylko wtedy gdy napięcie spadnie poniżej określonego poziomu przynajmniej na czas t_{BOD}.



Watchdog:
Impuls Reset jest wytwarzany na jeden cykl zegara gdy minie okres czasu licznika Watchdog. Przy opadającym zboczu tego impulsu jest uruchamiany licznik opóźniający start systemu.

Rejestr kontroli MCU – MCUCSR:
Rejestr kontroli MCUCSR dostarcza informacji o tym które źródło reset spowodowało wyzerowanie MCU

Bit	7	6	5	4	3	2	1	0	
	JTD	ISC2	-	JTRF	WDRF	BORF	EXTRF	PORF	MCUCSR
Odczyt/zapis	Z/O	Z/O	O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	opisane poniżej					

- **bit 4 – JTRF: flaga reset JTAG**

bit jest ustawiany, gdy reset wystąp przez wybranie AVR_RESET w JTAG. Kasowany jest automatycznie przez Power-on Reset lub przez wpisanie logicznego zera.

- **bit 3 – WDRF: Flaga Watchdog Reset**

Jedynka oznacza, że wystąpił reset od układu Watchdog. Kasowany przez Power-on lub przez wpisanie logicznego zera.

- **bit 2 – BORF: flaga reset Brown-out**

Jedynka oznacza, że wystąpił reset od układu Brown-out. Kasowany przez Power-on lub przez wpisanie logicznego zera.

- **bit 1 – EXTRF: flaga wystąpienia resetu zewnętrznego**

Jedynka oznacza, że wystąpił zewnętrzny sygnał reset. Kasowany przez Power-on lub przez wpisanie logicznego zera.

- **bit 0 – PORF: flaga resetu Power-on**

Jedynka oznacza, że wystąpił reset przy włączaniu. Kasowany tylko przez wpisanie logicznego zera.

Aby stwierdzić źródło sygnału reset należy odczytać zawartość rejestru MCUCSR i wyzerować go.

Wewnętrzne napięcie odniesienia:
Napięcie to jest używane przez układ Brown-out i może być użyte jako wejście komparatora analogowego lub dla przetwornika ADC (napięcie odniesienia 2,56V).
Źródło uruchamia się przez pewien czas podany w tabeli. Jest włączone podczas następujących sytuacji:

1. gdy BOD jest włączone (przez zaprogramowanie fuse-bitu BODEN)
2. gdy jest dołączone do komparatora analogowego (ustawienie bitu ACBG w rejestrze ACSR)
3. gdy przetwornik ADC jest aktywny.

Gdy BOD nie jest aktywne, po ustawieniu bitu ACBG lub włączeniu przetwornika użytkownik musi zawsze odczekać na uruchomienie się źródła zanim komparator analogowy czy przetwornik ADC się uruchomi.

Charakterystyka źródła napięcia odniesienia:

symbol	parametr	min	typ	max	jednostka
V_{BG}	Napięcie odniesienia	1,15	1,23	1,35	V
t_{BG}	Czas uruchamiania źródła napięcia		40	70	us
I_{BG}	Pobór prądu przez źródło		10		uA

Czasomierz Watchdog:

Układ Watchdog jest napędzany osobnym wewnętrznym oscylatorem pracującym przy 1 MHz. Jest to typowa wartość dla V_{CC}=5V. Preskaler Watchdog umożliwia ustalenie okresu czasu po którym nastąpi Reset jeżeli instrukcja WDR (reset Watchdog) nie zostanie wykonana. Czasomierz Watchdog zostanie również wyzerowany gdy się go wyłączy lub gdy nastąpi wyzerowanie całego mikrokontrolera. Okres czasu Watchdog można ustawić na osiem różnych wartości.

Rejestr kontroli Watchdog – WDTCR:

Bit	7	6	5	4	3	2	1	0	
	-	-	-	WDTOE	WDE	WDP2	WDP1	WDP0	WDTCR
Odczyt/zapis	O	O	O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- bity 7...5 – zarezerwowane

wartością odczytaną zawsze będzie zero.

- bit 4 – WDTOE: uaktywnienie wyłączania Watchdog

musi być ustawiony gdy do bitu WDE jest wpisywane zero. Inaczej Watchdog nie będzie wyłączony.

- bit 3 – WDE: uaktywnienie Watchdog

jeżeli do WDE wpisze się jedynkę Watchdog zostanie włączony. Jeśli wpisze się zero funkcje Watchdog zostaną wyłączone. WDE może być tylko wyzerowany gdy bit WDTOE jest ustawiony. Procedura do włączania i wyłączania układu Watchdog:

1. W tej samej operacji wpisać jedynkę do WDTOE i WDE. Jedynka musi być wpisana do WDE nawet wtedy gdy bit ten był wcześniej ustawiony.
2. w ciągu następnych czterech cykli wpisać zero do WDE. To wyłącza Watchdog.

- bity 2...0 – WDP2, WDP1, WDP0: preskaler licznika Watchdog:

bity te ustalają podział zegara dla układu Watchdog (gdy jest aktywny).

WDP2	WDP1	WDP0	Liczba cykli oscylatora Watchdog	Typowy czas do zresetowania przy V_{CC}=3V	Typowy czas do zresetowania przy V_{CC}=5V
0	0	0	16K (16384)	17,1 ms	16,3 ms
0	0	1	32K (32768)	34,3 ms	32,5 ms
0	1	0	64K (65536)	68,5 ms	65 ms
0	1	1	128K (131072)	0,14 s	0,13 s
1	0	0	256K (262144)	0,27 s	0,26 s
1	0	1	512K (524288)	0,55 s	0,52 s
1	1	0	1024K (1048576)	1,1 s	1,0 s
1	1	1	2048K (2097152)	2,2 s	2,1 s

Poniższy przykład pokazuje jak wyłączyć Watchdog (pod warunkiem, że w czasie wykonywania tej procedury nie wystąpi przerwanie):

Przykład w asemblerze

```
WDT_off:
        ; wpisanie logicznej jedynki do WDTOE i WDE
ldi r16, (1<<WDTOE)|(1<<WDE)
out WDTCR, r16
        ; wyłączenie WDT
ldi r16, (0<<WDE)
out WDTCR, r16
ret
```

Przykład w C

```
void WDT_off(void)
{       /* wpisanie logicznej jedynki do WDTOE i WDE */
WDTCR = (1<<WDTOE) | (1<<WDE);
        /* wyłączenie WDT */
WDTCR = 0x00;
}
```

Przerwania:

Wektory przerwań:

wektor	Adres w pamięci programu	Źródło	Rodzaj przerwania
1	$000	RESET	Sygnał z zewnątrz, reset po włączeniu, reset Brown-out, reset Watch-dog, reset JTAG AVR
2	$002	INT0	Zewnętrzne przerwanie 0
3	$004	INT1	Zewnętrzne przerwanie 1
4	$006	INT2	Zewnętrzne przerwanie 2
5	$008	TIMER2 COMP	Porównanie w TIMER 2
6	$00A	TIMER2 OVF	Przepełnienie w TIMER 2
7	$00C	TIMER1 CAPT	Przechwycenie w TIMER 1
8	$00E	TIMER1 COMPA	Porównanie A w TIMER 1
9	$010	TIMER1 COMPB	Porównanie B w TIMER 1
10	$012	TIMER1 OVF	Przepełnienie TIMER 1
11	$014	TIMER0 COMP	Porównanie TIMER 0
12	$016	TIMER0 OVF	Przepełnienie w TIMER 0
13	$018	SPI, STC	Zakończenie transmisji przez SPI
14	$01A	USART, RXC	Zakończenie odbierania przez USART
15	$01C	USART, UDRE	Pusty rejestr danych USART
16	$01E	USART, TXC	Zakończenie wysyłania przez USART
17	$020	ADC	Zakończenie przetwarzania przez ADC
18	$022	EE_RDY	Stan gotowości EEPROM
19	$024	ANA_COMP	Komparator analogowy
20	$026	TWI	Interfejs I^2C
21	$028	SPM_RDY	Gotowość na umieszczenie programu

Jeżeli program nigdy nie będzie wykorzystywał przerwań, może być umieszczony w obszarze wektorów. Możliwe jest to również wtedy gdy wektor przerwania Reset znajduje się w obszarze pamięci programu dla aplikacji a pozostałe przerwania w obszarze Boot.

Umiejscowienie wektora przerwania Reset

BOOTRST	IVSEL	Adres resetu	Początek adresów wektorów przerwań
1	0	$0000	$002
1	1	$0000	Boot Reset + $0002
0	0	Boot Reset	$0002
0	1	Boot Reset	Boot Reset + $0002

Większość programów ustawiających przerwania będzie wyglądała następująco:

```
adresy etykiety   program                  opis

$000              jmp     RESET            ; obsługa reset
$002              jmp     EXT_INT0         ; obsługa IRQ0
$004              jmp     EXT_INT1         ; obsługa IRQ1
$006              jmp     EXT_INT2         ; obsługa IRQ2
$008              jmp     TIM2_COMP        ; obsługa porównania Timer2
$00A              jmp     TIM2_OVF         ; obsługa przepełnienia Timer2
$00C              jmp     TIM1_CAPT        ; obsługa przechwycenia Timer1
$00E              jmp     TIM1_COMPA       ; obsługa porównania A Timer1
$010              jmp     TIM1_COMPB       ; obsługa porównania B Timer1
$012              jmp     TIM1_OVF         ; obsługa przepełnienia Timer1
$014              jmp     TIM0_COMP        ; obsługa porównania Timer0
$016              jmp     TIM0_OVF         ; obsługa przepełnienia Timer0
$018              jmp     SPI_STC          ; obsługa SPI Transfer Complete
$01A              jmp     USART_RXC        ; obsługa USART RX Complete
$01C              jmp     USART_UDRE       ; obsługa UDR Empty
$01E              jmp     USART_TXC        ; obsługa USART TX Complete
$020              jmp     ADC              ; obsługa ADC Conversion Complete
$022              jmp     EE_RDY           ; obsługa EEPROM Ready
$024              jmp ANA_COMP             ; obsługa Analog Comparator
$026              jmp TWI                  ; obsługa Two-wire Serial Interface
$028              jmp SPM_RDY              ; obsługa Store Program Memory Ready
;
$02A RESET:       ldi r16,high(RAMEND)     ; początek programu głównego
$02B              out SPH,r16              ; ustawienie stosu na koniec RAMu
$02C              ldi r16,low(RAMEND)
$02D              out SPL,r16
$02E              sei ; Enable interrupts
$02F              <instr> xxx
...               ...                ...
```

Jeśli buse-bit BOOTRST jest niezaprogramowany, obszar Boot jest ustawiony na 4KB i bit IVSEL w rejestrze GICR jest ustawiany przed włączeniem systemu przerwań, początek programu ustawiającego reset i wektory przerwań wyglądać będzie następująco:

```
adresy etykiety   program                  opis

$000 RESET:       ldi r16,high(RAMEND)     ; początek programu głównego
$001              out SPH,r16              ; ustawienie stosu na koniec RAMu
$002              ldi r16,low(RAMEND)
$003              out SPL,r16
$004              sei                      ; włączanie przerwań
$005              <instr> xxx
;
.org $3802
$3802             jmp EXT_INT0             ; obsługa IRQ0
$3804             jmp EXT_INT              ; obsługa IRQ1
... .... .. ;
$3828             jmp SPM_RDY              ; Store Program Memory Ready Handler
```

Jeśli fusebit BOOTRST jest zaprogramowany a obszar Boot jest ustawiony na 4KB, początek programu ustawiającego reset i wektory przerwań wyglądać będzie następująco:

```
adresy etykiety   program                 opis
.org $002
$002              jmp EXT_INT0            ; IRQ0 Handler
$004              jmp EXT_INT1            ; IRQ1 Handler
... .... .. ;
$028              jmp SPM_RDY             ; Store Program Memory Ready Handler
;
.org $3800
$3800 RESET:      ldi r16,high(RAMEND)    ; Main program start
$3801             out SPH,r16             ; Set Stack Pointer to top of RAM
$3802             ldi r16,low(RAMEND)
$3803             out SPL,r16
$3804             sei                     ; Enable interrupts
$3805             <instr> xxx
```

Jeśli buse-bit BOOTRST jest zaprogramowany, obszar Boot jest ustawiony na 4KB i bit IVSEL w rejestrze GICR jest ustawiany przed włączeniem systemu przerwań, początek programu ustawiającego reset i wektory przerwań wyglądać będzie następująco:

```
adresy etykiety   program                 opis
.org $3800
$3800             jmp RESET               ; Reset handler
$3802             jmp EXT_INT0            ; IRQ0 Handler
$3804             jmp EXT_INT1            ; IRQ1 Handler
... .... .. ;
$3828             jmp SPM_RDY             ; Store Program Memory Ready Handler
;
$382A RESET:      ldi r16,high(RAMEND)    ; Main program start
$382B             out SPH,r16             ; Set Stack Pointer to top of RAM
$382C             ldi r16,low(RAMEND)
$382D             out SPL,r16
$382E             sei                     ; Enable interrupts
$382F             <instr> xxx
```

Przemieszczanie przerwań pomiędzy obszarami aplikacji i Boot
Rejestr kontroli przerwań – GICR:

Bit	7	6	5	4	3	2	1	0	
	INT1	INT0	INT2	-	-	-	IVSEL	IVCE	GICR
Odczyt/zapis	Z/O	Z/O	Z/O	O	O	O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 1 – IVSEL:**

wyzerowanie bitu powoduje, że wektory przerwań są położone na początku pamięci Flash. Gdy w bicie jest jedynka, wektory przerwań są przesunięte na początek obszaru Flash boot-loader. Adres w tym obszarze ustalany jest fuse-bitami BOOTSZ. Procedura poprawnej zmiany adresu Boot:

1. ustawić bit zmiany wektorów przerwań IVCE
2. w czasie czterech cykli, wpisać żądaną wartość do IVSEL i zero do IVCE

podczas wykonywania tej procedury, przerwania będą automatycznie wyłączane (w momencie ustawiania IVCE).

- **bit 0 – IVCE:**

zmiany IVSEL będą widoczne dopiero po ustawieniu bitu IVCE. Jest kasowany sprzętowo po czterech cyklach od wpisania do niego lub podczas zapisu do IVSEL. Ustawienie IVCE wyłącza przerwania.

Przykład w asemblerze:

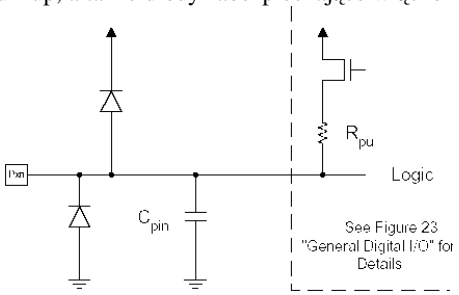
```
        ; uaktywnienie zmian wektorów przerwań:
ldi r16 , (1<<IVCE)
out GICR , r16
        ; przesunięcie przerwań do obszaru Boot:
ldi r16 , (1<<IVSEL)
out GICR , r16
ret
```

Przykład w C:

```
Void przesuwanie_przerwan (void)
{
        /*  uaktywnienie zmian wektorów przerwań */
        GICR = (1<<IVCE);
        /* przesunięcie przerwań do obszaru Boot*/
        GICR = (1<<IVSEL);
}
```

Porty I/O:

Porty AVR działają na zasadzie odczyt-modyfikacja-zapis gdy działają jako cyfrowe porty we/wy. Kierunek jednej końcówki portu może być zmieniony za pomocą instrukcji CBI i SBI bez zmiany kierunku pozostałych końcówek. Wyjścia mają odpowiednią moc do sterowania diodami LED. Każda końcówka portu posiada niezależnie wybierane rezystory pull-up, a także diody zabezpieczające włączone między masę a zasilanie.



każdy port ma trzy rejestry ulokowane w obszarze I/O: rejestr danych – PORTx, rejestr kierunku – DDRx i wejście portu PINx. PIN jest tylko do odczytu, PORT i DDR do odczytu i zapisu. Większość końcówek posiada dodatkowe funkcje związane z wbudowanymi urządzeniami.

Uaktywnienie niektórych dodatkowych funkcji końcówek w portach ogranicza stosowanie pozostałych końcówek tego portu jako ogólnego portu we/wy.

Cyfrowe porty we/wy:

Są to dwukierunkowe porty we/wy z dodatkowymi wewnętrznymi rezystorami podciągającymi.

Rejestr DDR służy do określenia kierunku portu. Gdy dany bit rejestru DDR jest ustawiony w stan wysoki, końcówka odpowiadająca numerowi tego bitu staje się wyjściem, wpisanie zera ustawia końcówkę jako wejście.

Gdy do bitu rejestru PORT wpisana jest jedynka podczas gdy port ustawiony jest jako wejście, końcówka jest podciągana do zasilania przez wewnętrzny rezystor pull-up. Wyłączenie rezystorów następuje przez wpisanie do odpowiednich bitów PORT zer. Końcówki portów są trzystanowe, gdy jest aktywny reset.

Gdy do PORTxn wpisana jest jedynka i port ustawiony jest jako wyjście, na końcówce jest stan wysoki. Wpisanie zera wymusza stan niski.

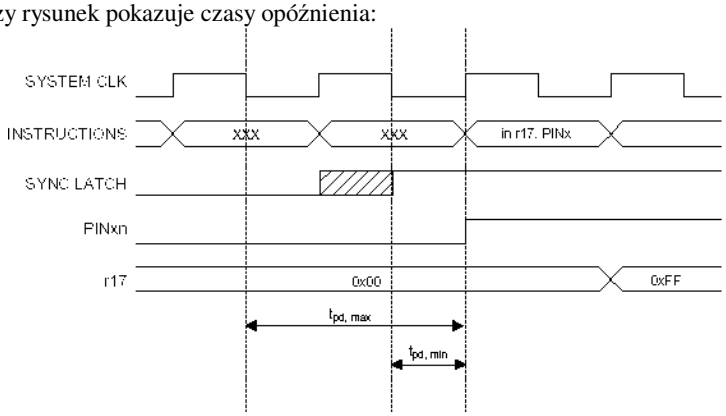
Zazwyczaj dla układów cyfrowych o wysokiej impedancji wejściowej nie ma różnicy między wymuszeniem stanu wysokiego (końcówka jest wyjściem ze stanem logicznym „1") a wejściem z podciąganiem rezystorowym pull-up. W każdej chwili można wyłączyć wszystkie rezystory podciągające we wszystkich portach przez odpowiednie ustawienie bitu PUD w rejestrze SFIOR.

Podczas przełączania między satnem: wejście z pull-up a wyjście ze stanem niskim, port będzie na krótką chwilę pracował jako: wejście trójstanowe lub wyjście ze stanem wysokim.

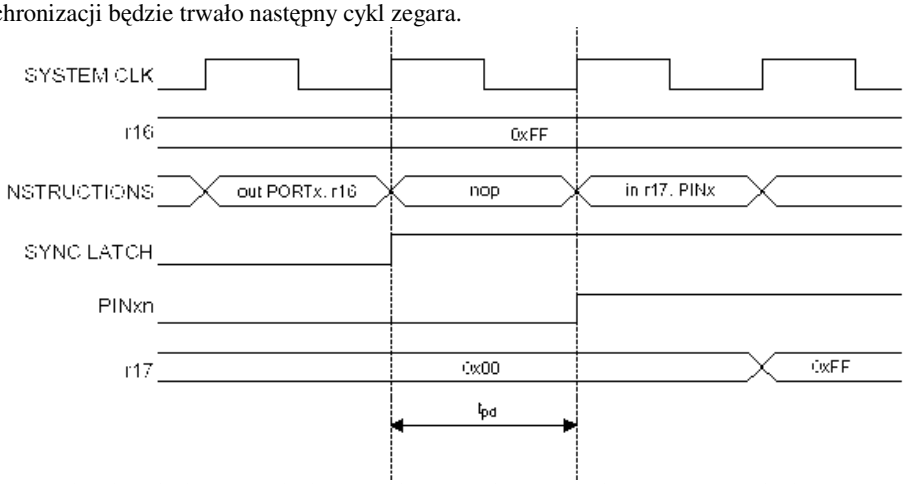
Konfiguracja końcówek:

DDRxn	PORTxn	PUD (w SFIOR)	I/O	Pull-up	Opis
0	0	x	Wejście	-	Trójstanowe (wysoka impedancja wejścia)
0	1	0	Wejście	√	Końcówka jest źródłem prądowym
0	1	1	Wejście	-	Trójstanowe (wysoka impedancja wejścia)
1	0	x	Wyjście	-	Wyjście w stanie niskim
1	1	x	Wyjście	-	Wyjście w stanie wysokim

Niezależnie od ustawienia rejestru kierunku portu DDRxn stan końcówki można odczytać z rejestru PINxn. Rejestr ten jest synchronizowany z przerzutnikiem, w celu uzyskania stabilnego sygnału podczas odczytu, gdy stan na końcówce zmienia się z prędkością bliską wewnętrznemu zegarowi. Wprowadza się jednak w ten sposób opóźnienie. Poniższy rysunek pokazuje czasy opóźnienia:



Przerzutnik zatrzaskuje się gdy sygnał zegarowy jest w stanie niskim i przepuszcza w stanie wysokim, co widać na zakreskowanym polu sygnały SYNC_LATCH. Poziom sygnału staje się zatrzaśnięty przy opadającym zboczu. Przekazany jest do rejestru PIN z narastającym zboczem sygnału. Całkowite opóźnienie wynosi od ½ do 1 ½ taktu zegara systemowego.
Jeżeli końcówka jest odczytywana po wcześniejszym ustawieniu jej stanu, pomiędzy OUT a IN musi znaleźć się instrukcja NOP. Instrukcja OUT ustawia sygnał SYNC_LATCH przy narastającym zboczu zegara. Opóźnienie w celu synchronizacji będzie trwało następny cykl zegara.



Poniższy przykład pokazuje jak ustawić końcówki 0, 1, 2 i 3 w porcie B do stanu niskiego a końcówki 4 – 7 jako wejście z podciąganiem pull-up na nóżkach 6 i 7. Stan tych końcówek będzie następnie odczytany (z uwzględnieniem opóźnienia NOP).

Przykład w asemblerze

```
...
        ; ustawienie pull-up i stanu wyjść w stanie wysokim
        ; ustawianie kierunku portu
ldi r16,(1<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0)
ldi r17,(1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0)
out PORTB,r16
out DDRB,r17
        ; instrukcja NOP dla synchronizacji
nop
        ; odczyt z końcówek portu
in r16,PINB
...
```

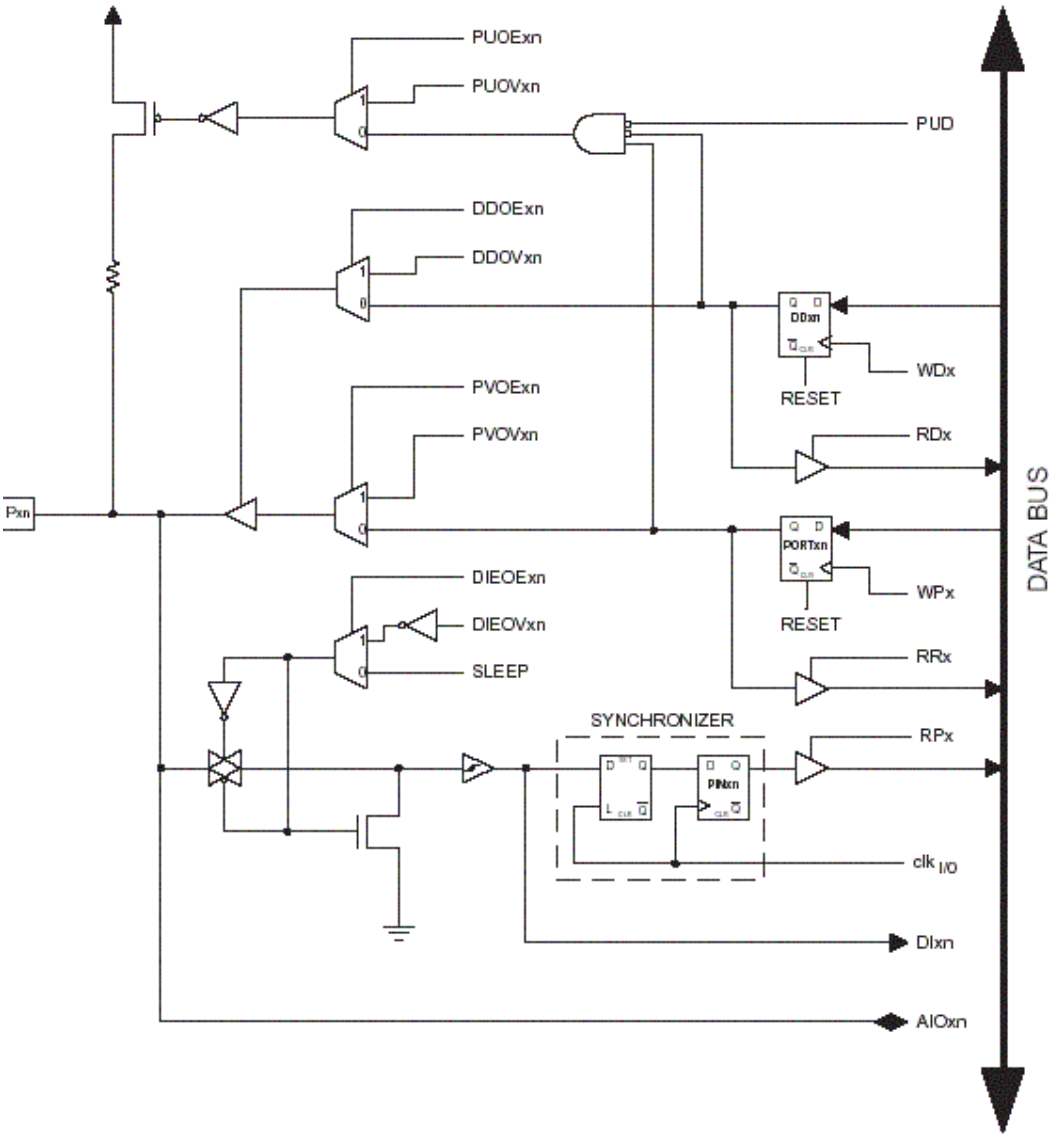
Przykład w C

```
unsigned char i;
...
        /* ustawienie pull-up i stanu wyjść w stanie wysokim */
        /* ustawianie kierunku portu */
PORTB = (1<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0);
DDRB = (1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0);
        /* instrukcja NOP dla synchronizacji */
_NOP();
        /* odczyt z końcówek portu */
i = PINB;
...
```

W przykładzie asemblerowym dwa tymczasowe rejestry r16 i r17 zostały użyte dla skrócenia czasu ustawiania portu.

Końcówki niepodłączone:
Wymagane jest aby nie używane końcówki miały określony. Mimo że większość wejść w trybie sleep jest wyłączone, powinno unikać się wiszących końcówek wejściowych w celu redukcji poboru prądu w pozostałych trybach pracy.
Najprostszą metodą jest włączenie wewnętrznych rezystorów pull-up0Rezystory te będą odłączane w czasie trwania Reset. Gdy wymagany jest niski pobór prądu podczas resetu, można podłączyć zewnętrzne rezystory podciągające (w górę lub w dół). Podłączanie końcówek bezpośrednio do Vcc lub GND nie jest zalecane, ze względu na możliwość przepływu dużego prądu gdy końcówka ustawiona zostanie jako wyjście.

Alternatywne funkcje portu:
Większość końcówek posiada dodatkowe funkcje.



1. WPx, WDx, RRx, RPx, RDx są podłączone do wszystkich końcówek w tym samym porcie. $clk_{I/O}$, SLEEP i PUD są podłączone do wszystkich portów. Pozostałe sygnały dotyczą pojedynczych końcówek.

Opis sygnałów:

Nazwa sygnału	Opis
PUOE	Jeśli jest ustawiony, rezystory podciągające są połączone przez sygnał PUOV. Jeżeli jest wyzerowany podciąganie jest aktywne gdy {DDxn, PORTxn, PUD} = 0b 0 1 0
PUOV	Gdy PUOE jest ustawione, podciąganie jest włączone / wyłączone dla PUOV ustawionego / wyzerowanego, niezależnie od ustawień rejestrów DDxn, PORTxn i PUD.
DDOE	Jeżeli jest ustawiony, sterownik wyjścia jest sterowany przez sygnał DDOV. Jeśli jest wyzerowany, sterownik wyjścia jest włączany przez rejestr DDR
DDOV	Gdy sygnał ten jest ustawiony i sterownik wyjścia jest włączony wartość portu zależy od sygnału PVOV. Gdy PVOE jest skasowane, sterownik wyjścia jest włączony i jest sterowany przez rejestr PORT.
PVOE	Gdy PVOE jest ustawiony, wartość portu jest ustawiana na PVOV niezależnie od ustawień rejestru PORTxn.
PVOV	Gdy jest ustawiony, włączanie wejść cyfrowych jest sterowane przez sygnał DIEOV. Gdy jest wyzerowany, włączanie wejść cyfrowych jest zależne od stanu mikrokontrolera (tryb normalny, sleep).
DIEOV	Gdy DIEOE jest ustawione, cyfrowe wejścia są włączone/ wyłączone gdy DIEOV jest włączone / wyłączone, niezależnie od stanu MCU.
DI	Cyfrowe wejście alternatywnych funkcji. Sygnał jest dołączony do wyjścia przez układ synchronizujący i przerzutnik Schmitta.
AIO	Analogowe wejście / wyjście do / z alternatywnych funkcji. Sygnał ten jest dołączony bezpośrednio do szyny i może być użyty dwukierunkowo.

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	-	ACME	PUD	PSR2	PSR10	SFIOR
Odczyt/zapis	Z/O	Z/O	Z/O	O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 2 – PUD: wyłączenie podciągania:**

gdy bit ten jest ustawiony, wewnętrzne rezystory podciągające są wyłączone nawet gdy rejestry DDxn i PORTxn są ustawione do podciągania wyjścia.

Funkcje alternatywne portu A:

Funkcją dodatkową portu A są wejścia analogowe dla przetwornika ADC. Jeżeli któraś z końcówek portu jest wtedy używana jako wyjście nie powinno przełączać się tych końcówek w trakcie przetwarzania ze względu na możliwość pojawienia się błędów w pomiarze.

Końcówka portu	Dodatkowa funkcja
PA7	ADC7 (kanał wejściowy 7)
PA6	ADC7 (kanał wejściowy 6)
PA5	ADC7 (kanał wejściowy 5)
PA4	ADC7 (kanał wejściowy 4)
PA3	ADC7 (kanał wejściowy 3)
PA2	ADC7 (kanał wejściowy 2)
PA1	ADC7 (kanał wejściowy 1)
PA0	ADC7 (kanał wejściowy 0)

Funkcje alternatywne portu B:

Końcówka portu	Dodatkowa funkcja
PB7	SCK (zegar dla SPI)
PB6	MISO (master wejście / slave wyjście – SPI)
PB5	MOSI (master wyjście / slave wejście – SPI)
PB4	SS (wejście wyboru slave – SPI)
PB3	AIN1 (wejście odwracające (-) komparatora analogowego OC0 – wyjście porównania licznika TC0
PB2	AIN0 (wejście nieodwracające (+) komparatora analogowego INT2 – wejście zewnętrznego przerwania
PB1	T1 (wejście zliczające – zegarowe licznika TC1)
PB0	T0 (wejście zliczające – zegarowe licznika TC0) XCK (wejście / wyjście zewnętrznego zegara USART)

- **SCK – Port B, bit 7**

SCK: wyjście zegara master, wejście zegara slave. Gdy SPI jest włączone i ustawione jako slave, końcówka jest skonfigurowana jako wejście niezależnie od ustawień DDB7. Gdy SPI jest włączone i ustawione jako master, kierunek końcówki jest zależny od DDB7. Gdy końcówka pracuje jako wejście dla SPI, wewnętrzne podciąganie jest zależne od bitu PORTB7

- **MISO – Port B, bit 6**

master wejście / slave wyjście dla SPI. Gdy SPI jest włączone i ustawione jako master, końcówka jest wejściem niezależnie od bitu DDB6. Gdy SPI jest włączone i ustawione jako slave, kierunek końcówki jest sterowany przez DDB6. . Gdy końcówka pracuje jako wejście dla SPI, wewnętrzne podciąganie jest zależne od bitu PORTB6

- **MOSI – Port B, bit 5**

Wyjście danych master / wejście danych slave w SPI. Gdy SPI jest włączone i ustawione jako slave, końcówka jest skonfigurowana jako wejście niezależnie od ustawień DDB5. Gdy SPI jest włączone i ustawione jako master, kierunek końcówki jest zależny od DDB5. Gdy końcówka pracuje jako wejście dla SPI, wewnętrzne podciąganie jest zależne od bitu PORTB5

- **SS – Port B, bit 4**

wejście wyboru slave – SPI. Gdy SPI jest włączone i ustawione jako slave, końcówka jest skonfigurowana jako wejście niezależnie od ustawień DDB4.SPI jest uaktywniane, gdy na końcówkę podawany jest stan niski. Gdy SPI jest włączone i ustawione jako master, kierunek końcówki jest zależny od DDB4. Gdy końcówka pracuje jako wejście dla SPI, wewnętrzne podciąganie jest zależne od bitu PORTB4.

- **AIN1/OC0 – Port B, bit 3**

AIN1 – wejście odwracające (-) komparatora analogowego. Gdy końcówka jest ustawiona jako wejście komparatora, wewnętrzne podciąganie pull-up powinno być wyłączone.
OC0 – wyjście porównania licznika TC0, końcówka powinna być ustawiona jako wyjście (DBB3 = 1), również podczas pracy licznika w trybie PWM.

- **AIN0/INT2 – Port B, bit 2**

AIN1 – wejście nieodwracające (+) komparatora analogowego. Gdy końcówka jest ustawiona jako wejście komparatora, wewnętrzne podciąganie pull-up powinno być wyłączone.
Źródło zewnętrznego przerwania 2: końcówka może służyć jako wejście przerwania MCU.

- **T1 – Port B, bit 1**

wejście zliczające – zegarowe licznika TC1

- **T0/XCK – Port B, bit 0**

wejście zliczające – zegarowe licznika TC0
XCK, zewnętrzny zegar USART. Rejestr kierunku (DDB0) określa czy zegar jest wyjściem (DDB ustawione), czy wejściem (DDB wyzerowane). XCK jest aktywne tylko gdy USART działa w trybie synchronicznym.

Funkcje alternatywne portu C:

Gdy JTAG jest włączone, podciąganie na PC5(TDI), PC3(TMS) i PC2(TCK) będzie aktywne nawet w trakcie Reset.

Końcówki Portu C	Funkcje dodatkowe
PC7	TOSC2 (oscylator licznika TC2 – końcówka 2)
PC6	TOSC1 (oscylator licznika TC2 – końcówka 1)
PC5	TDI (JTAG wejście Test Data)
PC4	TDO (JTAG wyjście Test Data)
PC3	TMS (JTAG tryb test)
PC2	TCK (JTAG Test Clock)
PC1	SDA (linia danych I^2C)
PC0	SCL (linia zegarowa I^2C)

- **TOSC2 – Port C, bit 7**

Gdy bit AS2 w rejestrze ASSR jest ustawiony, licznik TC2 pracuje asynchronicznie, końcówka 7 PC7 jest odłączona od portu i doprowadzona do wyjścia wzmacniacza w oscylatorze licznika TC2. W tym trybie kwarc jest dołączony do tej końcówki i nie może ona być używana jako normalna końcówka I/O.

- **TOSC1 – Port C, bit 6**

Gdy bit AS2 w rejestrze ASSR jest ustawiony, licznik TC2 pracuje asynchronicznie, końcówka 6 PC6 jest odłączona od portu i doprowadzona do wejścia wzmacniacza w oscylatorze licznika TC2. W tym trybie kwarc jest dołączony do tej końcówki i nie może ona być używana jako normalna końcówka I/O.

- **TDI – Port C, bit 5**

Szeregowe wejście danych JTAG. Gdy JTAG jest aktywny, końcówka ta nie może być używana jako normalna końcówka I/O.

- **TDO – Port C,bit 4**

Szeregowe wyjście danych JTAG. Gdy JTAG jest aktywny, końcówka ta nie może być używana jako normalna końcówka I/O.
Gdy TAP przesyła dane TDO jest końcówką trójstanową.

- **TMS – Port C, bit 3**

Wybór trybu Test. Końcówka ta jest używana do kierowania przez kontroler TAP. Gdy JTAG jest aktywny, końcówka ta nie może być używana jako normalna końcówka I/O.

- **TCK – Port C, bit 2**

Operacje JTAG odbywają się synchronicznie z zegarem TCK. Gdy JTAG jest aktywny, końcówka ta nie może być używana jako normalna końcówka I/O.

- **SDA – Port C, bit 1**

Linia danych interfejsu TWI. Ustawienie bitu TWEN w rejestrze TWCR powoduje włączenie interfejsu TWI, końcówka ta zostaje odłączona od portu i staje się wejściem / wyjściem danych TWI. W tym trybie do końcówki dołączony jest filtr usuwający impulsy zakłócające krótsze niż 50ns, a końcówka pracuje jako otwarty dren. Rezystor podciągający pull-up może być jednak sterowany przez bit w rejestrze PORTC1.

- **SCL – Port C, bit 0**

Linia zegarowa interfejsu TWI. Ustawienie bitu TWEN w rejestrze TWCR powoduje włączenie interfejsu TWI, końcówka ta zostaje odłączona od portu i staje się sygnałem zegarowym TWI. W tym trybie do końcówki dołączony jest filtr usuwający impulsy zakłócające krótsze niż 50ns, a końcówka pracuje jako otwarty dren. Rezystor podciągający pull-up może być jednak sterowany przez bit w rejestrze PORTC0.

Funkcje alternatywne portu D:

Końcówki Portu D	Funkcje dodatkowe
PD7	OC2 (wyjście układu porównania licznika TC2)
PD6	ICP (wejście przechwytywania licznika TC1)
PD5	OC1A (wyjście układu porównania A licznika TC1)
PD4	OC1B (wyjście układu porównania B licznika TC1)
PD3	INT1 (wejście zewnętrznego przerwania INT1)
PD2	INT0 (wejście zewnętrznego przerwania INT0)
PD1	TXD (wyjście USART)
PD0	RXD (wejście USART)

- **OC2 – Port D, bit 7**

wyjście układu porównania licznika TC2. końcówka może pracować jako zewnętrzne wyjście układu porównania (przy ustawieniu kierunku portu DDRD.7 w stan „1"). OC2 może również służyć za wyjście PWM.

- **ICP – Port D, bit 6**

ICP – wejście przechwytywania w liczniku TC1.

- **OC1A – Port D, bit 5**

wyjście układu porównania A licznika TC1. końcówka może pracować jako zewnętrzne wyjście układu porównania (przy ustawieniu kierunku portu DDRD.5 w stan „1"). OC1A może również służyć za wyjście PWM.

- **OC1B – Port D, bit 4**

wyjście układu porównania B licznika TC1. końcówka może pracować jako zewnętrzne wyjście układu porównania (przy ustawieniu kierunku portu DDRD.4 w stan „1"). OC1B może również służyć za wyjście PWM.

- **INT1 – Port D, bit 3**

Końcówka PD3 może służyć jako źródło przerwania zewnętrznego.

- **INT0 – Port D, bit 2**

Końcówka PD2 może służyć jako źródło przerwania zewnętrznego.

- **TXD – Port D,bit 1**

Gdy USART jest włączony, końcówka ta jest wyjściem niezależnie od ustawień rejestru kierunku portu.

- **RXD – Port D,bit 0**

Gdy USART jest włączony, końcówka ta jest wejściem niezależnie od ustawień rejestru kierunku portu, podciąganie pull-up zależy od wartości bitu w rejestrze PORTD.0

Rejestry portów:

Rejestr portu A:

Bit	7	6	5	4	3	2	1	0	
	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	PORTA
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr kierunku portu A:

Bit	7	6	5	4	3	2	1	0	
	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRA
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Adres końcówek wejściowych PINA

Bit	7	6	5	4	3	2	1	0	
	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	PINA
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr portu B:

Bit	7	6	5	4	3	2	1	0	
	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr kierunku portu B:

Bit	7	6	5	4	3	2	1	0	
	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Adres końcówek wejściowych PINB

Bit	7	6	5	4	3	2	1	0	
	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr portu C:

Bit	7	6	5	4	3	2	1	0	
	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	PORTC
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr kierunku portu C:

Bit	7	6	5	4	3	2	1	0	
	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Adres końcówek wejściowych PINC

Bit	7	6	5	4	3	2	1	0	
	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	PINC
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr portu D:

Bit	7	6	5	4	3	2	1	0	
	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr kierunku portu D:

Bit	7	6	5	4	3	2	1	0	
	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Adres końcówek wejściowych PIND:

Bit	7	6	5	4	3	2	1	0	
	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Zewnętrzne przerwania:

Zewnętrzne przerwania są wyzwalane przez końcówki INT0, INT1 i INT2. Przerwania zostają wyzwolone nawet jeżeli końcówki INT0...2 pracują jako wyjścia. Pozwala to na programowe wygenerowanie przerwania. Wyzwalać można zboczem narastającym, opadającym lub stanem (poziomem) niskim na końcówce (INT2 – tylko zboczem). Ustawiane jest to przez rejestry MCUCR i MCUCSR. Jeżeli zewnętrzne przerwanie jest aktywne i ustawione na wyzwalanie poziomem (tylko INT0 i INT1), przerwanie będzie wyzwalane tak długo jak końcówka ta będzie w stanie niskim. Reakcja na zbocze jest możliwa tylko wtedy gdy pracuje zegar systemowy $clk_{I/O}$. Poziom niski na INT0 i INT1 i zbocze na INT2 są wykrywane asynchronicznie. Wybudzanie procesora zewnętrznym przerwaniem można przeprowadzić w trybie Idle, ponieważ zegar $clk_{I/O}$ pracuje (w innych trybach nie). Należy zwrócić uwagę, że gdy do wybudzania procesora ze stanu Power-down posłuży zewnętrzne przerwanie wyzwalane poziomem, poziom ten musi utrzymywać się na końcówce przez pewien czas potrzebny do uruchomienia się MCU. Zmiana poziomu jest próbkowana dwukrotnie przez zegar Watchdog. Okres oscylatora watchdog wynosi 1us przy 5V i 25°C. Częstotliwość Watchdog jest w pewnym stopniu zależna od napięcia zasilającego. MCU zostanie uruchomiony gdy wejście będzie posiadało wymagany poziom podczas próbkowania (niski) lub poziom ten będzie utrzymywać się aż do zakończenia procesu uruchamiania MCU. Czas uruchamiania jest ustalany przez fuse-bit SUT (opis wcześniej). Jeżeli poziom jest dwukrotnie próbkowany przez Watchdog ale zniknie zanim MCU się wybudzi, układ nadal zostanie w trybie uśpienia.

Rejestr kontroli MCU – MCUCR:

Bit	7	6	5	4	3	2	1	0	
	SE	SM2	SM1	SM0	ISC11	ISC10	ISC01	ISC00	MCUCR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bity 3,2 – ISC11, ISC10 – bity kontroli przerwań.**

Przerwanie zewnętrzne nr 1 będzie uaktywnione przez końcówkę INT1 gdy bit I w rejestrze SREG i rejestr GICR (maski) będą odpowiednio ustawione. Poziom i zbocza uaktywniające przerwanie są przedstawione w tabeli poniżej. Wartość na końcówce INT1 jest próbkowana przed detekcją zbocza. Jeżeli wybrane do wyzwalania zostały stan niski lub zmiana stanu na przeciwny, przerwanie będzie wygenerowane, gdy ten stan na końcówce utrzyma się ponad jeden cykl zegara. Krótsze impulsy nie gwarantują poprawnej pracy. Gdy wybrana jest reakcja na stan niski, stan ten musi się utrzymać dopóki wykonywany aktualnie rozkaz nie zostanie zakończony (dopiero wtedy nastąpi skok do procedury przerwania).

ISC11	ISC10	opis
0	0	Poziom niski na INT1 generuje przerwanie
0	1	Zmiana stanu logicznego na INT1 generuje przerwanie
1	0	Opadające zbocze na INT1 generuje przerwanie
1	1	Narastające zbocze na INT1 generuje przerwanie

- **bity 1,0 – ISC01, ISC00 – bity kontroli przerwań.**

Przerwanie zewnętrzne nr 0 będzie uaktywnione przez końcówkę INT0 gdy bit I w rejestrze SREG i rejestr GICR (maski) będą odpowiednio ustawione. Poziom i zbocza uaktywniające przerwanie są przedstawione w tabeli poniżej. Wartość na końcówce INT0 jest próbkowana przed detekcją zbocza. Jeżeli wybrane do wyzwalania zostały stan niski lub zmiana stanu na przeciwny, przerwanie będzie wygenerowane, gdy ten stan na końcówce utrzyma się ponad jeden cykl zegara. Krótsze impulsy nie gwarantują poprawnej pracy. Gdy wybrana jest reakcja na stan niski, stan ten musi się utrzymać dopóki wykonywany aktualnie rozkaz nie zostanie zakończony (dopiero wtedy nastąpi skok do procedury przerwania).

ISC01	ISC00	opis
0	0	Poziom niski na INT0 generuje przerwanie
0	1	Zmiana stanu logicznego na INT0 generuje przerwanie
1	0	Opadające zbocze na INT0 generuje przerwanie
1	1	Narastające zbocze na INT0 generuje przerwanie

Rejestr kontroli i stanu – MCUCSR

Bit	7	6	5	4	3	2	1	0	
	JTD	ISC2	-	JTRF	WDRF	BORF	EXTRF	PORF	MCUCSR
Odczyt/zapis	Z/O	Z/O	O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	W opisie bitów					

- **bit 6 – ISC2 – przerwanie 2**

Przerwanie zewnętrzne nr 0 będzie uaktywnione przez końcówkę INT0 gdy bit I w rejestrze SREG i rejestr GICR (maski) będą odpowiednio ustawione. Gdy do ISC2 wpisane jest zero, przerwanie zostanie uruchomione opadającym zboczem na INT2, narastającym gdy wpisze się jedynkę. Zbocze na INT2 jest rejestrowane asynchronicznie. Impuls na INT2 szerszy niż minimalna szerokość impulsu opisana w tabeli, wygeneruje przerwanie. Impuls krótszy nie gwarantuje wyzwolenia przerwania. Zaleca się najpierw wyłączyć INT2 przez wykasowanie bitu INT2 w rejestrze GICR, wtedy można ustawić bit ISC2. Ostatecznie flaga przerwania INTF2 powinna być wyzerowana przez wpisanie do niej jedynki (rejestr flag GIFR) zanim przerwanie będzie ponownie uaktywnione.

symbol	parametr	min	typ	max	jednostka
t_{INT}	Minimalna szerokość impulsu		50		ns

Rejestr sterowania przerwań – GICR

Bit	7	6	5	4	3	2	1	0	
	INT1	INT0	INT2	-	-	-	IVSEL	IVCE	GICR
Odczyt/zapis	Z/O	Z/O	Z/O	O	O	O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – INT1: uaktywnienie zewnętrznego przerwania nr 1**

końcówka przerwania INT1 jest aktywna gdy bit INT1 i bit I w rejestrze SREG są ustawione. Aktywność na końcówce będzie powodować uruchomienie przerwania nawet jeśli końcówka pracuje jako wyjście.

- **bit 6 – INT0: uaktywnienie zewnętrznego przerwania nr 0**

końcówka przerwania INT0 jest aktywna gdy bit INT0 i bit I w rejestrze SREG są ustawione. Aktywność na końcówce będzie powodować uruchomienie przerwania nawet jeśli końcówka pracuje jako wyjście.

- **bit 5 – INT2: uaktywnienie zewnętrznego przerwania nr 2**

końcówka przerwania INT2 jest aktywna gdy bit INT2 i bit I w rejestrze SREG są ustawione. Aktywność na końcówce będzie powodować uruchomienie przerwania nawet jeśli końcówka pracuje jako wyjście.

Rejestr flag – GIFR:

Bit	7	6	5	4	3	2	1	0	
	INTF1	INTF0	INTF2	-	-	-	-	-	GIFR
Odczyt/zapis	Z/O	Z/O	Z/O	O	O	O	O	O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – INTF1: flaga przerwania zewnętrznego nr 1**

gdy zbocze lub zmiana poziomu logicznego na przeciwny na końcówce INT1 wyzwoli przerwanie, flaga INTF1 przyjmie stan wysoki, gdy bit I w SREG i bit INT1 w GICR będą ustawione, MCU skoczy do wektora przerwania a stamtąd do podprogramu obsługi przerwania. Flaga zostanie skasowana podczas gdy przerwania jest wykonywane. Może być też skasowana przez wpisanie do niej jedynki. Flaga jest zawsze wyzerowywana gdy reakcja INT1 jest ustawiona na poziom niski.

- **bit 6 – INTF0: flaga przerwania zewnętrznego nr 0**

gdy zbocze lub zmiana poziomu logicznego na przeciwny na końcówce INT0 wyzwoli przerwanie, flaga INTF0 przyjmie stan wysoki, gdy bit I w SREG i bit INT0 w GICR będą ustawione, MCU skoczy do wektora przerwania a stamtąd do podprogramu obsługi przerwania. Flaga zostanie skasowana podczas gdy przerwania jest wykonywane. Może być też skasowana przez wpisanie do niej jedynki. Flaga jest zawsze wyzerowywana gdy reakcja INT0 jest ustawiona na poziom niski.

- **bit 5 – INTF2: flaga przerwania zewnętrznego nr 2**

Pojawienie się zbocza na INT2 wyzwalającego przerwanie ustawi flagę INTF2. Gdy bit I w SREG i bit INT2 w GICR będą ustawione, MCU skoczy do wektora przerwania a stamtąd do podprogramu obsługi przerwania. Flaga zostanie skasowana podczas gdy przerwania jest wykonywane. Może być też skasowana przez wpisanie do niej jedynki. Gdy MCU zostanie wprowadzony w tryb sleep a przerwanie INT2 będzie wyłączone wejściowy bufor na tej końcówce będzie również wyłączony.

8-bit Timer/counter0 z PWM

Licznik TC0 – 8-bitowy licznik. Główne właściwości:

- pojedynczy licznik
- zerowanie przy porównaniu (auto-ładowanie)
- PWM z poprawną fazą
- Generator przebiegu prostokątnego
- 10-bitowy preskaler
- źródła przerwań od porównania i przepełnienia (OCF0 i TOV0)

Rejestry licznika TCNT0 i komparatora (wyjściowego) są 8-bitowe. Znaczniki przerwań są ustawiane w rejestrze TIRF. Wszystkie przerwania są niezależnie ustawiane w rejestrze TIMSK. Rejestry te są używane w pozostałych licznikach.
Licznik TC0 może być taktowany wewnętrznie, przez preskaler, lub źródło zegara wewnętrznego dołączonego do wejścia T0. Jest wyłączony, gdy nie jest wybrane żadne źródło.
Podwójnie buforowane wyjście rejestru komparatora (OCR0) jest stale porównywane z zawartością licznika. Wynik porównania może być użyte przez generator przebiegu do wytwarzania PWM lub sygnału o zmienianej częstotliwości na wyjściu porównania (OC0). Wynik porównania będzie również ustawiać Bit Porównania (OCF0) z którym może być użyty do wytwarzania przerwania (od porównania).
(określenia w oryginale:

BOTTOM – gdy licznik osiągnie 0x00
MAX – gdy licznik osiągnie 0xFF
TOP – gdy licznik osiągnie MAX (0xFF) lub wartość wpisaną do OCR0.)

Źródła zegara licznika TC0:
Licznik może być taktowany zewnętrznym lub wewnętrznym sygnałem zegarowym. Źródło sygnału jest wybierane przez bity CS02:0 w rejestrze TCCR0
Jednostka zliczająca:
Główna część licznika jest 8-bitową programowalną dwukierunkową jednostką.
W zależności od trybu pracy licznik jest zwiększany, zmniejszany lub zerowany w każdym cyklu zegara (clk_{T0}). Bit przepełnienia (TOV0) może być użyty do generowania przerwań.
Wyjściowy rejestr porównujący:
8-bitowy komparator ciągle porównuje zawartość licznika (TCNT0) z rejestrem OCR0. Osiągnięcie przez TCNT0 takiej samej wartości jak OCR0 jest przez komparator sygnalizowane. Spowoduje ono ustawienie bitu OCF0 w następnym cyklu zegara. Jeżeli bity OCIE0 i globalny system przerwań I w SREG są ustawione to jedynka w OCF wygeneruje przerwanie. OCF0 jest automatycznie kasowany w trakcie wykonywania przerwania. Może być również kasowany programowo przez wpisanie zera. Porównanie wykorzystywane jest do wytwarzania przebiegu prostokątnego, przy odpowiednim ustawieniu bitów WGM01:0 i COM01:0. Rejestr OCR0 jest podwójnie buforowany gdy licznik pracuje w trybie PWM. Podczas normalnej pracy i auto-ładowania (CTC) podwójne buforowanie jest wyłączone. Buforowanie to synchronizuje wpis do rejestru OCR0 i zapobiega w związku z tym pojawieniu się niesymetrycznych sygnałów PWM.
Gdy podwójne buforowanie jest aktywne, CPU ma dostęp do OCR0 przez bufor, gdy jest wyłączone dostęp do tego rejestru jest bezpośredni.
FOC:
W trybach innych niż PWM wyjście porównania komparatora może być wymuszone przez wpis jedynki do bitu FOC0. Wymuszenie to nie ustawi bitu OCF0 ani nie spowoduje zerowania / auto-ładowania licznika, ale końcówka OC0 będzie uaktualniona gdy porównanie rzeczywiście wystąpi. (o ile COM01:0 zostaną odpowiednio skonfigurowane).
Wszystkie operacje zapisu przez CPU do rejestru TCNT0 spowodują zablokowanie porównania w następnym cyklu zegarowym, nawet gdy licznik jest zatrzymany. To pozwala na zainicjowanie tej samej wartości w OCR0 jak TCNT0, bez wyzwalania przerwań gdy licznik jest włączony. Blokowanie to może jednak spowodować pewne komplikacje gdy, podczas użycia wyjścia porównania, zmiana w TCNT0 niezależnie od tego czy licznik pracuje, czy nie. Jeśli wartość wpisana do TCNT0 jest taka sama jak w OCR0, komparator nie zadziała, w skutek czego generowanie przebiegu prostokątnego będzie nieprawidłowe. Podobnie przy wpisaniu wartości różnej od zera, gdy licznik odlicza w dół.
Ustawienie OC0 powinno być spełnione zanim port będzie ustawiony jako wyjście przez rejestr DDR (rejestr kierunku portu). Najprostszym sposobem ustawienia wartości końcówki OC0 jest użycie bitu FOC0 w trybie normalnej pracy licznika. Rejestr OC0 przechowuje tą wartość nawet gdy jest zmieniana przez tryb generowania przebiegu.

Bity COM01:0 nie są podwójnie buforowane razem z wartością komparatora Zmiana COM01:0 wywoła natychmiastowy efekt.

Wyjściowa jednostka porównująca:

Bity wyjścia porównania (COM01:0) posiadają dwie funkcje. Generator przebiegu prostokątnego używa tych bitów do zdefiniowania stanu wyjścia porównania (OC0) przy następnym porównaniu. Stan OC0 dotyczy wewnętrznego rejestru, a nie końcówki.

Główne funkcje portu we/wy są ignorowane przez OC0, jeżeli któryś z bitów COM01:0 są ustawione. Jednak kierunek końcówki OC0 jest kontrolowane przez rejestr kierunku portu DDR. Kierunek portu dla końcówki OC0 (DDR_OC0) musi być ustawiona jako wyjście, zanim końcówka ta będzie wyjściem OC0.

Zmiana stanu bitów COM01:0 da efekt przy pierwszym porównaniu zaraz po wpisie do tych bitów. Dla trybów innych niż PWM ta sytuacja wymusi natychmiastową reakcję przez użycie bitu zbocza FOC0.

Tryby pracy:

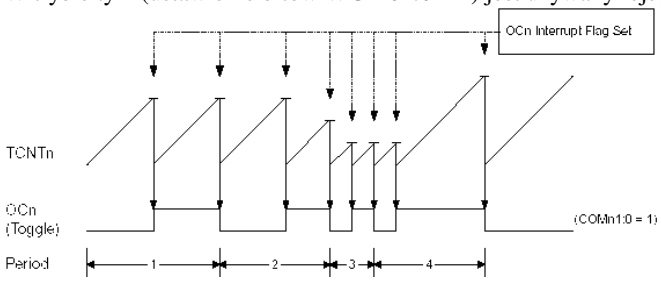
Poszczególne tryby pracy licznika (np. generowanie przebiegu; PWM) uzyskiwane są przez ustawienie bitów Generatora Przebiegu (WGM01:0) i Wyjścia Porównania (COM01:0). Dla trybów innych niż PWM bity COM01:0 kontrolują albo czy wyjście jest ustawione, wyzerowane, albo zmienione podczas porównania.

Tryb normalny: Jest to najprostszy tryb pracy (bity WGM01:0 = 0). W tym trybie licznik zawsze liczy w górę i jest zerowany po przepełnieniu (gdy zliczy do 0xFF). Bit przepełnienia jest ustawiany w tym samym cyklu zegara, gdy licznik zeruje się. Może być traktowany jako dziewiąty bit licznika, z tym że jest tylko ustawiany, nie kasowany. Jednak fakt automatycznego zerowania tego bitu w skutek przyjęcia przerwania może posłużyć do programowego zwiększenia rozdzielczości licznika. Nowa wartość licznika może być bez przeszkód wpisywana w dowolnej chwili.

Wyjście porównania może być użyte do wyzwalania przerwania w ustalonym czasie. Użycie wyjścia porównania do generowania przebiegu w trybie Normal nie jest zalecane, ponieważ zajmuje to za dużo czasu procesorowi.

Zerowanie licznika w trybie porównania (CTC):

W trybie tym (ustawienie bitów WGM01:0 = 2) jest używany rejestr OCR0.



W tym trybie licznik jest zerowany (wartość 0x00), gdy zliczy do takiej wartości jaka jest wpisana do rejestru OCR0. Rejestr ten definiuje największą wartość do której może doliczyć licznik i w ten sposób decyduje o jego rozdzielczości. Ten tryb jest najlepszą metodą na określenie częstotliwości wyjścia. Przerwanie może być generowane podczas przepełniania się licznika przez użycie bitu OCF0. Przerwanie może zmienić wartość szczytową dla licznika, jakkolwiek zmiana wartości szczytowej zbliżonej do zera, podczas gdy licznik pracuje z wyłączonym przeskalerem, lub ustawionym na niewielki podział, muszą być dokonywane ostrożnie, ponieważ w trybie CTC nie działa podwójne buforowanie. Jeżeli wpisana nowa wartość do OCR0 jest mniejsza od aktualnej wartości w TCNT0, licznik opóźni proces porównania. Licznik będzie musiał zliczyć do wartości maksymalnej (0xFF) i po wyzerowaniu zliczyć do wartości przy której wystąpi porównanie.

Generowanie przebiegu prostokątnego w trybie CTC na wyjściu OC0 uzyskuje się przez ustawienie bitów trybu Wyjścia Porównania (COM01:0 = 1). Wartość OC0 pojawi się na końcówce portu we/wy gdy w rejestrze kierunku portu końcówkę tą ustawi się jako wyjście.

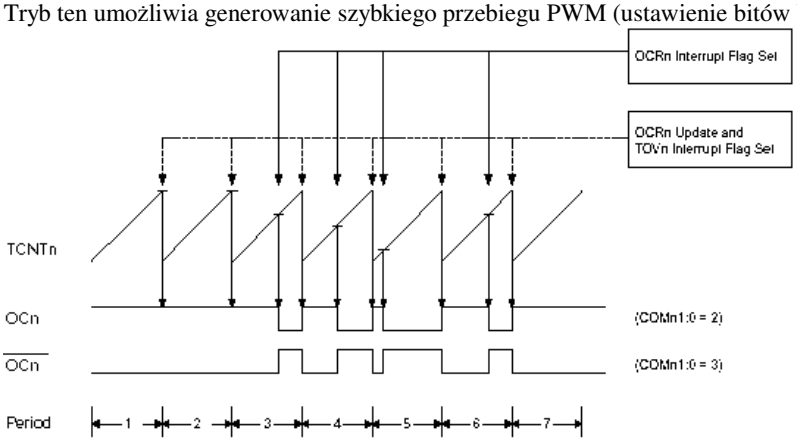
Generowany przebieg osiągnie najwyższą częstotliwość równą $f_{OC0}=f_{CLK\_I/O}/2$, gdy do OCR0 wpisane jest zero. Częstotliwość przebiegu wyrażona jest wzorem:

$$f_{OCn} = \frac{f_{CLK\_I/O}}{2 \cdot N \cdot (1 + OCRn)}$$

gdzie N jest podziałem preskalera (przyjmuje wartości 1, 8, 64, 256 lub 1024).

Tak jak w trybie Normal, bit przepełnienia TOV0 jest ustawiany w tym samym cyklu zegara w którym następuje wyzerowanie licznika w skutek przepełnienia.

Tryb Fast PWM:
Tryb ten umożliwia generowanie szybkiego przebiegu PWM (ustawienie bitów WGM01:0 = 3).



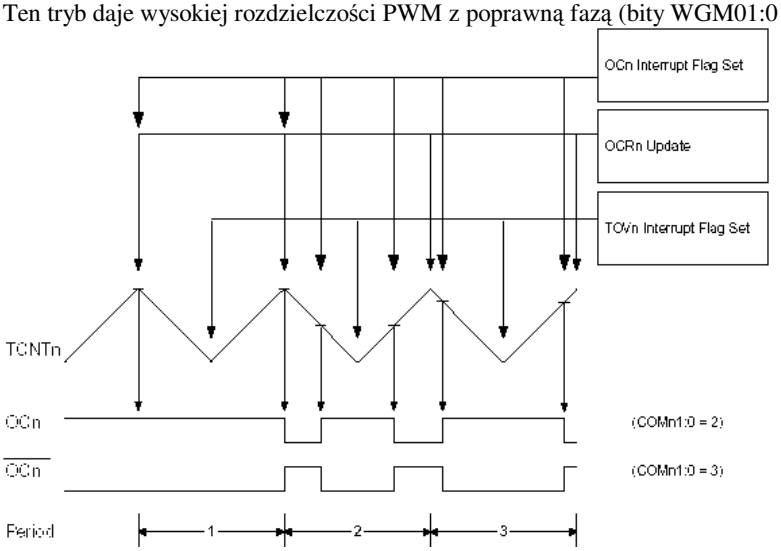
Tryb ten różni się od pozostałych operacją pojedynczego zliczania. Licznik zlicza od zera do maksymalnej (0xFF) i ponownie zaczyna od zera. W trybie bez odwrócenia wyjścia, wyjście porównania OC0 jest zerowane gdy wartość licznika jest większa lub równa wartości w komparatorze i ustawiane gdy licznik jest zerowany. W trybie z odwróconym wyjściem OC0 jest ustawiane podczas porównania i kasowane przy zerowaniu. Częstotliwość trybu Fast PWM może być dwa razy wyższa od trybu Correct Pwm używającego operacji podwójnego zliczania. Tryb ten jest odpowiedni do regulacji mocy, przetwarzania cyfrowo-analogowego. Wysoka częstotliwość PWM umożliwia zmniejszenie rozmiarów zewnętrznych elementów współpracujących – cewek czy kondensatorów i obniża koszty wykonania.
Licznik jest zwiększany aż osiągnie maksymalną wartość (0xFF), wtedy, w następnym cyklu zegara następuje jego wyzerowanie.
Bit przepełnienia (TOV0) jest ustawiany podczas osiągnięcia przez licznik wartości maksymalnej. Jeżeli przerwania są aktywne, to procedura przerwania może zmienić wartość rejestru porównania.
Przebieg generowany jest na końcówce OC0 gdy jest ona ustawiona jako wyjście w rejestrze kierunku portu (DDR). Ustawienie bitów COM01:0 jako 2 (COM00 = 0; COM01=1) uaktywnia nie odwrócony przebieg PWM na końcówce OC0, wpisanie do COM01:0 wartości 3 (11) powoduje odwrócenie przebiegu.
Częstotliwość nośna obliczana jest ze wzoru:

$$f_{OCn} = \frac{f_{CLK\_I/O}}{N \cdot 256}$$

gdzie N jest podziałem preskalera (przyjmuje wartości 1, 8, 64, 256 lub 1024).

Jeżeli do rejestru OCR0 jest wpisane zero, na wyjściu pojawiać się będzie krótka szpilka przy takcie zegara w którym licznik się przepełnia. Ustawienie OCR0 na maksimum (0xFF) ustawi wyjście w stanie niskim lub wysokim (zależnie od bitów COM01:0).
Przebieg z 50% wypełnieniem przy częstotliwości $f_{OC0}=f_{CLK\_I/O}/2$ uzyska się wpisując do OCR0 liczbę 0x00 (COM01:0 = 1). Różnicą do trybu CTC jest aktywne podwójne buforowanie).

PWM z poprawną fazą:
Ten tryb daje wysokiej rozdzielczości PWM z poprawną fazą (bity WGM01:0 ustawione na 1).



Zasada działania opiera się na podwójnym zliczaniu. Licznik powtarza zliczanie od wartości zerowej do maksymalnej i na odwrót. W trybie bez odwracania wyjść OC0 jest zerowane przy porównaniu pomiędzy TCNT0 a OCR0 podczas zliczania w górę i ustawiany podczas porównania przy zliczaniu w dół. W trybie z zanegowanym wyjściem sytuacja jest odwrotna. System podwójnego zliczania daje mniejszą częstotliwość niż we wcześniejszym trybie. Dzięki uzyskanej w ten sposób symetrii tryb ten jest zalecany do sterowania silnikami.
Rozdzielczość PWM z poprawną fazą jest stała i wynosi 8 bitów. Licznik jest zwiększany dopóki nie osiągnie maksimum – wtedy zmienia kierunek liczenia. Rejestr TCNT0 będzie miał wartość równą maksimum (0xFF) przez jeden cykl zegara taktującego licznik.
Bit przepełnienia (TOV0) jest ustawiany przy dojściu licznika do wartości maksymalnej.
Komparator powoduje generowanie przebiegu na końcówce OC0. Ustawienie bitów COM01:0 na 2 uruchomi nie odwrócony PWM. Tryb z zanegowanym wyjściem dostępny jest Przy COM01:0 ustawionych na 3. Bieżący stan OC0 będzie odzwierciedlony na końcówce OC0 gdy będzie ona ustawiona jako wyjście w rejestrze kierunku portu. Przebieg PWM jest uzyskiwany przez wyzerowanie (lub ustawienie) OC0 podczas porównania OCR0 i TCNT0 gdy licznik zlicza w górę i ustawiana (lub zerowana) przy porównaniu OCR0 i TCNT0 podczas pracy licznika w dół. Częstotliwość licznika określona jest zależnością:

$$f_{OCnPCPPWM} = \frac{f_{clk\_I/O}}{N \cdot 510}$$

gdzie N jest podziałem preskalera (przyjmuje wartości 1, 8, 64, 256 lub 1024).
Jeśli do rejestru OCR0 wpisana jest wartość minimalna (0x00), wyjście będzie ciągle w stanie niskim; jeśli będzie to wartość maksymalna (0xFF) to wyjście będzie w stanie wysokim (dla trybu bez odwracania wyjść – z inwersją odwrotnie).

Wyjście OC0n może zmienić się ze stanu wysokiego na niski chociaż nie zaszło porównanie. Miejsca tych zmian gwarantują symetrię. Istnieją dwa przypadki zmiany stanu bez zajścia porównania:

- zmienia się wartość OCR0A z maksymalnej; jeżeli wartoścsia OCR0A jest 0xFF wartość OC0 jest taka sama jak efekt porównania przy liczniku doliczającym w dół.
- licznik zaczął zliczać od wartości wyższej niż w OCR0A i opóźnia się porównanie.

Opis rejestrów 8-bitowego licznika
Rejestr sterujący licznikiem TCCR0:

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Odczyt/zapis	W_{zapis}	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – FOC0: Wymuszenie wyjścia porównania**

bit FOC0 jest aktywny tylko wtedy, gdy WGM00 ustawiają licznik w trybie innym niż PWM. Jakkolwiek ze względu na zgodność z przyszłymi układami bit ten musi być zerowany gdy rejestrem TCCR0 licznik jest skonfigurowany jako PWM. Wpisanie logicznej jedynki do FOC0 wymusza natychmiastowe porównanie

- **bity 6,3 – WGM01:0 – rodzaje generowanych przebiegów:**

bity te kontrolują sposób zliczania licznika: wartość maksymalną do której dąży licznik i rodzaj wytwarzanego przebiegu. Tryby pracy licznika: zwykły, zerowanie licznika podczas porównania i dwa rodzaje PWM (modulacji szerokości impulsu).

tryb	WGM01 (CTC0)	WGM00 (PWM0)	Rodzaj pracy licznika	Górna wartość licznika	Uaktualnienie OCR0	Ustawienie bitu TOV0
0	0	0	Zwykły licznik	0xFF	Natychmiast	MAX
1	0	1	PWM z poprawną fazą	0xFF	Wart. Szczyt.	BOTTOM
2	1	0	CTC	W rejestrze OCR0	Natychmiast	MAX
3	1	1	Szybki PWM	0xFF	Wart. Szczyt.	MAX

- **bity 5,6 – COM01:0 – rodzaj wyjścia trybu porównania**

określają zachowanie się wyjścia komparatora licznika (końcówka OC0). Jeśli jeden z bitów lub oba bity są ustawione to wyjście układu porównującego w liczniku zostaje dołączone do końcówki portu I/O. Rejestr kierunku portu (DDR) musi ustawić port jako wyjście. Gdy OC0 jest dołączone do końcówek portu, funkcje bitów COM01:0 zależą od ustawienia bitów WGM01:0

Tryb zwykły i porównania

COM01	COM00	Określenie
0	0	Zwykły tryb pracy portu, OC0 odłączone
0	1	Porównanie zmienia OC0
1	0	Porównanie zeruje OC0
1	1	Porównanie ustawia OC0

Tryb Fast PWM i porównania (1)

COM01	COM00	Określenie
0	0	Zwykły tryb pracy portu, OC0 odłączone
0	1	Zarezerwowane
1	0	Porównanie zeruje OC0, ustawia przy wartości szczytowej
1	1	Porównanie ustawia OC0, zeruje przy wartości szczytowej

Tryb PWM z poprawną fazą i porównania (1)

COM01	COM00	Określenie
0	0	Zwykły tryb pracy portu, OC0 odłączone
0	1	Zarezerwowane
1	0	Porównanie zeruje OC0 gdy licznik liczy w górę, ustawia gdy licznik liczy w dół
1	1	Porównanie ustawia OC0 gdy licznik liczy w górę, zeruje gdy licznik liczy w dół

(1) porównanie jest ignorowane, ale zerowanie lub ustawianie działa, gdy OCR0 jest takie samo jak wartość szczytowa licznika i COM01 jest w stanie 1.

- **bity 2:0 – CS02:0 – wybór sygnału zegarowego:**

te trzy bity decydują o źródle sygnału zegarowego dla licznika TC:

CS02	CS01	CS00	Określenie
0	0	0	Brak sygnału zegarowego – licznik stoi
0	0	1	$clk_{I/0}$ (pełna prędkość – bez preskalera)
0	1	0	$clk_{I/0}$ / 8 (z preskalerem)
0	1	1	$clk_{I/0}$ / 64 (z preskalerem)
1	0	0	$clk_{I/0}$ / 256 (z preskalerem)
1	0	1	$clk_{I/0}$ / 1024 (z preskalerem)
1	1	0	Zewnętrzny sygnał zegarowy na końcówce T0 – takt przy zboczu opadającym
1	1	1	Zewnętrzny sygnał zegarowy na końcówce T0 – takt przy zboczu narastającym

Jeżeli ustawione jest zewnętrzne źródło zegara dla licznika TC0, licznik będzie mógł być napędzany nawet jeśli port ustawiony jest jako wyjście. Umożliwia to programowe taktowanie licznika (przez zmianę stanu końcówki portu).

Rejestr licznika TCNT0:

Bit	7	6	5	4	3	2	1	0	
	TCNT0[7:0]								TCNT0
Odczyt/zapis	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr TCNT0 daje bezpośredni dostęp, poprzez operacje zapisu i odczytu, do stanu licznika (licznik jest 8-bitowy). Modyfikacja tego rejestru w trakcie pracy licznika grozi pominięciem chwili porównania między TCNT0 a OCR0. Wpis do TCNT0 zmienia układ porównujący na następny cykl zegara.

Rejestr wyjściowy układu porównującego OCR0:

Bit	7	6	5	4	3	2	1	0	
	OCR0[7:0]								OCR0
Odczyt/zapis	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Wartość początkowa	0	0	0	0	0	0	0	0	

Zawiera 8-bitów wartości stale porównywanej z wartością licznika (TCNT0). Porównanie może być użyte do generowania przerwania, lub przebiegu na końcówce OC0.

Rejestr maskowania przerwań TIMSK:

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 1 – OCIE0: uaktywnienie przerwania od układu porównującego:**

Ustawienie bitu OCIE0 i I w SREG w stan 1 powoduje uaktywnienie przerwania pochodzącego od układu porównującego licznika. Odpowiednie przerwanie jest wykonywane, gdy zajdzie porównanie w liczniku TC0 np. gdy ustawiony jest bit OCF0 w rejestrze flag przerwań licznika – TIFR.

- **bit 0 – TOIE0: uaktywnienie przerwania przy przepełnieniu**

Przerwanie jest aktywne gdy bit TOIE0 i I w SREG są ustawione.

Rejestr flag przerwań licznika TC0 TIFR:

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 1 – OCF0: flaga przerwania przy porównaniu**

OCF0 ustawi się gdy wystąpi porównanie między licznikiem TC0 a OCR0. Jest zerowany sprzętowo gdy wykona się przerwanie. Przerwanie zostanie wykonane, gdy I w SREG, OCIE i OCF0 są ustawione.

- **bit 0 – TOV0: flaga przepełnienia licznika:**

Przepełnienie się licznika ustawia bit TOV0, kasuje się gdy wykonuje się odpowiednie przerwanie. TOV0 jest kasowane przez wpisanie logicznej jedynki. Stan wysoki w TOIE0, I w SREG i TOV0 uruchamia obsługę przerwania. W trybie PWM z poprawną fazą bit ustawia się gdy licznik zmienia kierunek liczenia przy wartości $00.

Preskalery liczników TC0 i TC1:

Liczniki TC0 i TC1 korzystają z tego samego preskalera częstotliwości, ale mogą mieć ustawiony różny podział.

Wewnętrzne źródło zegara:

Licznik TC może być taktowany bezpośrednio przez zegar systemowy (dla ustawienia CSn2:0 = 1). Pozwala to na maksymalną częstotliwość pracy równą częstotliwości zegara systemowego procesora ($f_{CLK\_I/O}$). Jeden z czterech sygnałów preskalera może być użyty jako źródło sygnału o częstotliwościach: $f_{CLK\_I/O}/8$, $f_{CLK\_I/O}/64$, $f_{CLK\_I/O}/256$, $f_{CLK\_I/O}/1024$.

Zerowanie preskalera:

Działanie preskalera częstotliwości jest niezależne od wybranego źródła zegara dla licznika TC0 i TC1. Jeśli preskaler nie jest wybrany jako źródło sygnału zegarowego dla liczników to jego stan w danej chwili jest nie określony, np. jeśli preskaler jest użyty to zwiększenie stanu licznika po jego włączeniu może trwać od 1 do N+1 taktów zegara systemowego, gdzie N – podział preskalera (8, 64, 256 lub 1024).

Jest możliwe wyzerowanie preskalera w celu zsynchronizowania licznika z wykonywanym programem, ale może to spowodować chwilowe skrócenie okresu dla drugiego licznika, jeśli jest podłączony do tego preskalera.

Zewnętrzne źródło zegara:

Zewnętrzny sygnał zegarowy jest dołączony do końcówek T1 i T0. Stan tych końcówek jest próbkowany w każdym cyklu zegara systemowego. Spróbkowany sygnał przechodzi przez czujnik impulsu. Czujnik ten odcina sygnały o wyższych częstotliwościach. Wprowadza dodatkowo opóźnienie o 2,5 do 3,5 cykli systemowych od pojawienia się impulsu zegarowego na końcówkach T0, T1 do wpłynięcia na stan licznika.

Włączanie lub wyłączanie wejść zewnętrznego zegara musi być wykonane gdy T0, T1 są stabilne.

Połowa okresu sygnału zegara zewnętrznego musi być dłuższa od jednego cyklu zegara systemowego w celu poprawnego spróbkowania. Częstotliwość zewnętrzna musi być mniejsza od połowy częstotliwości systemowej ($f_{ExtClk} < f_{clk\_I/O} / 2$) z 50% wypełnieniem.

Ze względu na tolerancję rezonatora kwarcowego lub wewnętrznego generatora maksymalna częstotliwość zewnętrznego zegara nie powinna przekraczać $f_{CLK\_I/O} / 2{,}5$.

Rejestr specjalny SFIOR:

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	-	ACME	PUD	PSR2	PSR10	SFIOR
Odczyt/zapis	Z/O	Z/O	Z/O	O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 0 – PSR10 – reset preskalera liczników TC1 i TC0:**

wyzerowanie preskalera uzyska się wpisując 1 do bitu PSR10. Bit wyzerowany zostanie sprzętowo po wykonaniu zerowania preskalera. Wpisanie zera programowo będzie nieskuteczne. Wyzerowanie preskalera wpłynie na obydwa liczniki, ponieważ oba wspólnie z niego korzystają. Gdy bit jest odczytywany wynikiem zawsze będzie zero.

16-bit licznik TC1

podstawowe cechy:

- 16 bitów
- dwa niezależne układy porównujące
- podwójnie buforowane rejestry wyjścia porównania
- układ przechwytywania
- wejście przechwytywania z redukcją zakłóceń
- tryb auto – ładowania
- modulator szerokości impulsów PWM
- zmienne wypełnienie PWM
- generator częstotliwości
- zliczanie impulsów zewnętrznych
- cztery niezależne źródła przerwań (TOV1, OCF1A, OCF1B, ICF1)

rejestry:

Rejestry licznika TCNT1, komparatora (OCR1A/B) i przechwytywania (ICR1) są 16-bitowe. Dostęp do tych rejestrów dobywa się w specyficzny sposób. Rejestry sterujące licznikiem TCCR1A/B są 8-bitowe i są dostępne w tradycyjny sposób. Stan przerwań jest widoczny w TFIR. Są one indywidualnie ustawiane w rejestrze TIMSK.
Licznik TC1 może być taktowany wewnętrznie, przez preskaler, lub źródło zegara wewnętrznego dołączonego do wejścia T1. sygnał zegarowy służy do zmniejszania / zwiększania zawartości licznika. Licznik jest wyłączony, gdy nie jest wybrane żadne źródło. Wyjście zegarowe z układu wybierającego źródło zegara jest określane jako clk_{T1}.
Podwójnie buforowane wyjście rejestru komparatora (OCR1A/B) jest stale porównywane z zawartością licznika. Wynik porównania może być użyte przez generator przebiegu do wytwarzania PWM lub sygnału o zmienianej częstotliwości na wyjściu porównania (OC1A/B). Wynik porównania będzie również ustawiać Bit Porównania (OCF1A/B) z którym może być użyty do wytwarzania przerwania (od porównania). Rejestr przechwytywania (ICR) służy do przechwycenia zawartości licznika poprzez wejście przechwytywania (końcówka ICP1) lub komparator analogowy. Wejście przechwytujące posiada filtr zakłóceń.

Zgodność:

16-bit licznik został uaktualniony w stosunku do poprzednich wersji 16-bit liczników AVR.

- wszystkie 16-bitowe rejestry licznika znajdują się w pod tym samym adresem I/O
- nie są zmienione pozycje i znaczenie bitów w rejestrach
- takie same wektory przerwań

zmianie uległy niektóre nazwy bitów:

- PWM10 jest zastąpione WGM10
- PWM11 jest zastąpione WGM11
- CTC1 jest zastąpione WGM12

Bity dodane do rejestrów licznika:

- FOC1A i FOC1B dodano do TCCR1A
- WGM13 dodano do TCCR1B

Dostęp do 16-bitowych rejestrów:

TCNT1, OCR1A/B i ICR1 są 16-bitowymi rejestrami które dostępne są przez 8-bit szynę danych. Dostęp odbywa się przez dwie bajtowe operacje zapisu lub odczytu. Wszystkie 16-bit rejestry posiadają 8-bitowy rejestr pomocniczy przechowujący starszą część 16-bit rejestru podczas operacji dostępu. Jeżeli młodsza część 16-bitowego rejestru jest zapisywana przez CPU, część starsza bajtu jest zapisana do rejestru pomocniczego jednocześnie. Gdy młodsza część jest odczytywana, starsza część jest kopiowana do rejestru pomocniczego w tym samym czasie.
Nie wszystkie rejestry 16-bit używają pomocniczego rejestru. Odczyt z OCR1A/B nie wymaga stosowania go.
Do zapisu 16-bitów część starsza musi być wpisana najpierw (potem młodsza). Przy odczycie, odczytać należy najpierw część młodszą.
Poniższy przykład pokazuje sposób dostępu do 16-bit rejestrów (zakładając, że przerwania są wyłączone). Przykład ten może być użyty również do rejestrów OCR1A/B i ICR1: (kompilator C operuje wartościami 16-bitowymi)

Przykład w asemblerze:

```
...
; wpisz 0x1FF do TCNT1
ldi r17 , 0x01
ldi r16 , 0xFF
out TCNT1H , r17
out TCNT1L , r16
; odczyt TCNT1 do r17:r16
in r16 , TCNT1L
in r17 , TCNT1H
...
```

Przykład w C:

```
Unsigned int i;
...
/* wpisz 0x1FF do TCNT1 */
TCNT1 = 0x1FF;
/* odczyt TCNT1 do i */
i = TCNT1;
...
```

Przykładowy program w asemblerze oddaje wartość TCNT1 w rejestrach r17,r16.

Dostęp do 16-bit rejestrów jest operacją nierozłączną. Jeśli jakieś przerwanie wystąpi między obydwoma instrukcjami dostępu (zapisu lub odczytu), i przerwania to zmodyfikuje rejestr pomocniczy przez operację odczytu / zapisu do tego samego lub innego 16-bit rejestru, wynik poprzedniej operacji będzie nieprawidłowy. Dlatego gdy główny program i procedura przerwania odwołują się przez rejestr pomocniczy, program główny powinien na ten czas wyłączyć przerwania.
Poniższy przykład pokazuje nierozłączny odczyt z TCNT1, dotyczy on również innych rejestrów.

Przykład w asemblerze:

```
TIM16_ReadTCNT1:
; zachowaj rejestr SREG
in r18 , SREG
; wyłącz przerwania
cli
; odczytaj TCNT1 do r17:r16
in r16 , TCNT1L
in r17 , TCNT1H
; przywróć rejestr SREG
out SRET , r18
ret
```

Przykład w C:

```
unsigned int TIM16_ReadTCNT1( void )
{
    unsigned char sreg;
    unsigned int i;
    /* zachowaj rejestr SREG */
    sreg = SREG;
    /* wyłącz przerwania */
    _CLI();
    /* odczytaj TCNT1 do i */
    i = TCNT1;
    /* przywróć SREG */
    SREG = sreg;
    return i;
}
```

Przykład poniżej pokazuje nierozłączny zapis do rejestrów 16-bit:

Przykład w asemblerze:

```
TIM16_WriteTCNT1:
; zachowaj rejestr SREG
in r18 , SREG
; wyłącz przerwania
cli
; zapisz do TCNT1 z r17:r16
out TCNT1L , r16
out TCNT1H , r17
; przywróć rejestr SREG
out SRET , r18
ret
```

Przykład w C:

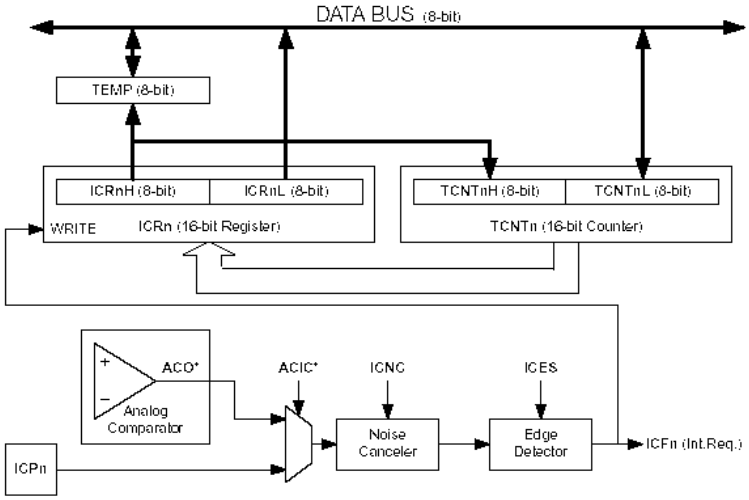
```
unsigned int TIM16_WriteTCNT1( void )
{
    unsigned char sreg;
    unsigned int i;
    /* zachowaj rejestr SREG */
    sreg = SREG;
    /* wyłącz przerwania */
    _CLI();
    /* wpisz i do TCNT1 */
    TCNT1 = i;
    /* przywróć SREG */
    SREG = sreg;
    return i;
}
```

Jeżeli zapisuje się do większej ilości 16-bitowych rejestrów, ale starszy bajt jest taki sam dla wszystkich rejestrów, rejestr pomocniczy potrzebny jest tylko raz do zapisu.

Źródła zegara licznika TC1:
Licznik może być taktowany zewnętrznym lub wewnętrznym sygnałem zegarowym. Źródło sygnału jest wybierane przez bity CS12:0 w rejestrze TCCR1B

Jednostka zliczająca:
Główna część licznika jest 16-bitową programowalną dwukierunkową jednostką.
Zawartość licznika jest dostępna w dwóch rejestrach: starszym TCNT1H i młodszym TCNT1L. Dostęp do TCNT1H jest tylko pośredni przez rejestr pomocniczy. Rejestr ten odczytuje lub zapisuje część starszą rejestru 16-bitowego, gdy CPU odczytuje lub zapisuje część młodszą.
W zależności od trybu pracy licznik jest zwiększany, zmniejszany lub zerowany w każdym cyklu zegara (clk_{T0}). Bit przepełnienia (TOV0) może być użyty do generowania przerwań. Wartość TCNT2 może być dostępna przez CPU bez względu na to czy sygnał clk_{T2} występuje czy nie.
Sposób pracy licznika jest zależny od bitów WGM31:0 umieszczonych w rejestrach sterujących A i B licznika TC1 (TCCR1A i TCCR1B).

Rejestr przechwytujący:
Funkcja przechwytywania może posłużyć do określenia czasu wystąpienia zewnętrznego sygnału powodującego przechwycenie. Sygnał ten może pochodzić z końcówki ICP1 lub z alternatywnej lub z komparatora analogowego. Funkcja ta może być użyta do obliczania częstotliwości, wypełnienia lub innych czynności związanych z obróbką sygnału.



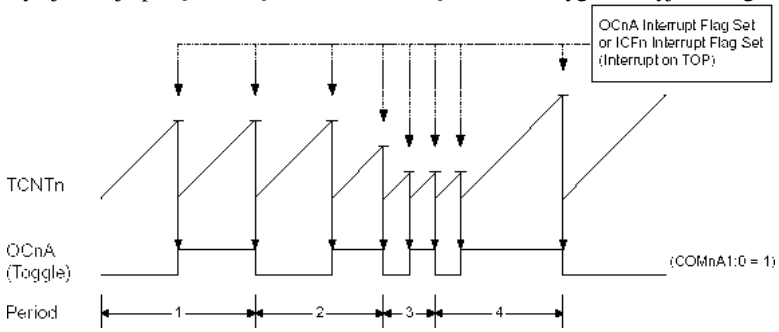
Zmiana stanu logicznego na końcówce ICP1 lub wyjściu komparatora analogowego (ACO) odpowiednio do ustawień, spowoduje przechwycenie zawartości licznika. 16-bitowa zawartość licznika (TCNT1) będzie wtedy wpisana do rejestru przechwycenia (ICR1). Flaga przechwycenia (ICF1) ustawi się w tym samym cyklu zegara gdy zawartość licznika będzie wpisywana do ICR1. jeśli bit TICIE jest ustawiony, Flaga przechwycenia powoduje wyzwolenie przerwania i ulega automatycznie wyzerowaniu gdy przerwanie jest wykonywane. Flaga ta może być też wyzerowana sprzętowo, gdy wpisze się do niej jedynkę.
Odczyt z 16-bit rejestru przechwycenia następuje najpierw z młodszej części (ICR1L) potem ze starszej (ICR1H). Gdy młodsza część jest odczytywana, starsza jest kopiowana do rejestru pomocniczego (TEMP). Gdy CPU doczytuje część starszą ICR1H z lokacji I/O, w rzeczywistości odwołuje się do rejestru TEMP.
Wpis do rejestru ICR1 może zajść tylko gdy użyty jest tryb generowania przebiegu. ICR1 określa wartość szczytową (TOP) do której zlicza licznik. Bity wyboru generowanego przebiegu (WGM13:0) muszą być ustawione, zanim wartość szczytowa będzie wpisana do ICR1.
Część starsza ICR1H musi być wpisana przed młodszą ICR1L.

Źródło wyzwalania:
Głównym źródłem wyzwalającym przechwycenie jest końcówka ICP1. Użyte może być też wyjście komparatora analogowego. Komparator jako źródło ustawiany jest przez: Włączenie komparatora (przez ustawienie bitu ACIC w rejestrze sterowania ACSIR). Zmiana źródła wyzwalania może spowodować zadziałanie przechwycenia. Dlatego Flaga przechwycenia musi być do tego czasu wyzerowana po zmianie.
Stan wejścia końcówki ICP1 i wyjścia komparatora (ACO) jest próbkowany tak samo jak wejście zegarowe licznika T1. Detektor impulsu jest taki sam. Po uaktywnieniu układu eliminującego zakłócenia, opóźnienie trwa cztery cykle systemowe. Jeżeli w trybie generowania przebiegu rejestr ICR1 służy do ustalania wartości szczytowej licznika, układ odkłócania i detektor impulsu są zawsze włączone.
Wyzwalanie przechwycenia może być sterowane programowo przez kontrolę końcówki ICP1.
Układ odkłócania:
Próbkowanie sygnału trwa cztery cykle zegara. Układ ten włącza się przez ustawienie bitu ICNC1 w rejestrze kontrolnym B licznika TC1 TCCR1B. Powoduje on opóźnienie o cztery cykle pomiędzy zmianą na końcówce wyzwalającej a przechwyceniem do rejestru ICR1. Układ odkłócania korzysta z systemowego zegara, bez preskalera.
Używanie układu przechwytującego:
Jeśli procesor nie może odczytać wartości przechwyconej w rejestrze ICR1 przed następnym przechwyceniem, wartość w ICR1 będzie zastąpiona nową.
Gdy używane jest przerwanie do przechwycenia, rejestr ICR1 powinien być odczytany w programie obsługi przerwania najwcześniej jak to możliwe. Pomimo, że przerwanie to ma wysoki priorytet, maksymalny czas wykonywania przerwania jest zależny od maksymalnej liczby cykli potrzebnej do obsługi odpowiedzi na dowolne przerwanie.
Używanie układu przechwytującego podczas zmian wartości szczytowej licznika (pracującego w dowolnym trybie) nie jest zalecane.
Odmierzenie wypełnienia zewnętrznego sygnału wymaga zmiany zbocza wyzwalającego po każdym przechwyceniu. Zmiana ta musi być dokonana jak najszybciej po odczycie rejestru ICR1. Po zmianie Flaga przechwycenia ICF1 musi być wyzerowana programowo (przez wpis jedynki). Przy odmierzaniu częstotliwości kasowanie flagi ICF1 nie jest konieczne (jeżeli jest używane przerwanie).

Układ porównujący:
16-bitowy komparator ciągle porównuje zawartość licznika (TCNT1) z rejestrem OCR1x. Osiągnięcie przez TCNT takiej samej wartości jak OCR1x jest przez komparator sygnalizowane. Spowoduje ono ustawienie bitu OCF1x w następnym cyklu zegara. Jeżeli bity OCIE1x i globalny system przerwań I w SREG są ustawione to jedynka we fladze OCF1x wygeneruje przerwanie. OCF1x jest automatycznie kasowany w trakcie wykonywania przerwania. Może być również kasowany programowo przez wpisanie jedynki. Porównanie wykorzystywane jest do wytwarzania przebiegu prostokątnego, przy odpowiednim ustawieniu bitów WGM13:0 i COM1x1:0. Sygnały wartości szczytowej lub maksymalnej są używane do generowania przebiegów.
Układ porównujący A pozwala na zdefiniowanie wartości szczytowej (TOP) do której będzie liczył licznik (rozdzielczość), dodatkowo określa okres generowanego przebiegu.
Rejestr OCR1x jest podwójnie buforowany gdy licznik pracuje w trybie PWM. Podczas normalnej pracy i auto-ładowania (CTC) podwójne buforowanie jest wyłączone. Buforowanie to synchronizuje wpis do rejestru OCR1x i zapobiega w związku z tym pojawieniu się niesymetrycznych sygnałów PWM.
Gdy podwójne buforowanie jest aktywne, CPU ma dostęp do OCR1x przez bufor, gdy jest wyłączone dostęp do tego rejestru jest bezpośredni. Zawartość rejestru OCR1x może być zmieniona tylko przez wpisanie do niej wartości (Licznik nie uaktualnia wartości tego rejestru tak jak TCNT1 czy ICR1). Dlatego OCR1x nie jest odczytywany przez rejestr dodatkowy TEMP. Jednak zalecane jest odczytywanie najpierw młodszego bajtu a później starszego z 16-bit rejestrów. Zapis do OCR1x musi być dokonane przez rejestr pośredniczący TEMP, ponieważ wszystkie szesnaście bitów tego rejestru są jednocześnie porównywane z zawartością licznika. Więc należy najpierw zapisać bajt starszy a potem młodszy.
FOC:
W trybach innych niż PWM wyjście porównania komparatora może być wymuszone przez wpis jedynki do bitu FOC1x. Wymuszenie to nie ustawi bitu OCF1x ani nie spowoduje zerowania / auto-ładowania licznika, ale końcówka OC1x będzie uaktualniona gdy porównanie rzeczywiście wystąpi. (o ile COM11:0 zostaną odpowiednio skonfigurowane).
Blokowanie układu porównania przez zapis do TCNT1:
Wpisywanie przez CPU wartości do TCNT1 zablokowuje układ porównania do pojawienia się następnego cyklu zegara, nawet gdy licznik jest zatrzymany. Umożliwia to ustawienie rejestru o takiej samej wartości jak TCNT1 bez wyzwalania przerwania gdy licznik pracuje.

Używanie układu porównującego:
Blokowanie na jeden takt zegara układu porównującego podczas wpisu do TCNT1 może powodować pewne problemy gdy używa się wyjścia porównującego, niezależnie do tego czy licznik pracuje, czy nie. Jeśli wartość wpisana do TCNT1 jest równa wartości w OCR1x to porównanie nie zajdzie, co spowoduje nieprawidłowe wygenerowanie sygnału. Nie należy wpisywać do TCNT1 wartości szczytowej TOP w trybie PWM, gdy wartość ta jest zmienna, ponieważ wynik układu porównującego może być zignorowany i licznik będzie liczył do końca (do 0xFFFF). Podobnie nie należy wpisywać wartości 0x00 gdy licznik liczy w dół.
Ustawienie do pracy OC1x powinno zajść zanim ustawi się kierunek portu jako wyjście. Najprostszą metodą na ustawienie wartości OC1x jest użycie Wymuszenia Wyjścia Porównania (FOC1x) w zwykłym trybie pracy. Rejestr OC1x przechowuje zawartość nawet podczas zmian między trybami generowania przebiegów.
Bity COM1x1:0 nie są podwójnie buforowane razem z wartością porównywaną. Zmiana ustawień bitów COM1x1:0 da natychmiastowy skutek.
Wyjściowa jednostka porównująca:
Bity wyjścia porównania (COM1x1:0) posiadają dwie funkcje. Generator przebiegu prostokątnego używa tych bitów do zdefiniowania stanu wyjścia porównania (OC1x) przy następnym porównaniu. Stan OC1x dotyczy wewnętrznego rejestru, a nie końcówki.
Jeśli system zostanie zresetowany, rejestr OC1x jest ustawiony na „0".
Podstawowe funkcje portu I/O nie są zachowane, gdy uaktywnione są wyjścia porównania OC1x (przez ustawienie któregoś z bitów COM1x1:0). Jednak kierunek portu na końcówce OC1x jest kontrolowany przez rejestr kierunku portu DDR. Bit w DDR odpowiadający tej końcówce musi być ustawiony na 1, żeby sygnał wystąpił na OC1x.
Tryby pracy generatora przebiegów i układu porównującego:
Generator przebiegu używa bitów COM1x1:0 w trybach Normal, CTC i modulacjach PWM. We wszystkich trybach wpis COM1x1:0 = 0 powoduje że OC2 nie zadziała przy następnym porównaniu. Zmiana zawartości bitów COM21:0 da efekt przy pierwszym porównaniu po zmodyfikowaniu tych bitów. W trybach innych niż PWM, zadziałanie OC2 można wymusić natychmiastowo przez użycie bitu zbocza FOC2.

Tryby pracy:
Poszczególne tryby pracy licznika (np. generowanie przebiegu; PWM) uzyskiwane są przez ustawienie bitów Generatora Przebiegu (WGM13:0) i Wyjścia Porównania (COM1x1:0). Bity COM nie będą oddziaływać na sposób zliczania w trybie generowania przebiegu. Kontrolują czy wyjście PWM będzie odwrócone lub nie odwrócone.
Dla trybów innych niż PWM bity COM1x1:0 kontrolują czy wyjście jest ustawione, wyzerowane, albo zmienione podczas porównania.
Tryb normalny:
Jest to najprostszy tryb pracy (bity WGM13:0 = 0). W tym trybie licznik zawsze liczy w górę i jest zerowany po przepełnieniu wartością 16-bitową (gdy zliczy do 0xFFFF). Flaga przepełnienia (TOV1) jest ustawiany w tym samym cyklu zegara, gdy licznik zeruje się. Może być traktowany jako 17 bit licznika, z tym że jest tylko ustawiany, nie kasowany. Jednak fakt automatycznego zerowania bitu TOV1 w skutek przyjęcia przerwania może posłużyć do programowego zwiększenia rozdzielczości licznika. Nowa wartość licznika może być wpisywana w dowolnej chwili.
W trybie Normal wejście przechwytujące jest proste w użyciu. Maksymalny czas między momentami przechwycenia nie może być dłuższy niż wynikało by to z pojemności licznika. Jeśli czas ten jest zbyt długi, należy użyć preskalera lub przerwania od przepełnienia (dla zasygnalizowania faktu przepełnienia się licznika).
Wyjście porównania może być użyte do wyzwalania przerwania w ustalonym czasie. Użycie wyjścia porównania do generowania przebiegu w trybie Normal nie jest zalecane, ponieważ zajmuje to za dużo czasu procesorowi.
Zerowanie licznika w trybie porównania (CTC):
W trybie zerowania licznika podczas porównania (ustawienie bitów WGM13:0 = 4 lub 12) używany jest rejestr OCR1A lub ICR1 do określenia rozdzielczości licznika. W trybie CTC licznik jest zerowany, gdy zliczy do takiej wartości jaka jest wpisana do rejestru OCR1A (WGM13:0 = 4) lub ICR1 (WGM13:0 = 12). Rejestr ten definiuje największą wartość do której może doliczyć licznik i w ten sposób decyduje o jego rozdzielczości. Ten tryb jest najlepszą metodą na określenie częstotliwości sygnału wyjściowego.



Przerwanie może być generowane podczas przepełniania się licznika przez użycie bitu OCF1A lub ICF1.
Gdy przerwania są aktywne, procedura obsługi przerwania może być wykorzystana do zmiany wartości maksymalnej, do której zlicza licznik.
Przerwanie może zmienić wartość szczytową dla licznika, jakkolwiek zmiana wartości szczytowej zbliżonej do zera, podczas gdy licznik pracuje z wyłączonym przeskalerem, lub ustawionym na niewielki podział, muszą być dokonywane ostrożnie, ponieważ w trybie CTC nie działa podwójne buforowanie. Jeżeli wpisana nowa wartość do OCR1A lub ICR1 jest mniejsza od aktualnej wartości w TCNT1, licznik opóźni proces porównania. Licznik będzie musiał zliczyć do wartości maksymalnej (0xFFFF) i po wyzerowaniu zliczyć do wartości przy której wystąpi porównanie. W większości przypadków ta cecha nie jest pożądana. Alternatywą będzie użycie w trybie Fast PWM OCR1A do określenia wartości szczytowej (TOP) – (WGM13:0 = 15), ponieważ OCR1A jest podwójnie buforowane.
W celu generowania przebiegu prostokątnego w trybie CTC, (wyjście OC1A zmieniać się na przeciwne), należy ustawić bity COM1A1:0 = 1. Wartość OC1A pojawi się na końcówce portu we/wy gdy w rejestrze kierunku portu końcówkę tą ustawi się jako wyjście.
Generowany przebieg osiągnie najwyższą częstotliwość równą $f_{OC1}=f_{CLK\_I/O}/2$, gdy do OCR1A wpisane jest zero (0x0000). Częstotliwość przebiegu wyrażona jest wzorem:

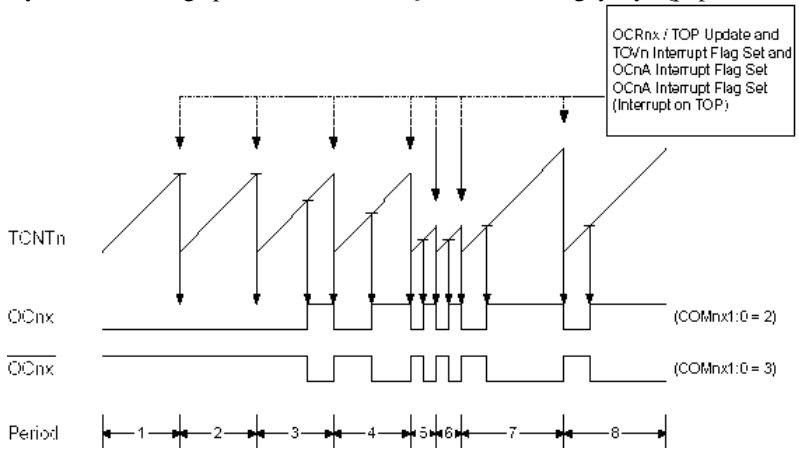
$$f_{CCnA} = \frac{f_{CLK\_I/O}}{2 \cdot N \cdot (1 + OCRnA)}$$

gdzie N jest podziałem preskalera (przyjmuje wartości 1, 8, 64, 256 lub 1024).
Tak jak w trybie Normal, bit przepełnienia TOV1 jest ustawiany w tym samym cyklu zegara w którym następuje wyzerowanie licznika w skutek przepełnienia.

Tryb Fast PWM:
Tryb ten umożliwia generowanie szybkiego przebiegu PWM (ustawienie bitów WGM13:0 = 5,6,7,14 lub 15). Tryb ten różni się od pozostałych operacją pojedynczego zliczania. Licznik zlicza od zera do maksymalnej (0xFFFF) i ponownie zaczyna od zera. W trybie bez odwrócenia wyjścia, wyjście porównania OC1x jest ustawiane gdy wartość licznika jest większa lub równa wartości w komparatorze i zerowane gdy licznik jest zerowany. W trybie z odwróconym wyjściem OC1x jest zerowane podczas porównania i ustawiane przy zerowaniu.
Częstotliwość trybu Fast PWM może być dwa razy wyższa od trybu Correct Pwm używającego operacji podwójnego zliczania. Tryb ten jest odpowiedni do regulacji mocy, przetwarzania cyfrowo-analogowego. Wysoka częstotliwość PWM umożliwia zmniejszenie rozmiarów zewnętrznych elementów współpracujących – cewek czy kondensatorów i obniża koszty wykonania.
Rozdzielczość modulatora PWM może wynosić 8, 9 lub 10 lub może być zdefiniowana przez ICR1 lub OCR1A. Minimalną rozdzielczością są 2-bity (ICR1 lub OCR1A = 0x0003), a maksymalna 16-bitowa (ICR1 lub OCR1A ustawione na maksimum – 0xFFFF). Rozdzielczość może być policzona ze wzoru:

$$R_{FPWM} = \frac{\log(TOP+1)}{\log(2)}$$

Licznik jest zwiększany aż osiągnie jedną z wartości 0x00FF, 0x01FF, 0x03FF (WGM13:0 = 5,6 lub 7), wartość w ICR1 (WGM13:0 = 14) lub OCR1A(WGM13:0 = 15) wtedy, w następnym cyklu zegara następuje jego wyzerowanie. Flaga przerwania OC1x będzie ustawiona gdy wystąpi porównanie.



Bit przepełnienia (TOV1) jest ustawiany podczas osiągnięcia przez licznik wartości maksymalnej. Dodatkowo flagi OC1A i ICF1 są ustawiane w tym samym cyklu zegara licznika co TOV1 (jest ustawiany, gdy OCR1A lub ICR1 jest użyty do określenia wartości szczytowej licznika). Jeżeli przerwania są aktywne, to w procedurze przerwania można zmienić wartość rejestru porównania.
Podczas gdy zmieniana jest wartość szczytowa TOP, program musi zapewnić, że nowa wartość TOP jest większa lub równa wartości wszystkich rejestrów porównujących. Jeżeli TOP byłoby niższe od zawartości któregoś z rejestrów porównujących, porównanie nigdy nie wystąpi między TCNT1 a OCR1x.
Procedura uaktualnienia rejestru ICR1 różni się od procedury dla OCR1A gdy używane są do określenia wartości TOP, ponieważ rejestr ICR1 nie jest podwójnie buforowany. Oznacza to, że jeśli do ICR1 wpisywana jest niska wartość gdy licznik pracuje bez lub z niskim podziałem preskalera, wartość ta jest niższa od aktualnej wartości TCNT1. Porównanie będzie opóźnione. Licznik doliczy do maksimum (0xFFFF) i wyzeruje się zanim nastąpi porównanie. Rejestr OCR1A jest natomiast podwójnie buforowany. Dzięki temu do OCR1A można wpisywać w dowolnym czasie. Gdy następuje wpis, wartość wpisywana jest do rejestru pomocniczego. Stamtąd zostanie przepisana do OCR1A w tym samym cyklu zegara, gdy nastąpi przepełnienie i wyzerowanie licznika z ustawieniem flagi TOV1.
ICR1 może być użyty do zdefiniowania wartości szczytowej do której zlicza licznik (TOP) gdy wartość ta jest stała. Dzięki temu drugi rejestr – OCR1A jest wolny i może być użyty do generowania przebiegu PWM. Ale jeśli częstotliwość nośna przebiegu PWM jest zmieniana (przez zmianę wartości TOP), najlepszą opcją jest użycie do tego celu rejestru OCR1A ze względu na podwójne buforowanie.
W trybie Fast PWM, jednostka porównująca pozwala na generowanie przebiegu PWM na końcówkach OC1x. Ustawienie COM1x1:0 = 2 spowoduje, że odwrócony i nie odwrócony przebieg może być wytworzony przez ustawienie bitów COM1x1:0 = 3. Przebieg pojawi się na końcówce tylko gdy kierunek portu dla tek końcówki będzie ustawiony jako wyjście (DDR_OC1x = 1). Przebieg prostokątny jest wytwarzany przez ustawienie (lub wyzerowanie) OC1x przy porównaniu między rejestrami OCR1x i TCNT1 i zerowane (ustawiane) przy zerowaniu licznika (przejściu z wartości maksymalnej 0xFFFF do 0x0000).

Częstotliwość nośna obliczana jest ze wzoru:

$$f_{OCnxPWM} = \frac{f_{CLK\_I/O}}{N \cdot (1+TOP)}$$

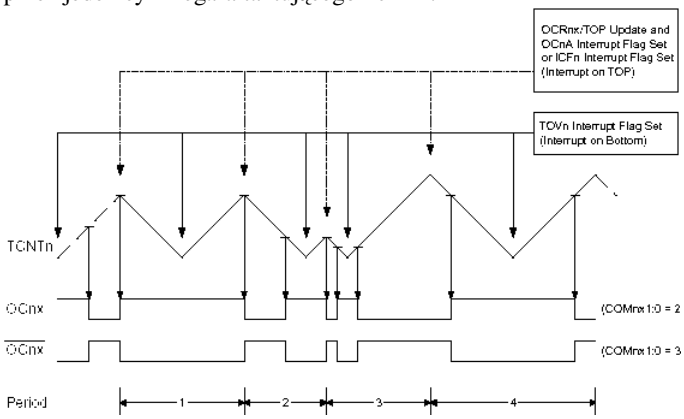
gdzie N jest podziałem preskalera (przyjmuje wartości 1, 8, 64, 256 lub 1024).

Jeżeli do rejestru OCR1x jest wpisane zero, na wyjściu pojawiać się będzie krótka szpilka przy takcie zegara w którym licznik się przepełnia. Ustawienie OCR1x na maksimum (0xFFFF) ustawi wyjście w stanie niskim lub wysokim (zależnie od bitów COM1x1:0).
Przebieg z 50% wypełnieniem przy częstotliwości $f_{oc1}=f_{CLK\_I/O}/2$ uzyska się zmieniając OC1A w każdym porównaniu (COM1A1:0 = 1). Stosowane jest to tylko gdy OCR1A jest używane do zdefiniowania wartości TOP (WGM13:0 = 15). Przebieg będzie miał maksymalną częstotliwość $f_{OC1A} = f_{clk\_I/O}/2$ gdy OCR1A będzie równa 0x0000.

PWM z poprawną fazą:
Ten tryb daje wysokiej rozdzielczości PWM z poprawną fazą (bity WGM13:0 ustawione na 1,2,3,10 lub 11). Zasada działania, podobnie jak w trybie z poprawną częstotliwością, opiera się na podwójnym zliczaniu. Licznik powtarza zliczanie od wartości zerowej do maksymalnej i na odwrót. W trybie bez odwracania wyjść OC1x jest zerowane przy porównaniu pomiędzy TCNT1 a OCR1x podczas zliczania w górę i ustawiany podczas porównania przy zliczaniu w dół. W trybie z zanegowanym wyjściem sytuacja jest odwrotna. System podwójnego zliczania daje mniejszą częstotliwość niż we wcześniejszym trybie. Dzięki uzyskanej w ten sposób symetrii tryb ten jest zalecany do sterowania silnikami.
Rozdzielczość modulatora PWM może wynosić 8, 9 lub 10 lub może być zdefiniowana przez ICR1 lub OCR1A. Minimalną rozdzielczością są 2-bity (ICR1 lub OCR1A = 0x0003), a maksymalna 16-bitowa (ICR1 lub OCR1A ustawione na maksimum – 0xFFFF). Rozdzielczość może być policzona ze wzoru:

$$R_{FPWM} = \frac{\log(TOP+1)}{\log(2)}$$

Licznik jest zwiększany aż osiągnie jedną z wartości 0x00FF, 0x01FF, 0x03FF (WGM13:0 = 1,2 lub 3), wartość w ICR1 (WGM13:0 = 10) lub OCR1A(WGM13:0 = 11). Licznik jest zwiększany dopóki nie osiągnie maksimum – wtedy zmienia kierunek liczenia. Rejestr TCNT1 będzie miał wartość równą maksimum (0xFFFF) przez jeden cykl zegara taktującego licznik.



Bit przepełnienia (TOV1) jest ustawiany przy dojściu licznika do wartości BOTTOM. Gdy któryś z OCR1A lub ICR1 jest użyty do określenia wartości TOP, flagi OC1A lub ICF1 są ustawiane w tym samym cyklu zegara co rejestr OCR1x, gdy jest uaktualniany przez bufor dodatkowy (przy wartości TOP) (?). Flagę przerwania można użyć do generowania przerwania za każdym razem gdy licznik dotrze do wartości minimalnej (BOTTOM) lub szczytowej (TOP). Podczas gdy zmieniana jest wartość szczytowa TOP, program musi zapewnić, że nowa wartość TOP jest większa lub równa wartości wszystkich rejestrów porównujących. Jeżeli TOP byłoby niższe od zawartości któregoś z rejestrów porównujących, porównanie nigdy nie wystąpi między TCNT1 a OCR1x. Odkąd OCR1x modyfikuje wartość TOP, okres PWM zaczyna się i kończy na wartości TOP.
Tryby poprawnej fazy i częstotliwości są zalecane, gdy wartość TOP zmieniana jest podczas pracy licznika. Jeżeli wartość TOP jest stała, nie ma różnic pomiędzy tymi trybami.
Komparator powoduje generowanie przebiegu na końcówce OC1x. Ustawienie bitów COM1x1:0 na 2 uruchomi nie odwrócony PWM. Tryb z zanegowanym wyjściem dostępny jest przy COM1x1:0 ustawionych na 3. Bieżący stan OC1x będzie odzwierciedlony na końcówce OC1x gdy będzie ona ustawiona jako wyjście w rejestrze kierunku portu.

Przebieg PWM jest uzyskiwany przez ustawienie (lub wyzerowanie) OC1x podczas porównania OCR1x i TCNT1 gdy licznik zlicza w górę i zerowana (lub ustawiana) przy porównaniu OCR1x i TCNT1 podczas pracy licznika w dół. Częstotliwość licznika określona jest zależnością:

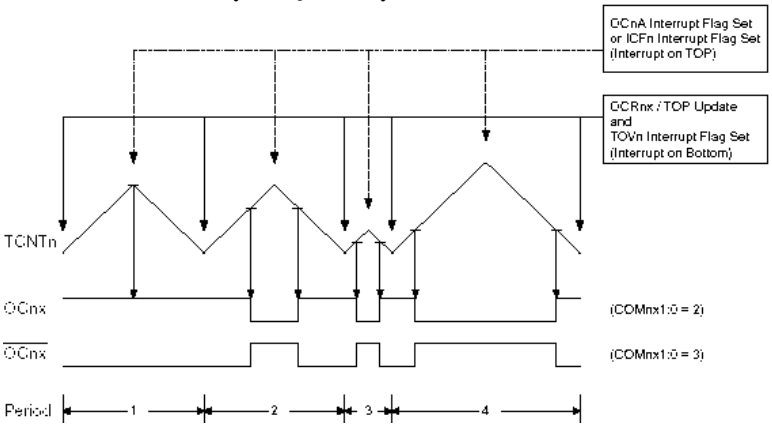
$$f_{OCnxPWM} = \frac{f_{CLK\_I/O}}{N \cdot (1+TOP)}$$

gdzie N jest podziałem preskalera (przyjmuje wartości 1, 8, 32, 64, 128, 256 lub 1024).
Jeśli do rejestru OCR1x wpisana jest wartość minimalna (0x00), wyjście będzie ciągle w stanie niskim; jeśli będzie to wartość maksymalna (0xFF) to wyjście będzie w stanie wysokim (dla trybu bez odwracania wyjść – z inwersją odwrotnie). Gdy OCR1A określi wartość TOP (WGM13:0 = 11) i COM1A1:0 = 1, wyjście OC1A będzie się zmieniać z 50% wypełnieniem.

Tryb PWM z poprawną faza i częstotliwością:
Umożliwia generowanie przebiegu PWM o wysokiej rozdzielczości z poprawną fazą i częstotliwością (WGM13:0 = 8 lub 9). Tryb ten podobnie jak poprzedni opiera się na podwójnym zliczaniu. . Licznik powtarza zliczanie od wartości zerowej do maksymalnej i na odwrót. W trybie bez odwracania wyjść OC1x jest zerowane przy porównaniu pomiędzy TCNT1 a OCR1x podczas zliczania w górę i ustawiany podczas porównania przy zliczaniu w dół. W trybie z zanegowanym wyjściem sytuacja jest odwrotna. System podwójnego zliczania daje mniejszą częstotliwość niż we wcześniejszym trybie. Dzięki uzyskanej w ten sposób symetrii tryb ten jest zalecany do sterowania silnikami.
Różnicą między trybem z poprawną fazą a trybem z poprawną fazą i częstotliwością jest moment wymiany wartości pomiędzy rejestrem OCR1x a jego buforem. Rozdzielczość może być policzona ze wzoru:

$$R_{FPWM} = \frac{\log(TOP+1)}{\log(2)}$$

Licznik jest zwiększany aż osiągnie wartość w ICR1 (WGM13:0 = 10) lub OCR1A(WGM13:0 = 11). Licznik jest zwiększany dopóki nie osiągnie maksimum – wtedy zmienia kierunek liczenia. Podczas gdy zmieniana jest wartość szczytowa TOP, program musi zapewnić, że nowa wartość TOP jest większa lub równa wartości wszystkich rejestrów porównujących. Jeżeli TOP byłoby niższe od zawartości któregoś z rejestrów porównujących, porównanie nigdy nie wystąpi między TCNT1 a OCR1x. Odkąd OCR1x modyfikuje wartość TOP, okres PWM zaczyna się i kończy na wartości TOP.



W trybie Fast PWM, jednostka porównująca pozwala na generowanie przebiegu PWM na końcówkach OC1x. Ustawienie COM1x1:0 = 2 spowoduje, że odwrócony i nie odwrócony przebieg może być wytworzony przez ustawienie bitów COM1x1:0 = 3. Przebieg pojawi się na końcówce tylko gdy kierunek portu dla tek końcówki będzie ustawiony jako wyjście (DDR_OC1x = 1). Przebieg prostokątny jest wytwarzany przez ustawienie (lub wyzerowanie) OC1x przy porównaniu między rejestrami OCR1x i TCNT1 i zerowane (ustawiane) przy zerowaniu licznika (przejściu z wartości maksymalnej 0xFFFF do 0x0000).
Częstotliwość nośna obliczana jest ze wzoru:

$$f_{OCnxPWM} = \frac{f_{CLK\_I/O}}{N \cdot (1+TOP)}$$

gdzie N jest podziałem preskalera (przyjmuje wartości 1, 8, 64, 256 lub 1024).
Jeśli do rejestru OCR1x wpisana jest wartość minimalna (0x00), wyjście będzie ciągle w stanie niskim; jeśli będzie to wartość maksymalna (0xFF) to wyjście będzie w stanie wysokim (dla trybu bez odwracania wyjść – z inwersją odwrotnie). Gdy OCR1A określi wartość TOP (WGM13:0 = 11) i COM1A1:0 = 1, wyjście OC1A będzie się zmieniać z 50% wypełnieniem.

Opis rejestrów 16-bitowego licznika

Rejestr sterujący A licznikiem TCCR1A:

Bit	7	6	5	4	3	2	1	0	
	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	TCCR1A
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z	Z	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bity 7,6 – COM1A1:0 – rodzaj wyjścia trybu porównania A**
- **bity 5,4 – COM1B1:0 – rodzaj wyjścia trybu porównania B**

określają zachowanie się wyjścia komparatora licznika (końcówki OC1x). Jeśli jeden z bitów lub oba bity COM1A1:0 lub COM1B1:0 są ustawione to wyjście układu porównującego w liczniku zostaje dołączone do końcówki portu I/O. Rejestr kierunku portu (DDR) musi ustawić port jako wyjście.
Gdy OC1A lub OC1B jest dołączone do końcówek portu, funkcje bitów COM1x1:0 zależą od ustawienia bitów WGM13:0
Tryb porównania (bez PWM)

COM1A1/COM1B1	COM1A0/COM1B0	Opis
0	0	Zwykły tryb pracy portu, OC1A/OC1B odłączone
0	1	Porównanie zmienia OC1A/OC1B
1	0	Porównanie zeruje OC1A/OC1B
1	1	Porównanie ustawia OC1A/OC1B

Tryb Fast PWM i porównania [(1)]

COM1A1/COM1B1	COM1A0/COM1B0	Określenie
0	0	Zwykły tryb pracy portu, OC1A/OC1B odłączone
0	1	WGM13:0 = 15: zmienia OC1A podczas porównania, OC1B – odłączone (normalna praca portu). Dla pozostałych ustawień WGM13:0 zwykły tryb pracy portu, OC1A,OC1B odłączone
1	0	Porównanie zeruje OC1A/OC1B, ustawia przy wartości szczytowej
1	1	Porównanie ustawia OC1A/OC1B, zeruje przy wartości szczytowej

Tryb PWM z poprawną fazą, PWM z poprawną fazą i częstotliwością, i porównania [(1)]

COM1A1/COM1B1	COM1A0/COM1B0	Określenie
0	0	Zwykły tryb pracy portu, OC1A/OC1B odłączone
0	1	WGM13:0 = 14: zmienia OC1A podczas porównania, OC1B – odłączone (normalna praca portu). Dla pozostałych ustawień WGM13:0 zwykły tryb pracy portu, OC1A,OC1B odłączone
1	0	Porównanie zeruje OC1A/OC1B gdy licznik liczy w górę, ustawia gdy licznik liczy w dół
1	1	Porównanie ustawia OC1A/OC1B gdy licznik liczy w górę, zeruje gdy licznik liczy w dół

(2) – występuje specjalny przypadek gdy OCR1A/OCR1B jest takie samo jak wartość szczytowa licznika i COM1A1/COM1B1 są ustawione.

- **bit 3 – FOC1A: Wymuszenie wyjścia porównania A**
- **bit 2 – FOC1B: Wymuszenie wyjścia porównania B**

bity FOC1A/FOC1B są aktywne tylko wtedy, gdy WGM ustawiają licznik w trybie innym niż PWM. Ze względu na zgodność z przyszłymi układami bit ten musi być zerowany gdy poprzez rejestr TCCR1A licznik jest skonfigurowany jako PWM. Wpisanie logicznej jedynki do FOC1A/FOC1B wymusza natychmiastowe porównanie.
Przejście stanu FOC1A/FOC1B nie generuje przerwania ani nie wyzeruje licznika w trybie CTC gdy OCR1A jest jako wartość maksymalna.
FOC1A/FOC1B zawsze odczytywane jest jako zero.

- **bity 1,0 – WGM11:0 – rodzaje generowanych przebiegów:**

bity te, razem z bitami WGM13:0 z rejestru TCCR1B, kontrolują sposób zliczania licznika: wartość maksymalną do której dąży licznik i rodzaj wytwarzanego przebiegu. Tryby pracy licznika: zwykły, zerowanie licznika podczas porównania i trzy rodzaje PWM (modulacji szerokości impulsu).

Tryb	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Tryb pracy licznika	TOP	Przepisanie z OCR1x	Moment ustawienia flagi TOV1
0	0	0	0	0	Zwykły	0xFFFF	Natychmiast	MAX
1	0	0	0	1	PWM, poprawna faza, 8-bit	0x00FF	TOP	0x0000
2	0	0	1	0	PWM, poprawna faza, 9-bit	0x01FF	TOP	0x0000
3	0	0	1	1	PWM, poprawna faza, 10-bit	0x03FF	TOP	0x0000
4	0	1	0	0	CTC	OCR1A		MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	TOP	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	TOP	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	TOP	TOP
8	1	0	0	0	PWM, popr. faza i częstotliwość	ICR1	0x0000	0x0000
9	1	0	0	1	PWM, popr. faza i częstotliwość	OCR1A	0x0000	0x0000
10	1	0	1	0	PWM, poprawna faza	ICR1	TOP	0x0000
11	1	0	1	1	PWM, poprawna faza	OCR1A	TOP	0x0000
12	1	1	0	0	CTC	ICR1	natychmiast	MAX
13	1	1	0	1	Zarezerwowane	-	-	-
14	1	1	1	0	Fast PWM	ICR1	TOP	TOP
15	1	1	1	1	Fast PWM	OCR1A	TOP	TOP

Rejestr sterujący B licznikiem TCCR1B:

Bit	7	6	5	4	3	2	1	0	
	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Odczyt/zapis	Z/O	Z/O	O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – ICNC1: odkłócanie wejścia przechwytującego:**

Ustawienie tego bitu włącza odkłócanie wejścia wyzwalającego przechwytywanie. Wejście to (końcówka ICP1) jest wtedy filtrowana. Proces ten wymaga czterokrotnego próbkowania końcówki ICP1. Przechwycenie następuje wtedy z opóźnieniem o cztery cykle zegara w stosunku do pojawienia się sygnału na końcówce.

- **bit 6 – ICES1: ustawianie zbocza do przechwycenia:**

ICES1 = 0 powoduje uruchomienie przechwycenia podczas opadającego zbocza na końcówce ICP1. ICES1 = 1 – podczas narastającego.

Wyzwolenie przechwycenia powoduje skopiowanie zawartości licznika do rejestru ICR1. Ustawiona zostanie wtedy flaga przechwycenia ICF1 i uruchomione przerwanie, o ile system przerwań jest aktywny.

Gdy rejestr ICR1 jest użyty do jako wartość szczytowa, do której zlicza licznik, ICP1 jest odłączone od końcówki i funkcja przechwytywania nie działa.

- **bit 5 – zarezerwowany:**

Może być użyty w przyszłych układach. Do tego bitu należy wpisywać zero, gdy zapisywany jest rejestr TCCR1B.

- **bity 4,3 – WGM13:2 – rodzaje generowanego przebiegu:**

tak jak w opisie TCCR1A

- **bity 2...0 – CS12:0 – wybór sygnału zegarowego:**

CS12	CS11	CS10	Opis
0	0	0	Brak sygnału zegarowego – licznik jest zatrzymany
0	0	1	$clk_{I/O}$ (pełna prędkość – bez preskalera)
0	1	0	$clk_{I/O}$ / 8 (z preskalerem)
0	1	1	$clk_{I/O}$ / 64 (z preskalerem)
1	0	0	$clk_{I/O}$ / 256 (z preskalerem)
1	0	1	$clk_{I/O}$ / 1024 (z preskalerem)
1	1	0	Zewnętrzny sygnał zegarowy na końcówce T1 – takt przy zboczu opadającym
1	1	1	Zewnętrzny sygnał zegarowy na końcówce T1 – takt przy zboczu narastającym

Jeżeli ustawione jest zewnętrzne źródło zegara dla licznika TC0, licznik będzie mógł być napędzany nawet jeśli port ustawiony jest jako wyjście. Umożliwia to programowe taktowanie licznika (przez zmianę stanu końcówki portu).

Rejestr licznika TCNT1H i TCNT1L:

Bit	7	6	5	4	3	2	1	0	
	TCNT1[15:8]								TCNT1H
	TCNT1[7:0]								TCNT1L
Odczyt/zapis	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestry TCNT1H i TCNT1L tworzące wspólnie TCNT1 dają bezpośredni dostęp, poprzez operacje zapisu i odczytu, do stanu licznika (licznik jest 16-bitowy). Zapis i odczyt do młodszej i starszej części odbywa się jednocześnie dzięki obecności 8-bitowego rejestru pomocniczego TEMP. Wpis do TCNT1 zmienia układ porównujący na następny cykl zegara.
Modyfikacja tego rejestru w trakcie pracy licznika grozi pominięciem chwili porównania między TCNT1 a OCR1x.

Rejestry porównujące 1A – OCR1AH i OCR1AL:

Bit	7	6	5	4	3	2	1	0	
	OCR1A [15:8]								OCR1AH
	OCR1A [7:0]								OCR1AL
Odczyt/zapis	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestry porównujące 1B – OCR1BH i OCR1BL:

Bit	7	6	5	4	3	2	1	0	
	OCR1B [15:8]								OCR1BH
	OCR1B [7:0]								OCR1BL
Odczyt/zapis	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestry porównania zawierają 16-bit wartość stale porównywaną z wartością licznika (TCNT1). Porównanie może być użyte do generowania przerwania, lub przebiegu na końcówce OC1x.
Rejestry te posiadają 16-bitów. CPU ma dostęp jednocześnie do starszej i młodszej części dzięki 8-bit rejestrowi pomocniczemu.

Rejestr przechwytujący 1 – ICR1H i ICR1L:

Bit	7	6	5	4	3	2	1	0	
	ICR1 [15:8]								ICR1H
	ICR1 [7:0]								ICR1L
Odczyt/zapis	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr ten jest zapisywany zawartością licznika TCNT1 wtedy gdy pojawi się odpowiedni stan na końcówce ICP1 (ewentualnie zmieni się stan komparatora analogowego). Może być użyty do zdefiniowania wartości szczytowej TOP licznika.
Rejestry te posiadają 16-bitów. CPU ma dostęp jednocześnie do starszej i młodszej części dzięki 8-bit rejestrowi pomocniczemu.

Rejestr maskowania przerwań TIMSK:

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 5 – TICIE1: przerwanie od przechwycenia licznika TC1**

Ustawienie tego bitu i I w SREG powoduje uaktywnienie przerwania. Procedura obsługi przerwania będzie wykonana, gdy flaga ICF1 w TIFR będzie ustawiona.

- **bit 4 – OCIE1A: uaktywnienie przerwania od układu porównującego A:**

Ustawienie bitu OCIE1A i I w SREG w stan 1 powoduje uaktywnienie przerwania pochodzącego od układu porównującego licznika. Odpowiednie przerwanie jest wykonywane, gdy ustawiony jest bit OCF1A w rejestrze flag przerwań licznika – TIFR.

- **bit 3 – OCIE1B: uaktywnienie przerwania od układu porównującego A:**

Ustawienie bitu OCIE1B i I w SREG w stan 1 powoduje uaktywnienie przerwania pochodzącego od układu porównującego licznika. Odpowiednie przerwanie jest wykonywane, gdy ustawiony jest bit OCF1B w rejestrze flag przerwań licznika – TIFR.

- **bit 2 – TOIE1: uaktywnienie przerwania przy przepełnieniu**

Przerwanie jest aktywne gdy bit TOIE1 i I w SREG są ustawione.

Rejestr flag przerwań licznika TC1 TIFR:

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 5 – ICF1: Flaga przechwycenia w liczniku TC1**

Flaga ta jest ustawiana, gdy wystąpi przechwycenie wyzwolone przez końcówkę ICP1. Jeżeli rejestr ICR1 służy jako wartość szczytowa TOP licznika (ustawiana przez WGM13:0), flaga ICF1 jest ustawiana gdy licznik osiągnie wartość TOP.
Jest automatycznie kasowana, gdy przerwanie zostaje wykonane. Może być też wyzerowana przez wpisanie jedynki.

- **bit 4 – OCF1A: flaga przerwania przy porównaniu A**

OCF1A ustawi się gdy wystąpi porównanie między licznikiem TC1 a OCR1A. FOC1A nie ustawia tej flagi mimo wystąpienia porównania. Jest zerowany sprzętowo gdy wykona się przerwanie. Może być wyzerowany przez wpisanie logicznej jedynki.

- **bit 3 – OCF1B: flaga przerwania przy porównaniu B**

OCF1B ustawi się gdy wystąpi porównanie między licznikiem TC1 a OCR1B. FOC1B nie ustawia tej flagi mimo wystąpienia porównania. Jest zerowany sprzętowo gdy wykona się przerwanie. Może być wyzerowany przez wpisanie logicznej jedynki.

- **bit 2 – TOV1: flaga przepełnienia licznika:**

ustawienie się tej flagi zależy od bitów WGM13:0. W trybach CTC i zwykłej pracy licznika, TOV1 jest ustawiana po przepełnieniu się licznika. Kasowany jest automatycznie gdy przerwanie jest wykonane.
TOV1 jest kasowane przez wpisanie logicznej jedynki.

8-bit licznik Timer/counter2 z PWM i asynchronicznym zegarem

Licznik TC2 – 8-bitowy licznik. Główne właściwości:

- pojedynczy licznik
- zerowanie przy porównaniu (auto-ładowanie)
- PWM z poprawną fazą
- Generator przebiegu prostokątnego
- 10-bitowy preskaler
- źródła przerwań od porównania i przepełnienia (OCF2 i TOV2)
- źródło sygnału zegarowego z zewnętrznego kwarcu zegarkowego 32,768 kHz.

Rejestry licznika TCNT2 i komparatora (OCR2) są 8-bitowe. Znaczniki przerwań są ustawiane w rejestrze TIRF. Wszystkie przerwania są niezależnie ustawiane w rejestrze TIMSK. Rejestry te są używane w pozostałych licznikach.

Licznik TC2 może być taktowany wewnętrznie, przez preskaler, lub asynchronicznie z kwarcu zewnętrznego dołączonego do wejść TOSC 1/2. Działanie asynchroniczne licznika jest sterowane rejestrem ASSR. Układ wyboru źródła zegara używa sygnału do zmniejszania lub zwiększania zawartości licznika. Licznik jest wyłączony, gdy nie jest wybrane żadne źródło.

Podwójnie buforowane wyjście rejestru komparatora (OCR2) jest stale porównywane z zawartością licznika. Wynik porównania może być użyte przez generator przebiegu do wytwarzania PWM lub sygnału o zmienianej częstotliwości na wyjściu porównania (OC2). Wynik porównania będzie również ustawiać Bit Porównania (OCF2) z którym może być użyty do wytwarzania.

Źródła zegara licznika TC0:

Licznik może być taktowany przez wewnętrzny, synchroniczny zegar lub przez zewnętrzny asynchroniczny sygnał. Normalnie źródłem sygnału jest zegar systemowy $clk_{I/O}$. Wpisanie do bitu AS2 w rejestrze ASSR logicznej jedynki sygnał zegarowy jest pobierany z zewnętrznego kwarcu dołączonego do końcówek TOSC1 i TOSC2.

Jednostka zliczająca:

Główna część licznika jest 8-bitową programowalną dwukierunkową jednostką.

W zależności od trybu pracy licznik jest zwiększany, zmniejszany lub zerowany w każdym cyklu zegara (clk_{T2}). clk_{T2} może być wytwarzane z wewnętrznego lub zewnętrznego źródła sygnału zegarowego, co jest wybierane przez bity CS22:0. Gdy nie jest wybrane źródło zegara (CS22:0 = 0) licznik stoi. Wartość TCNT2 może być dostępna przez CPU bez względu na to czy sygnał clk_{T2} występuje czy nie. CPU ma pierwszeństwo w kasowaniu liczników i operacjach na nich.

Sposób pracy licznika jest zależny od bitów WGM21 i WGM20 umieszczonych w rejestrze licznika TC2 (TCCR2).

Bit przepełnienia (TOV2) może być użyty do generowania przerwań.

Wyjściowy rejestr porównujący:

8-bitowy komparator ciągle porównuje zawartość licznika (TCNT2) z rejestrem OCR2. Osiągnięcie przez TCNT2 takiej samej wartości jak OCR2 jest przez komparator sygnalizowane. Spowoduje ono ustawienie bitu OCF2 w następnym cyklu zegara. Jeżeli bity OCIE2 i globalny system przerwań I w SREG są ustawione to jedynka w OCF wygeneruje przerwanie. OCF2 jest automatycznie kasowany w trakcie wykonywania przerwania. Może być również kasowany programowo przez wpisanie zera. Porównanie wykorzystywane jest do wytwarzania przebiegu prostokątnego, przy odpowiednim ustawieniu bitów WGM21:0 i COM21:0. Sygnały wartości szczytowej lub maksymalnej są używane do generowania przebiegów.

Rejestr OCR2 jest podwójnie buforowany gdy licznik pracuje w trybie PWM. Podczas normalnej pracy i auto-ładowania (CTC) podwójne buforowanie jest wyłączone. Buforowanie to synchronizuje wpis do rejestru OCR2 i zapobiega w związku z tym pojawieniu się niesymetrycznych sygnałów PWM.

Gdy podwójne buforowanie jest aktywne, CPU ma dostęp do OCR2 przez bufor, gdy jest wyłączone dostęp do tego rejestru jest bezpośredni.

FOC:

W trybach innych niż PWM wyjście porównania komparatora może być wymuszone przez wpis jedynki do bitu FOC2. Wymuszenie to nie ustawi bitu OCF2 ani nie spowoduje zerowania / auto-ładowania licznika, ale końcówka OC2 będzie uaktualniona gdy porównanie rzeczywiście wystąpi. (o ile COM21:0 zostaną odpowiednio skonfigurowane).

Wpisywanie przez CPU wartości do TCNT2 zablokowuje układ porównania do pojawienia się następnego cyklu zegara, nawet gdy licznik jest zatrzymany. Umożliwia to ustawienie rejestru o takiej samej wartości jak TCNT2 bez wyzwalania przerwania gdy licznik pracuje.

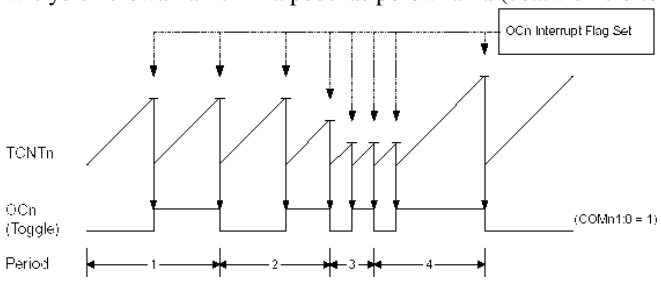
Używanie układu porównującego:
Blokowanie na jeden takt zegara układu porównującego podczas wpisu do TCNT2 może powodować pewne problemy gdy używa się wyjścia porównującego, niezależnie do tego czy licznik pracuje, czy nie. Jeśli wartość wpisana do TCNT2 jest równa wartości w OCR2 to porównanie nie zajdzie, co spowoduje nieprawidłowe wygenerowanie sygnału. Podobnie, nie wpisywanie liczb równych zeru do TCNT2 gdy licznik liczy w dół.
Ustawienie do pracy OC2 powinno zajść zanim ustawi się kierunek portu jako wyjście. Najprostszą metodą na ustawienie wartości OC2 jest użycie Wymuszenia Wyjścia Porównania (FOC2) w zwykłym trybie pracy. Rejestr OC2 przechowuje zawartość nawet podczas zmian między trybami generowania przebiegów.
Bity COM21:0 nie są podwójnie buforowane razem z wartością porównywaną. Zmiana ustawień bitów COM21:0 da natychmiastowy skutek.
Wyjściowa jednostka porównująca:
Bity wyjścia porównania (COM21:0) posiadają dwie funkcje. Generator przebiegu prostokątnego używa tych bitów do zdefiniowania stanu wyjścia porównania (OC2) przy następnym porównaniu. Stan OC2 dotyczy wewnętrznego rejestru, a nie końcówki.
Główne funkcje portu we/wy są ignorowane przez OC2, jeżeli któryś z bitów COM21:0 są ustawione. Jednak kierunek końcówki OC2 jest kontrolowane przez rejestr kierunku portu DDR. Kierunek portu dla końcówki OC2 (DDR_OC2) musi być ustawiona jako wyjście, zanim końcówka ta będzie wyjściem OC2.
Niektóre ustawienia bitów COM21:0 są zarezerwowane dla pewnych trybów.

Tryby pracy generatora przebiegów i układu porównującego:
Generator przebiegu używa bitów COM21:0 w trybach Normal, CTC i modulacjach PWM. We wszystkich trybach wpis COM21:0 = 0 powoduje że OC2 nie zadziała przy następnym porównaniu. Zmiana zawartości bitów COM21:0 da efekt przy pierwszym porównaniu po zmodyfikowaniu tych bitów. W trybach innych niż PWM, zadziałanie OC2 można wymusić natychmiastowo przez użycie bitu zbocza FOC2.

Tryby pracy:
Poszczególne tryby pracy licznika (np. generowanie przebiegu; PWM) uzyskiwane są przez ustawienie bitów Generatora Przebiegu (WGM21:0) i Wyjścia Porównania (COM21:0). Bity COM nie będą oddziaływać na sposób zliczania w trybie generowania przebiegu. Kontrolują czy wyjście PWM będzie odwrócone lub nie odwrócone.
Dla trybów innych niż PWM bity COM21:0 kontrolują czy wyjście jest ustawione, wyzerowane, albo zmienione podczas porównania.

Tryb normalny:
Jest to najprostszy tryb pracy (bity WGM21:0 = 0). W tym trybie licznik zawsze liczy w górę i jest zerowany po przepełnieniu (gdy zliczy do 0xFF). Bit przepełnienia jest ustawiany w tym samym cyklu zegara, gdy licznik zeruje się. Może być traktowany jako dziewiąty bit licznika, z tym że jest tylko ustawiany, nie kasowany. Jednak fakt automatycznego zerowania tego bitu w skutek przyjęcia przerwania może posłużyć do programowego zwiększenia rozdzielczości licznika. Nowa wartość licznika może być wpisywana w dowolnej chwili.
Wyjście porównania może być użyte do wyzwalania przerwania w ustalonym czasie. Użycie wyjścia porównania do generowania przebiegu w trybie Normal nie jest zalecane, ponieważ zajmuje to za dużo czasu procesorowi.

Zerowanie licznika w trybie porównania (CTC):
W trybie zerowania licznika podczas porównania (ustawienie bitów WGM21:0 = 2) jest używany rejestr OCR2.



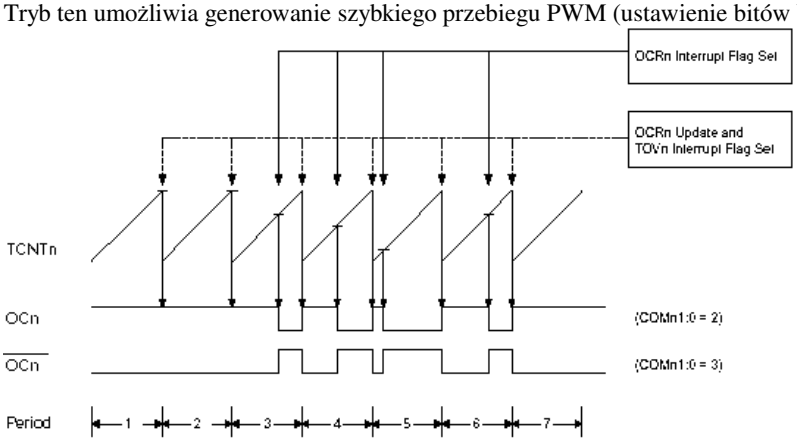
W trybie CTC licznik jest zerowany, gdy zliczy do takiej wartości jaka jest wpisana do rejestru OCR2. Rejestr ten definiuje największą wartość do której może doliczyć licznik i w ten sposób decyduje o jego rozdzielczości. Ten tryb jest najlepszą metodą na określenie częstotliwości sygnału wyjściowego.
Przerwanie może być generowane podczas przepełniania się licznika przez użycie bitu OCF2.

Gdy przerwania są aktywne, procedura obsługi przerwania może być wykorzystana do zmiany wartości maksymalnej, do której zlicza licznik.
Przerwanie może zmienić wartość szczytową dla licznika, jakkolwiek zmiana wartości szczytowej zbliżonej do zera, podczas gdy licznik pracuje z wyłączonym przeskalerem, lub ustawionym na niewielki podział, muszą być dokonywane ostrożnie, ponieważ w trybie CTC nie działa podwójne buforowanie. Jeżeli wpisana nowa wartość do OCR2 jest mniejsza od aktualnej wartości w TCNT2, licznik opóźni proces porównania. Licznik będzie musiał zliczyć do wartości maksymalnej (0xFF) i po wyzerowaniu zliczyć do wartości przy której wystąpi porównanie.
Generowanie przebiegu prostokątnego w trybie CTC na wyjściu OC2 uzyskuje się przez ustawienie bitów trybu Wyjścia Porównania (COM21:0 = 1). Wartość OC2 pojawi się na końcówce portu we/wy gdy w rejestrze kierunku portu końcówkę tą ustawi się jako wyjście.
Generowany przebieg osiągnie najwyższą częstotliwość równą $f_{OC2}=f_{CLK\_I/O}/2$, gdy do OCR2 wpisane jest zero. Częstotliwość przebiegu wyrażona jest wzorem:

$$f_{OCn} = \frac{f_{CLK\_I/O}}{2 \cdot N \cdot (1 + OCRn)}$$

gdzie N jest podziałem preskalera (przyjmuje wartości 1, 8, 64, 256 lub 1024).
Tak jak w trybie Normal, bit przepełnienia TOV2 jest ustawiany w tym samym cyklu zegara w którym następuje wyzerowanie licznika w skutek przepełnienia.

Tryb Fast PWM:
Tryb ten umożliwia generowanie szybkiego przebiegu PWM (ustawienie bitów WGM21:0 = 3).



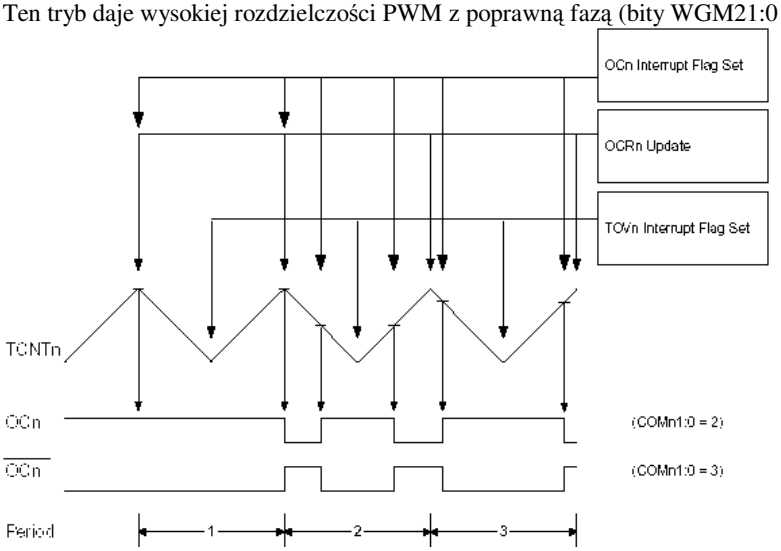
Tryb ten różni się od pozostałych operacją pojedynczego zliczania. Licznik zlicza od zera do maksymalnej (0xFF) i ponownie zaczyna od zera. W trybie bez odwrócenia wyjścia, wyjście porównania OC2 jest zerowane gdy wartość licznika jest większa lub równa wartości w komparatorze i ustawiane gdy licznik jest zerowany. W trybie z odwróconym wyjściem OC2 jest ustawiane podczas porównania i kasowane przy zerowaniu. Częstotliwość trybu Fast PWM może być dwa razy wyższa od trybu Correct Pwm używającego operacji podwójnego zliczania. Tryb ten jest odpowiedni do regulacji mocy, przetwarzania cyfrowo-analogowego. Wysoka częstotliwość PWM umożliwia zmniejszenie rozmiarów zewnętrznych elementów współpracujących – cewek czy kondensatorów i obniża koszty wykonania.
Licznik jest zwiększany aż osiągnie maksymalną wartość (0xFF), wtedy, w następnym cyklu zegara następuje jego wyzerowanie.
Bit przepełnienia (TOV2) jest ustawiany podczas osiągnięcia przez licznik wartości maksymalnej. Jeżeli przerwania są aktywne, to w procedurze przerwania można zmienić wartość rejestru porównania.
Przebieg generowany jest na końcówce OC2 gdy jest ona ustawiona jako wyjście w rejestrze kierunku portu (DDR). Ustawienie bitów COM21:0 = 2 (COM20 = 0; COM21=1) uaktywnia nie odwrócony przebieg PWM na końcówce OC2, wpisanie do COM21:0 wartości 3 powoduje odwrócenie przebiegu.
Przebieg prostokątny jest wytwarzany przez ustawienie (lub wyzerowanie) OC2 przy porównaniu między rejestrami OCR2 i TCNT2 i zerowane (ustawiane) przy zerowaniu licznika (przejściu z wartości maksymalnej 0xFF do 0x00).
Częstotliwość nośna obliczana jest ze wzoru:

$$f_{OCnPWM} = \frac{f_{CLK\_I/O}}{N \cdot 256}$$

gdzie N jest podziałem preskalera (przyjmuje wartości 1, 8, 32, 64, 128, 256 lub 1024).

Jeżeli do rejestru OCR2 jest wpisane zero, na wyjściu pojawiać się będzie krótka szpilka przy takcie zegara w którym licznik się przepełnia. Ustawienie OCR2 na maksimum (0xFF) ustawi wyjście w stanie niskim lub wysokim (zależnie od bitów COM21:0).
Przebieg z 50% wypełnieniem przy częstotliwości $f_{oc2}=f_{CLK\_I/O}/2$ uzyska się wpisując do OCR2 liczbę 0x00 (COM21:0 = 1). Różnicą do trybu CTC jest aktywne podwójne buforowanie).

PWM z poprawną fazą:
Ten tryb daje wysokiej rozdzielczości PWM z poprawną fazą (bity WGM21:0 ustawione na 1).



Zasada działania opiera się na podwójnym zliczaniu. Licznik powtarza zliczanie od wartości zerowej do maksymalnej i na odwrót. W trybie bez odwracania wyjść OC2 jest zerowane przy porównaniu pomiędzy TCNT2 a OCR2 podczas zliczania w górę i ustawiany podczas porównania przy zliczaniu w dół. W trybie z zanegowanym wyjściem sytuacja jest odwrotna. System podwójnego zliczania daje mniejszą częstotliwość niż we wcześniejszym trybie. Dzięki uzyskanej w ten sposób symetrii tryb ten jest zalecany do sterowania silnikami.
Rozdzielczość PWM z poprawną fazą jest stała i wynosi 8 bitów. Licznik jest zwiększany dopóki nie osiągnie maksimum – wtedy zmienia kierunek liczenia. Rejestr TCNT2 będzie miał wartość równą maksimum (0xFF) przez jeden cykl zegara taktującego licznik.
Bit przepełnienia (TOV2) jest ustawiany przy dojściu licznika do wartości BOTTOM. Flagę przerwania można użyć do generowania przerwania za każdym razem gdy licznik dotrze do wartości minimalnej (BOTTOM).
Komparator powoduje generowanie przebiegu na końcówce OC2. Ustawienie bitów COM21:0 na 2 uruchomi nie odwrócony PWM. Tryb z zanegowanym wyjściem dostępny jest przy COM21:0 ustawionych na 3. Bieżący stan OC2 będzie odzwierciedlony na końcówce OC2 gdy będzie ona ustawiona jako wyjście w rejestrze kierunku portu. Przebieg PWM jest uzyskiwany przez wyzerowanie (lub ustawienie) OC2 podczas porównania OCR2 i TCNT2 gdy licznik zlicza w górę i ustawiana (lub zerowana) przy porównaniu OCR2 i TCNT2 podczas pracy licznika w dół. Częstotliwość licznika określona jest zależnością:

$$f_{OCnPCPPWM} = \frac{f_{clk\_I/O}}{N \cdot 510}$$

gdzie N jest podziałem preskalera (przyjmuje wartości 1, 8, 32, 64, 128, 256 lub 1024).
Jeśli do rejestru OCR2 wpisana jest wartość minimalna (0x00), wyjście będzie ciągle w stanie niskim; jeśli będzie to wartość maksymalna (0xFF) to wyjście będzie w stanie wysokim (dla trybu bez odwracania wyjść – z inwersją odwrotnie).

Wyjście OC2n może zmienić się ze stanu wysokiego na niski chociaż nie zaszło porównanie. Miejsca tych zmian gwarantują symetrię. Istnieją dwa przypadki zmiany stanu bez zajścia porównania:

- zmienia się wartość OCR2A z maksymalnej; jeżeli wartością OCR2A jest 0xFF wartość OC2 jest taka sama jak efekt porównania przy liczniku doliczającym w dół.
- licznik zaczął zliczać od wartości wyższej niż w OCR2A i opóźnia się porównanie.

Opis rejestrów 8-bitowego licznika
Rejestr sterujący licznikiem TCCR2:

Bit	7	6	5	4	3	2	1	0	
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
Odczyt/zapis	Z	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – FOC2: Wymuszenie wyjścia porównania**

bit FOC2 jest aktywny tylko wtedy, gdy WGM ustawiają licznik w trybie innym niż PWM. Ze względu na zgodność z przyszłymi układami bit ten musi być zerowany gdy poprzez rejestr TCCR2 licznik jest skonfigurowany jako PWM. Wpisanie logicznej jedynki do FOC2 wymusza natychmiastowe porównanie.

Przejście stanu FOC2 nie generuje przerwania ani nie wyzeruje licznika w trybie CTC gdy OCR2 jest jako wartość maksymalna.

FOC2 zawsze odczytywane jest jako zero.

- **bity 6,3 – WGM21:0 – rodzaje generowanych przebiegów:**

bity te kontrolują sposób zliczania licznika: wartość maksymalną do której dąży licznik i rodzaj wytwarzanego przebiegu. Tryby pracy licznika: zwykły, zerowanie licznika podczas porównania i dwa rodzaje PWM (modulacji szerokości impulsu).

tryb	WGM21 (CTC2)	WGM20 (PWM2)	Rodzaj pracy licznika	Górna wartość licznika	Uaktualnienie OCR0	Ustawienie bitu TOV2
0	0	0	Zwykły licznik	0xFF	Natychmiast	MAX
1	0	1	PWM z poprawną fazą	0xFF	Wart. Szczyt.	BOTTOM
2	1	0	CTC	W rejestrze OCR0	Natychmiast	MAX
3	1	1	Szybki PWM	0xFF	Wart. Szczyt.	MAX

- **bity 5,6 – COM21:0 – rodzaj wyjścia trybu porównania**

określają zachowanie się wyjścia komparatora licznika (końcówka OC2). Jeśli jeden z bitów lub oba bity są ustawione to wyjście układu porównującego w liczniku zostaje dołączone do końcówki portu I/O. Rejestr kierunku portu (DDR) musi ustawić port jako wyjście. Gdy OC2 jest dołączone do końcówek portu, funkcje bitów COM21:0 zależą od ustawienia bitów WGM21:0

Tryb zwykły i porównania

COM21	COM20	Określenie
0	0	Zwykły tryb pracy portu, OC2 odłączone
0	1	Porównanie zmienia OC2
1	0	Porównanie zeruje OC2
1	1	Porównanie ustawia OC2

Tryb Fast PWM i porównania (1)

COM21	COM20	Określenie
0	0	Zwykły tryb pracy portu, OC2 odłączone
0	1	Zarezerwowane
1	0	Porównanie zeruje OC2, ustawia przy wartości szczytowej
1	1	Porównanie ustawia OC2, zeruje przy wartości szczytowej

Tryb PWM z poprawną fazą i porównania (1)

COM21	COM20	Określenie
0	0	Zwykły tryb pracy portu, OC2 odłączone
0	1	Zarezerwowane
1	0	Porównanie zeruje OC2 gdy licznik liczy w górę, ustawia gdy licznik liczy w dół
1	1	Porównanie ustawia OC2 gdy licznik liczy w górę, zeruje gdy licznik liczy w dół

(1) porównanie jest ignorowane, ale zerowanie lub ustawianie działa, gdy OCR2 jest takie samo jak wartość szczytowa licznika i COM21 jest w stanie 1.

- **bity 2:0 – CS02:0 – wybór sygnału zegarowego:**

te trzy bity decydują o źródle sygnału zegarowego dla licznika TC:

CS02	CS01	CS00	Określenie
0	0	0	Brak sygnału zegarowego – licznik jest zatrzymany
0	0	1	clk_{T2S} (pełna prędkość – bez preskalera)
0	1	0	clk_{T2S} / 8 (z preskalerem)
0	1	1	clk_{T2S} / 32 (z preskalerem)
1	0	0	clk_{T2S} / 64 (z preskalerem)
1	0	1	clk_{T2S} / 128 (z preskalerem)
1	1	0	clk_{T2S} / 256 (z preskalerem)
1	1	1	clk_{T2S} / 1024 (z preskalerem)

Rejestr licznika TCNT2:

Bit	7	6	5	4	3	2	1	0	
	TCNT2[7:0]								TCNT2
Odczyt/zapis	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr TCNT2 daje bezpośredni dostęp, poprzez operacje zapisu i odczytu, do stanu licznika (licznik jest 8-bitowy). Wpis do TCNT2 zmienia układ porównujący na następny cykl zegara.
Modyfikacja tego rejestru w trakcie pracy licznika grozi pominięciem chwili porównania między TCNT2 a OCR2.

Rejestr wyjściowy układu porównującego OCR2:

Bit	7	6	5	4	3	2	1	0	
	OCR2[7:0]								OCR2
Odczyt/zapis	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Wartość początkowa	0	0	0	0	0	0	0	0	

Zawiera 8-bitów wartości stale porównywanej z wartością licznika (TCNT2). Porównanie może być użyte do generowania przerwania, lub przebiegu na końcówce OC2.

<u>Praca asynchroniczna licznika TC2:</u>

Rejestr stanu ASSR:

Bit	7	6	5	4	3	2	1	0	
	-	-	-	-	AS2	TCN2UB	OCR2UB	TCR2UB	ASSR
Odczyt/zapis	O	O	O	O	Z/O	O	O	O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 3 – AS2: asynchroniczny licznik TC2**

wpisanie zera do AS2 powoduje, że licznik jest taktowany z zegara systemowego I/O, $clk_{I/O}$. Ustawienie jedynki powoduje taktowanie licznika z kwarcu dołączonego do wbudowanego oscylatora (końcówka TOSC1). Zmodyfikowanie bitu AS2 może zmienić zawartość rejestrów TCNT2, OCR2 i TCCR2.

- **bit 2 – TCN2UB: bit zajętości licznika TC2:**

Jeżeli licznik pracuje asynchronicznie a do rejestru TCNT2 jest zapisywana liczba, to bit ten jest ustawiany. Gdy licznik przyjmie nową zawartość poprzez rejestr pomocniczy, bit ten wyzeruje się sam. Zero w bicie oznacza, że licznik jest gotowy do modyfikacji

- **bit 1 – OCR2UB: bit zajętości układu porównującego:**

Jeżeli licznik pracuje asynchronicznie a do rejestru OCR2 jest zapisywana liczba, bit ten przyjmuje wartość jeden. Gdy licznik zaktualizuje nowe ustawienie przez rejestr pomocniczy, to bit wyzeruje się. Zero oznacza gotowość licznika na zmianę rejestru OCR2.

- **bit 0 – TCR2UB: bit zajętości rejestru sterującego licznikiem:**

Jeżeli licznik pracuje asynchronicznie a do rejestru TCCR2 jest zapisywana liczba, bit przyjmuje wartość 1. Gdy rejestr TCCR2 zostanie zaakceptowany przez licznik, bit jest automatycznie kasowany. Stan zera w bicie oznacza gotowość rejestru do wpisania nowej wartości.
Jeżeli którykolwiek z tych rejestrów będzie zmieniany, zanim odpowiedni bit gotowości zostanie wyzerowany, może spowodować to niezamierzone pojawienie się przerwania.
Sposób odczytu z rejestrów TCNT2, OCR2 i TCCR2 jest odmienny: odczyt z TCNT2 daje aktualną zawartość licznika; odczyt rejestrów OCR2 i TCCR2 następuje z pomocniczego bufora.

Praca asynchroniczna:

Przełączenie między trybami synchronicznym a asynchronicznym może spowodować niezamierzoną zmianę rejestrów TCNT2, OCR2, TCCR2.

Bezpieczna procedura do zmiany trybu pracy:

a. wyłączyć przerwania od licznika TC2 przez wyzerowanie bitów OCIE2 i TOIE2.
b. wybrać źródło zegara przez odpowiednie ustawienie bitu AS2.
c. Wpisać nowe liczby do rejestrów TCNT2, OCR2 i TCCR2.
d. odczekać aż bity TCN2UB, OCR2UB i TCR2UB przyjmą odpowiednią wartość aby można było przełączyć zegar licznika.
e. wykasować flagi przerwań licznika TC2
f. włączyć przerwania jeśli są konieczne.

- oscylator w liczniku jest zoptymalizowany do pracy z kwarcem zegarkowym 32,768 kHz. Dołączanie zewnętrznego sygnału zegarowego do końcówki TOSC1 może spowodować nieprawidłową pracę licznika TC2. Częstotliwość pracy CPU musi być ponad czterokrotnie większa od częstotliwości licznika.
- Wpis do któregoś z rejestrów TCNT2, OCR2 lub TCCR2 odbywa się przez bufor pomocniczy i przepisywany jest do licznika po dwóch impulsach na TOSC1. Użytkownik nie powinien wpisywać nowych wartości zanim zawartość rejestru bufora pomocniczego nie zostanie przepisana w miejsce docelowe. Te trzy rejestry posiadają własne bufory pomocnicze, więc wpis np. do TCNT2 nie naruszy przepisywania OCR2. do sprawdzania toku zapisu tych rejestrów służy rejestr ASSR.
- Gdy procesor wprowadzany jest w tryb oszczędzania energii lub w tryb uśpienia po zapisie do TCNT2, OCR2 lub TCCR2, użytkownik musi odczekać aż zapisane wartości zostaną przyjęte przez licznik, jeżeli służy on do wybudzania procesora. Inaczej procesor przejdzie w tryb uśpienia zanim nowe wpisy zostaną zaktualizowane. Jest to ważne gdy przerwanie od porównania (OC2) jest używane do wybudzania układu, odkąd funkcja wyjścia komparatora jest nieaktywna podczas zapisu do OCR2 lub TCNT2. Jeżeli cykle zapisu nie zakończą się a CPU przejdzie w stan uśpienia zanim bit OCR2UB wróci do stanu zera, układ nigdy nie odbierze przerwania i nie wybudzi się z powrotem.
- Jeżeli TC2 jest używany do budzenia układu z trybu Power-save lub Standby, musi być zachowana ostrożność jeżeli użytkownik chce ponownie wprowadzić jednostkę w jeden ze stanów: układ przerwań wymaga jednego cyklu TOSC1 do zresetowania się. Jeżeli czas między obudzeniem się a ponownym wprowadzeniem w tryb uśpienia trwa krócej niż jeden cykl (okres), przerwanie nie pojawi się i układ nie wybudzi się. Poprawny proces:
  1. wpisać wartość do TCCR2, TCNT2 lub OCR2.
  2. czekać aż odpowiedni bit zajętości w rejestrze ASSR wróci do stanu zero.
  3. wprowadzić układ w tryb Power-save lub Standby.
- Jeżeli wybrana jest praca asynchroniczna, oscylator 32,768 kHz działa ciągle za wyjątkiem stanu Power-save i Standby. Po resecie lub wybudzeniu oscylator może potrzebować nawet do jednej sekundy na ustabilizowanie się. Użytkownik powinien odczekać ponad sekundę zanim licznik TC2 zostanie użyty. Zawartość rejestrów licznika jest tracona w tym czasie.
- Opis obudzenia z trybu Power-save, gdy licznik jest taktowany asynchronicznie i gdy przerwanie jest zgłoszone, proces budzenia jest zapoczątkowany na danym cyklu timera tzn. timer jest zawsze wysunięty na przód (?) przez co najmniej jeden zanim procesor przeczyta wartość licznika. Po przebudzeniu MCU jest zatrzymany na 4 cykle, wykonuje przerwanie, wznawia wykonanie instrukcji następnej po Sleep.
- odczyt z TCNT2 zaraz po obudzeniu procesora da nieprawidłową wartość. Odkąd licznik jest taktowany asynchronicznie przez TOSC odczyt musi być wykonany poprzez rejestr synchronizujący z wewnętrznym zegarem systemowym $clk_{I/O}$ synchronizacja zachodzi w każdym narastającym zboczu TOSC1. Gdy układ wybudza się i zegar systemowy $clk_{I/O}$ jest ponownie aktywny, TCNT2 będzie odczytywane jako poprzednia wartość (zanim układ wszedł w tryb uśpienia) aż do narastającego impulsu na OSC1. Prawidłowy odczyt z licznika TCNT2:
  1. Wpis dowolnej liczby do któregoś z rejestrów OCR2 lub TCCR2.
  2. poczekać aż odpowiednie bity zajętości będą wyzerowane
  3. odczytać TCNT2.
- Podczas asynchronicznej pracy, synchronizacja Flagi przerwań pobiera 3 cykle + 1 cykl licznika. Procesor musi odczekać co najmniej jeden cykl zanim będzie mógł odczytać zawartość licznika. Wyjście końcówki porównania jest zmienione na cyklu zegarowym licznika i nie jest synchronizowane do zegara procesora.

Rejestr maskowania przerwań TIMSK:

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – OCIE2: uaktywnienie przerwania od układu porównującego:**

Ustawienie bitu OCIE2 i I w SREG w stan 1 powoduje uaktywnienie przerwania pochodzącego od układu porównującego licznika. Odpowiednie przerwanie jest wykonywane, gdy zajdzie porównanie w liczniku TC2 np. gdy ustawiony jest bit OCF2 w rejestrze flag przerwań licznika – TIFR.

- **bit 6 – TOIE2: uaktywnienie przerwania przy przepełnieniu**

Przerwanie jest aktywne gdy bit TOIE2 i I w SREG są ustawione.

Rejestr flag przerwań licznika TC2 TIFR:

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – OCF2: flaga przerwania przy porównaniu**

OCF2 ustawi się gdy wystąpi porównanie między licznikiem TC2 a OCR2. Jest zerowany sprzętowo gdy wykona się przerwanie. Może być wyzerowany przez wpisanie logicznej jedynki. Przerwanie zostanie wykonane, gdy I w SREG, OCIE2 i OCF2 są ustawione.

- **bit 6 – TOV2: flaga przepełnienia licznika:**

Przepełnienie się licznika ustawia bit TOV2, kasuje się gdy wykonuje się odpowiednie przerwanie. TOV2 jest kasowane przez wpisanie logicznej jedynki. Stan wysoki w TOIE2, I w SREG i TOV2 uruchamia obsługę przerwania. W trybie PWM z poprawną fazą bit ustawia się gdy licznik zmienia kierunek liczenia przy wartości $00.

<u>Preskaler licznika TC2:</u>

Źródło sygnału zegarowego dla licznika TC2 jest określany jako clk_{T2S}. Normalnie clk_{T2S} jest podłączone do głównego zegara systemowego. Przez ustawienie bitu AS2 w rejestrze ASSR, licznik TC2 jest asynchronicznie taktowany z końcówki TOSC1. Daje to możliwość użycia licznika TC2 jako zegara czasu rzeczywistego (RTC). Gdy AS2 jest ustawione, końcówki TOSC1 i TOSC2 są dołączone do portu C. Kwarc będzie włączony między TOSC1 a TOSC2. Wewnętrzny oscylator jest zoptymalizowany pod kątem użycia kwarcu zegarkowego o częstotliwości 32,768 kHz. Dołączanie zewnętrznego sygnału zegarowego do TOSC1 nie jest zalecane.

Możliwe jest wybranie preskalera częstotliwości o podziałach: clk_{T2S} / 8, clk_{T2S} / 32, clk_{T2S} / 64, clk_{T2S} / 128, clk_{T2S} / 256 i clk_{T2S} / 1024. Dodatkowo można wybrać clk_{T2S} jako zero (zatrzymanie). Ustawienie bitu PSR2 w rejestrze SFIOR powoduje wyzerowanie preskalera.

Rejestr specjalny SFIOR:

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	-	ACME	PUD	PSR2	PSR10	SFIOR
Odczyt/zapis	Z/O	Z/O	Z/O	O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 1 – PSR2 – reset preskalera licznika TC2:**

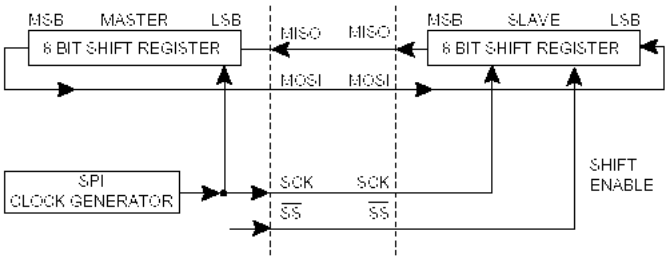
wyzerowanie preskalera uzyska się wpisując 1 do bitu PSR2. Bit wyzerowany zostanie sprzętowo po wykonaniu zerowania preskalera. Wpisanie zera programowo będzie nieskuteczne. Gdy bit jest odczytywany wynikiem zawsze będzie zero gdy licznik napędzany jest wewnętrznym przebiegiem. Jeśli ten bit jest zapisywany jeśli licznik działa w trybie asynchronicznym, bit ustawi się z powrotem, gdy preskaler zostanie wyzerowany.

Interfejs szeregowy SPI:

Szeregowy interfejs SPI pozwala na szybką synchroniczną transmisję danych między ATmega32 a urządzeniami zewnętrznymi lub pomiędzy układami ATmega32.

Cechy:

- Pełny-duplex, transmisja 3-przewodowa
- Działanie jako Master lub Slave
- Transmisja rozpoczynająca się od pierwszego lub ostatniego bitu w bajcie
- Flaga przerwania zakończenia transmisji
- Ochrona przez kolizją zapisu
- Funkcja wybudzania z Idle Mode
- Tryb podwójnej prędkości dla układu Master



Połączenie między układami.

System składa się z dwóch rejestrów przesuwających i generatora zegarowego układu Master. Układ Master rozpoczyna komunikację przez wymuszenie stanu niskiego na linię Slave Select SS. Master i Slave przygotowują dane do wysłania i master generuje sygnał zegarowy na linię SCK w celu wymiany danych. Dane są zawsze przesyłane z master do slave przez końcówkę Master Out – Slave In : MOSI, i z slave do master przez końcówkę Master In – Slave Out : MISO. Po wysłaniu danych slave jest synchronizowany przez master przez wystawienie stanu wysokiego na końcówce SS.

Jeśli układ jest skonfigurowany jako master, interfejs SPI nie będzie sterował automatycznie linią SS. Procedura ta musi być wykonana programowo przed rozpoczęciem transmisji. Wpisanie do rejestru danych SPDR powoduje uruchomienie generatora zegarowego SPI i przesuwanie wysyłanych bitów wysyłanych do slave. Po przesunięciu i przesłaniu jednego bajtu, zegar SPI zostaje zatrzymany i ustawia się flaga zakończenia transmisji (SPIF). Jeżeli przerwania SPI są włączone (SPIE w SPCR) włącza się obsługa przerwania. Master może kontynuować wysyłanie przez wpisanie kolejnego bajtu do rejestru SPDR, lub zasygnalizować koniec transmisji przez ustawienie SS w stan wysoki. Ostatni bajt przychodzący będzie zapamiętany w buforze w celu późniejszego użycia.

Jeśli układ jest ustawiony jako slave, interfejs SPI pozostaje w trybie uśpienia z linią MISO jako trójstanową tak długo jak linia SS jest w stanie wysokim. W tym stanie program może zmienić zawartość rejestru SPDR, ale dane nie będą wysłane w takt nadchodzących impulsów zegarowych na SCK dopóki na SS nie pojawi się stan niski. Po przesunięciu jednego bajtu będzie ustawiona flaga końca transmisji SPIF. Układ slave może umieścić nową daną do wysłania w SPDR zanim odczyta nadchodzącą daną. Ostatni nadchodzący bajt będzie zapamiętany w buforze w celu późniejszego użycia.

System jest pojedynczo buforowany w kierunku nadawania i podwójnie w kierunku odbierania. Oznacza to, że bajt, który ma być wysłany nie może być wpisany do rejestru SPDR zanim poprzedni proces przesuwania bitów nie zostanie zakończony. Przy odbiorze dana musi być odczytana z rejestru zanim następny bajt zostanie przysłany. Jeśli nie, poprzedni bajt będzie skasowany.

W trybie slave SPI, przysyłany sygnał zegarowy na linii SCK jest próbkowany, stąd częstotliwość SPI nie może przekraczać $f_{OSC}/4$.

Znaczenie końcówek SPI:

Końcówka	Kierunek, master SPI	Kierunek, slave SPI
MOSI	Zdefiniowane przez użytkownika	Wejście
MISO	Wejście	Zdefiniowane przez użytkownika
SCK	Zdefiniowane przez użytkownika	Wejście
SS	Zdefiniowane przez użytkownika	Wejście

Poniższy kod pokazuje jak zainicjalizować SPI jako master i jak przygotować prostą transmisję. W miejscu DDR_SPI w tym przykładzie należy umieścić właściwy rejestr DDR z SPI. DD_MOSI, DDMISO, DD_SCK muszą być zamienione z prawdziwymi końcówkami spełniającymi te funkcje np.: jeżeli MOSI jest na końcówce PB5, należy zamienić DD_MOSI na DDB5 i DDR_SPI na DDRB.

Przykład w asemblerze:

```
SPI_MasterInit:
        ; ustaw MOSI i SCK jako wyjścia, pozostałe jako wejścia
ldi r17,(1<<DD_MOSI)|(1<<DD_SCK)
out DDR_SPI,r17
        ; włącz SPI, ustaw na Master, częstotliwość fck/16
ldi r17,(1<<SPE)|(1<<MSTR)|(1<<SPR0)
out SPCR,r17
ret
SPI_MasterTransmit:
        ; rozpocznij transmisję danych (r16)
out SPDR,r16
Wait_Transmit:
        ; czekaj na zakończenie transmisji
sbis SPSR,SPIF
rjmp Wait_Transmit
ret
```

Przykład w C:

```
void SPI_MasterInit(void)
{
        /* ustaw MOSI i SCK jako wyjścia, pozostałe jako wejścia */
    DDR_SPI = (1<<DD_MOSI)|(1<<DD_SCK);
        /* włącz SPI, ustaw na Master, częstotliwość fck/16*/
    SPCR = (1<<SPE)|(1<<MSTR)|(1<<SPR0);
}
void SPI_MasterTransmit(char cData)
{
        /* rozpocznij transmisję danych */
    SPDR = cData;
        /* czekaj na zakończenie transmisji */
    while(!(SPSR & (1<<SPIF)))
    ;
}
```

Poniższy przykład pokazuje jak zainicjować SPI jako slave:

Przykład w asemblerze:

```
SPI_SlaveInit:
        ; ustaw MISO jako wyjście, resztę jako wejścia
ldi r17,(1<<DD_MISO)
out DDR_SPI,r17
        ; włącz SPI
ldi r17,(1<<SPE)
out SPCR,r17
ret
SPI_SlaveReceive:
        ; czekaj na zakończenie odbierania
sbis SPSR,SPIF
rjmp SPI_SlaveReceive
        ; odczytaj odebrane dane
in r16,SPDR
ret
```

Przykład w C:

```
void SPI_SlaveInit(void)
{
        /* ustaw MISO jako wyjście, resztę jako wejścia */
    DDR_SPI = (1<<DD_MISO);
        /* włącz SPI */
    SPCR = (1<<SPE);
}
char SPI_SlaveReceive(void)
{
        /* czekaj na zakończenie odbierania */
    while(!(SPSR & (1<<SPIF)))
    ;
        /* odczytaj odebrane dane */
    return SPDR;
}
```

Funkcje końcówki SS:

Tryb slave:

Gdy SPI jest ustawione jako slave, końcówka Slave Select (SS) jest zawsze wejściem. Gdy SS jest utrzymywana w stanie niskim, SPI jest uaktywniane, MISO staje się wyjściem jeśli zostanie skonfigurowane przez użytkownika. Pozostałe końcówki są wejściami. Gdy SS zmieni stan na wysoki, wszystkie końcówki stają się wejściami i SPI nie odbiera nadchodzących danych. Logika SPI będzie zresetowana gdy SS zmieni stan na wysoki.

Końcówka ta jest użyteczna do synchronizowania wysyłania danych w takt zegara master. Stan wysoki resetuje układ SPI nie pozwalając na wysłanie niekompletnych danych z rejestru przesuwającego.

Tryb master:

Użytkownik może wpływać na kierunek linii SS.

Jeżeli jest ustawiona jako wyjście to nie oddziałuje na system SPI.

Jeżeli będzie ustawiona na wejście, musi być utrzymana w stanie wysokim w celu zapewnienia pracy SPI. Jeżeli SS jest ściągnięty do zera przez zewnętrzny układ, system SPI zinterpretuje to tak jak by inny układ wybrał SPI jako slave i rozpoczął transmisję do niego. W celu uniknięcia połączenia szyny należy:

1. bit MSTR w SPCR jest ustawiony i SPI staje się układem slave. MOSI i SCK stają się wejściem
2. flaga SPIF w SPSR jest ustawiona i jeżeli przerwania od SPI są włączone i I w SREG ustawiony, procedura przerwania zostanie wykonana.

Gdy używane jest przerwanie od SPI w trybie master, procedura obsługi przerwania powinna zawsze sprawdzać czy bit MSTR jest nadal ustawiony. Jeśli MSTR zostanie wyzerowany przez wybranie trybu slave, musi być ustawiony przez użytkownika w celu przywrócenia trybu master.

Rejestr sterujący SPCR:

Bit	7	6	5	4	3	2	1	0	
	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPCR1	SPCR0	SPCR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – SPIE: włączenie przerwań od SPI:**

Przerwanie zostanie wykonane, gdy flaga SPIF w SPSR i I w SREG są ustawione (gdy SPIE jest w stanie „1”).

- **bit 6 – włączenie SPI:**

ustawienie jedynki włącza SPI. Musi być ustawiony w celu jakichkolwiek operacji na SPI.

- **bit 5 – DORD:**

DORD = 1 – transmisja zaczyna się od pierwszego bitu w bajcie

DORD = 0 - transmisja zaczyna się od ostatniego bitu w bajcie

- **bit 4 – MSTR: wybór master/slave:**

Ustawienie bitu oznacza, że SPI pracuje jako master, wyzerowanie – jako slave. Jeśli SS jest wejściem i jest wymuszony stan niski podczas gdy MSTR jest ustawione, MSTR będzie wyzerowane a SPIF w SPSR będzie ustawione. Użytkownik może ustawić MSTR w celu przywrócenia trybu master.

- **bit 3 – CPOL: biegunowość zegara:**

stan wysoki na SCK jest stanem nieaktywnym gdy CPOL = 1. Gdy CPOL = 0, stanem nieaktywnym na SCK jest stan niski.

- **bit 2 – CPHA: faza zegara:**

od ustawienia tego bitu zależy czy dana będzie odebrana przy zboczu pierwszym czy ostatnim sygnału zegarowego SCK.

- **bity 1,0 – SPR1, SPR0: podział częstotliwości zegara SPI**

bity te sterują podziałem częstotliwości w trybie master. SPR1 i SPR0 nie działają w trybie slave.

SPI2X	SPR1	SPR0	Częstotliwość SPI
0	0	0	$f_{osc}/4$
0	0	1	$f_{osc}/16$
0	1	0	$f_{osc}/64$
0	1	1	$f_{osc}/128$
1	0	0	$f_{osc}/2$
1	0	1	$f_{osc}/8$
1	1	0	$f_{osc}/32$
1	1	1	$f_{osc}/64$

Rejestr stanu SPI – SPSR:

Bit	7	6	5	4	3	2	1	0	
	SPIF	WCOL	-	-	-	-	-	-	SPI2X
Odczyt/zapis	Z/O	Z/O	O	O	O	O	O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – SPIF: SPI flaga przerwań:**

flaga SPIF jest ustawiana gdy transmisja jest zakończona. Przerwanie jest generowane, gdy SPIE w SPCR i I w SREG są ustawione. Będzie również ustawiona gdy SS jest wejściem i jest w stanie niskim gdy SPI pracuje jako master. SPIF jest kasowana sprzętowo gdy wykonana zostaje obsługa przerwania.

- **bit 6 – WCOL: flaga kolizji zapisu:**

WCOL zostanie ustawione gdy nastąpi wpis do rejestru danych SPI (SPDR) podczas wysyłania danych. Zostaje wyzerowana przez najbliższe odczytanie z rejestru danych.

- **bity 5...1 – zarezerwowane:**

odczytywane są zawsze jako zero.

- bit 0 – SPI2X – podwójna prędkość SPI:

Wpisanie jedynki podwaja częstotliwość SPI w trybie master. Wtedy minimalnym okresem są dwa cykle zegara CPU. Jeśli SPI pracuje jako slave gwarantowana jest praca tylko z częstotliwością $f_{osc}/4$ lub mniejszą.

Interfejs SPI w ATmega32 jest używane również do programowania EEPROM i Flash.

Rejestr danych SPI – SPDR:

Bit	7	6	5	4	3	2	1	0	
	MSB							LSB	SPDR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

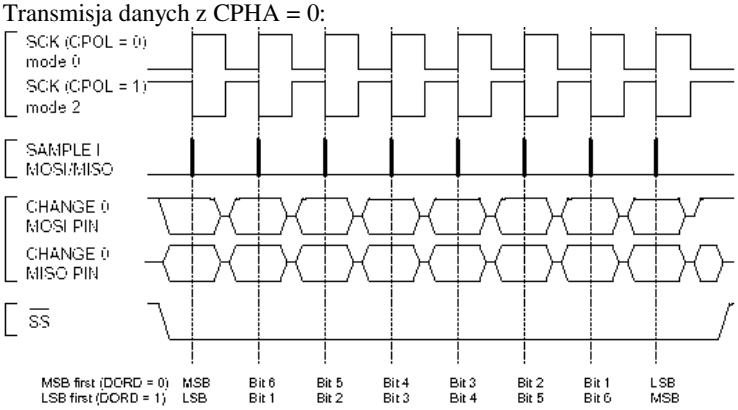
Jest rejestrem do zapisu/odczytu danych . Wpisanie do rejestru rozpoczyna transmisję. Odczyt powoduje przygotowanie bufora rejestru przesuwającego do odczytu.

Tryby danych:

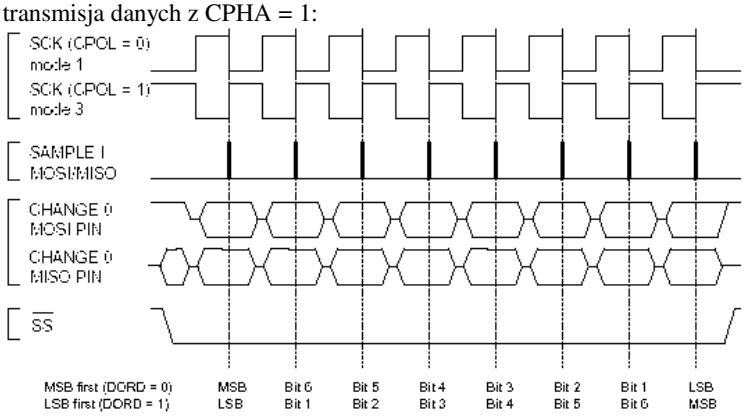
Są cztery tryby reakcji na fazę i biegunowość sygnału zegarowego ustalany przez bity CPHA i CPOL

	Zbocze pierwsze	Zbocze ostatnie	Tryb SPI
CPOL = 0 ,CPHA = 0	Pobieranie (narastanie)	Ustawianie (opadanie)	0
CPOL = 0 ,CPHA = 1	Ustawianie (narastanie)	Pobieranie (opadanie)	1
CPOL = 1 ,CPHA = 0	Pobieranie (opadanie)	Ustawianie (narastanie)	2
CPOL = 1 ,CPHA = 1	Ustawianie (opadanie)	Pobieranie (narastanie)	3

Transmisja danych z CPHA = 0:



transmisja danych z CPHA = 1:



Interfejs I^2C

Cechy:
- Łatwa do zastosowania komunikacja wymagająca tylko dwóch przewodów
- Tryb Master i Slave
- Możliwość pracy jako nadajnik i odbiornik
- 7-bit przestrzeń adresowa dająca 128 różnych adresów Slave
- Układ Multi-master
- Prędkość przesyłu do 400 kHz
- Mechanizm redukcji zakłóceń na linii
- Wybudzanie procesora po rozpoznaniu adresu Slave

Zasada działania:
TWI jest bardzo dobrym rozwiązaniem dla typowych aplikacji z udziałem mikrokontrolerów. Protokół TWI umożliwia połączenie między sobą do 128 różnych jednostek używając tylko dwóch przewodów, jednego dla sygnału zegarowego (SCL) i jednego dla danych (SDA). Zewnętrzny sprzęt wymaga tylko istnienia podciągania linii rezystorami (pull-up) dla obu przewodów. Wszystkie jednostki podłączone do linii mają własny adres i mechanizm przydzielania linii jest wbudowany w protokół TWI

Oznaczenia I^2C:

Zwrot	znaczenie
Master	Układ nadrzędny, kieruje transmisją, generuje sygnał zegarowy
Slave	Jednostka podrzędna, adresowana przez Master
Nadajnik	(Transmitter) Jednostka, która wysyła dane przez linię
Odbiornik	(Receiver) Jednostka, która odbiera dane z linii

Połączenia elektryczne:
Linie są dołączone do dodatniego napięcia zasilania przez rezystory. Układy sterujące linię TWI posiadają zawsze wyjścia typu otwarty kolektor lub otwarty dren. Realizowana jest przez to funkcja galwaniczne – AND. Niski poziom na linii jest wymuszany, gdy jeden lub więcej jednostek wystawia zero na linię. Wysoki poziom istnieje na linii gdy wszystkie układy TWI przełączą wyjścia w stan wysokiej impedancji, co powoduje że linia przechodzi w stan wysoki dzięki rezystorom podciągającym. Wszystkie układy AVR dołączone do linii TWI muszą być zasilane.
Ilość urządzeń, które mogą być dołączone do linii jest ograniczona tylko pojemnością do 400pF i 7-bitowym adresem. Możliwa jest praca przy 100 kHz i 400 kHz.

Sposób przesyłania danych

Przesyłanie bitów:
Każdy bit przesyłany przez linię TWI odbywa się w takt linii zegarowej. Poziom linii danych nie może się zmieniać gdy sygnał zegarowy jest w stanie wysokim. Wyjątkiem jest generowanie sekwencji start i stop.

Sygnał start i stop
Układ master rozpoczyna i zakańcza transmisję danych. Transmisja jest rozpoczynana gdy master wyśle w linię sekwencję startową i zakańczana po wysłaniu sekwencji stop. Pomiędzy START a STOP linia jest zajęta i żaden inny master nie może przejąć kontroli nad linią. Wyjątkowa sytuacja może wystąpić, gdy nowa sekwencja START zostanie wypuszczona pomiędzy sekwencją START – STOP. Taka powtórzona sekwencja startowa jest używana gdy układ master rozpoczyna nową transmisję bez zwalniania linii. Po powtórzonym starcie linia jest zajęta aż do następnej sekwencji STOP. Sekwencje te powstają przez zmianę poziomu linii danych SDA, gdy linia zegarowa SCL jest w stanie wysokim.

Adresowanie:
Każdy adres układów wysyłany przez TWI posiada 9-bitów, z czego 7-bitów adresowych, jeden bit określający zapis / odczyt i bit potwierdzenia. Jeżeli bit zapis / odczyt jest ustawiony, zostaje przygotowana operacja odczytu. Jeśli układ slave rozpozna adres startowy, powinien odpowiedzieć przez zmianę stanu SDA na niski w dziewiątym cyklu zegara CLK (ACK). Jeśli adresowany układ slave jest zajęty, lub z jakiegoś powodu nie może obsłużyć układu master, linia SDA powinna pozostać w stanie wysokim w cyklu potwierdzenia (ACK). Master powinien wysłać sekwencję Stop lub Powtórzony Start w celu rozpoczęcia nowej transmisji. Sekwencja adresująca zawiera adres slave i bit READ – odczytu lub WRITE – zapisu.
Najstarsza część adresu slave jest wysyłana jako pierwsza. Można wybrać dowolny adres slave, za wyjątkiem 0000 0000, który jest zarezerwowany na wywołania ogólne.

Na wysłanie wywołania ogólnego (0000 0000) wszystkie slave powinny odpowiedzieć przez wymuszenie stanu niskiego na SDA jako cykl potwierdzający w cyklu zegara ACK (9-bit). Wywołanie to jest używane, gdy master chce wysłać tą samą treść do wszystkich dołączonych układów slave. Jeżeli adres ogólny zostanie wysłany z bitem Write – zapisu, wszystkie slave podadzą stan wysoki na SDA w cyklu potwierdzenia ACK. Następne wysłane dane będą odebrane przez wszystkie te układy. /Wysłanie adresu ogólnego w bitem odczytu READ nie powinno mieć miejsca ze względu na to, że różne układy wysyłały by równocześnie różne dane.

Format sekwencji danych:

Dane wysyłane przez TWI mają 9-bitów długości, składają się na nie 8 bitów danych i bit potwierdzenia. Podczas wysyłania danych, układ master wytwarza sygnał zegarowy, sekwencje Start i Stop, podczas gdy slave jest odpowiedzialny za wysłanie bitu potwierdzenia. Potwierdzenie jest sygnalizowane przez odbiornik wymuszający stan niski na linii SDA w 9 cyklu SCL. Jeżeli odbiornik pozostawi SDA w stanie wysokim, zasygnalizuje koniec transmisji NACK. Gdy odbiornik otrzymuje ostatni bajt, lub nie może odebrać więcej bajtów, powinien wysłać właśnie sygnał NACK zaraz po ostatnim bajcie.

Mieszana sekwencja adresu i danych:

Podstawowa transmisja składa się z sekwencji Start, adresu slave i bitu R/W, jednego lub więcej bajtów i sekwencji Stop. Pusta transmisja składa się z Start i zaraz po niej Stop jest niedopuszczalna. Slave może przetrzymać dłużej stan niski na SCL, dzięki czemu spadnie częstotliwość zegara przy zbyt szybkich master, lub gdy wymagany jest dłuższy okres czasu między odbieranymi danymi. W ten sposób slave może zredukować prędkość transmisji Przez TWI.

System multimaster:

Protokół TWI pozwala na podłączenie kilku master do linii.

Układ TWI:

Końcówki SCL i SDA:

Wejścia posiadają zabezpieczenia usuwające impulsy krótsze niż 50ns. Wewnętrzne rezystory podciągające pull-up mogą być włączone przez odpowiednie ustawienie portu dla linii SCL i SDA, co pozwala w pewnych przypadkach na eliminację zewnętrznych rezystorów.

W trybie master kontrolowany jest okres trwania sygnału zegarowego SCL przez ustawienie rejestru TWRB i stopnia podziału preskalera w rejestrze TWSR. Praca w trybie slave nie jest zależna od ustawienia powyższych rejestrów, ale częstotliwość zegara CPU musi być przynajmniej 16 razy wyższa od częstotliwości SCL. Układ pracujący jako Slave może przedłużyć okres zegara SCL w celu zmniejszenia prędkości interfejsu TWI. Częstotliwość SCL jest generowana wg zależności:

$$\text{Czestotliwosc SCL} = \frac{\text{Czestotliwosc } CPU}{16 + 2(TWBR) \cdot 4^{TWPS}}$$

gdzie:

- TWBR – ustawienia bitów w rejestrze TWRB
- TWPS – ustawienia bitów preskalera

TWRB powinno być 10 lub więcej gdy układ pracuje jako master w przeciwnym przypadku generowane będą błędne przebiegi SCL i SDA.

Jednostka TWI posiada rejestr danych i adresu (TWDR), start/stop i wykrywanie arbitracji. TWDR zawiera adres lub daną do wysłania lub odebrania. Dodatkowo zawiera rejestr posiadający bit (N)ACK. Bit ten nie jest dostępny bezpośrednio drogą programową. Jeśli dane są odbierane bit ten może być ustawiony przez odpowiednie skonfigurowanie rejestru kontroli TWCR. W trybie nadawania stan bitu zależy od wartości w TWSR.

Kontroler start/stop jest odpowiedzialny z generowanie i wykrywanie sekwencji start i stop i powtórzony start. Kontroler może wykryć te sekwencje, nawet jeśli CPU jest w stanie uśpienia (powodując wtedy jego wybudzenie).

Adresowanie:

Układ adresowania sprawdza czy odebrany 7-bit adres jest taki jak w rejestrze TWAR. Jeśli bit TWGCE w rejestrze TWAR jest ustawiony, wszystkie przychodzące adresy będą porównywane z właściwym adresem wywołania. TWI może lub nie musi odpowiadać na adres zależnie od ustawień w TWCR.

Układ kontrolujący:

Generuje odpowiedź stosowną do ustawień w TWCR.

Tak długo jak flaga przerwania od TWI jest ustawiona, linia SCL jest w stanie niskim, dzięki czemu program może dokończyć pracę zanim transmisja będzie kontynuowana.

Flaga TWINT jest ustawiana:

- po wysłaniu sekwencji START/STOP/powtórzony START
- po wysłaniu adresu slave SLA + RW
- po utraceniu arbitracji
- gdy TWI będzie zaadresowane własnym adresem slave lub przez ogólne wywołanie
- po odebraniu bajtu danych
- gdy pojawi się błąd linii spowodowany niewłaściwą sekwencją start/stop.

Opis rejestrów TWI:

Rejestr TWBR:

Bit	7	6	5	4	3	2	1	0	
	TWBR7	TWBR6	TWBR5	TWBR4	TWBR3	TWBR2	TWBR1	TWBR0	TWBR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bity 7...0:**

określają podział generatora dla zegara SCL w trybie master. Wzór do obliczania częstotliwości jest powyżej.

Rejestr sterujący TWCR:

Bit	7	6	5	4	3	2	1	0	
	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE	TWCR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	O	Z/O	O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

Rejestr ten jest używany do kontroli pracy interfejsu TWI: włączania TWI, zainicjowania transmisji jako master przez wysłanie sekwencji start na linię, generowanie potwierdzenia przy odbiorze, sekwencji stop i kontroli wstrzymywania linii podczas wpisywania danej do rejestru TWRD wysłanej linią. Wskazuje również na kolizję gdy próbuje się wpisać daną do TWDR gdy jest on niedostępny.

- **bit 7 – TWINT: flaga przerwania od TWI:**

jest ustawiany sprzętowo, gdy TWI zakończy pracę. Gdy TWIE w rejestrze TWCR i I w SREG są ustawione, MCU wykona skok do procedury obsługi przerwania TWI. Ustawienie flagi powoduje przeciąganie stanu niskiego na linii SCL. Flaga ta musi być wyzerowana programowo przez wpisanie jedynki logicznej. Należy zwrócić uwagę, że TWINT nie jest automatycznie zerowana w momencie wejścia w obsługę przerwania. Wyzerowanie powoduje wznowienie transmisji, stąd dostęp do rejestrów TWAR, TWSR i TWDR jest możliwy tylko przed wyzerowaniem flagi.

- **bit 6 – TWEA: uaktywnienie potwierdzenia**

TWEA = 1 – wysyłany jest bit potwierdzenia (ACK) gdy:

1. odebrany będzie własny adres slave
2. odebrany będzie adres ogólny (0000 0000), w czasie gdy TWGCE w TWAR jest ustawione
3. dana będzie odebrana gdy układ pracuje jako odbiornik master, lub odbiornik slave

Przez wpisanie zera jednostka może być czasowo odłączona od linii TWI. Rozpoznawanie adresu można wtedy wznowić przez wpisanie jedynki do TWEA.

- **bit 5 – TWSTA: bit sekwencji startowej**

ustawiany gdy aplikacja chce użyć TWI jako master. Wtedy układ TWI sprawdza czy linia jest wolna i wysyła sekwencję startową. Jeśli linia jest zajęta, TWI czeka aż pojawi się sekwencja stop i dopiero wysyła sygnał start. TWSTA musi być wyzerowana programowo po wysłaniu startu.

- **bit 4 – TWSTO: bit sekwencji stop:**

Wpisanie jedynki w trybie master spowoduje wysłanie sekwencji stop i automatyczne wyzerowanie bitu TWSTO. W trybie slave bit ten nie generuje sygnału stop natomiast przełącza linie SCL i SDA w stan wysokiej impedancji.

- **bit 3 – TWWC: flaga kolizji zapisu**

jest ustawiany w czasie próby zapisu do TWDR gdy TWINT jest w stanie niskim. Flaga ta jest zerowana przez wpis do rejestru TWDR gdy TWINT jest w stanie wysokim.

- **bit 2 – TWEN: włączenie TWI**

włącza interfejs TWI gdy TWEN = 1.

- **bit 1 – zarezerwowany**

zawsze odczytywany jest jako zero

- **bit 0 – TWIE: uaktywnienie przerwania od TWI**

przerwanie TWI będzie tak długo aktywne jak TWINT będzie w stanie wysokim i gdy TWIE , I w SREG będą ustawione.

Rejestr stanu TWI – TWSR:

Bit	7	6	5	4	3	2	1	0	
	TWS7	TWS6	TWS5	TWS4	TWS3	-	TWPS1	TWPS0	TWSR
Odczyt/zapis	O	O	O	O	O	O	Z/O	Z/O	
Wartość początkowa	1	1	1	1	1	0	0	0	

- **bity 7...3 – TWS:**

określają stan interfejsu TWI. Kombinacje bitów opisane będą później. Dane odczytywane z rejestru to 5 bitów określających stan i dwa bity preskalera. Bity preskalera powinny być zamaskowane do zera jeżeli odczytywane są bity stanu. Dzięki temu uzyska się informacje tylko o interfejsie bez informacji o preskalerze.

- **bit 2- zarezerwowany**
- **bity 1...0 – TWPS: bity preskalera TWI**

możliwy jest zapis i odczyt z tych bitów

TWPS1	TWPS0	Podział
0	0	1
0	1	4
1	0	16
1	1	64

Rejestr danych – TWDR:

Bit	7	6	5	4	3	2	1	0	
	TWD7	TWD6	TWD5	TWD4	TWD3	TWD2	TWD1	TWD0	TWDR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

W trybie nadawania rejestr TWDR zawiera następną daną do wysłania, w trybie odbioru zawiera ostatnio odebrany bajt. Może być zapisywany gdy w TWI zostanie zakończony proces przesuwania bitów co jest sygnalizowane przez sprzętowe ustawienie flagi TWINT. Dane w TWDR są przechowywane tak długo jak TWINT jest w stanie wysokim, nie jest więc możliwa modyfikacja tego rejestru przed przyjściem przerwania. Gdy CPU zostaje wybudzony z trybów redukcji poboru mocy, wartość w TWDR jest nieokreślona. Obsługa bitu ACK jest automatycznie sterowana przez logikę TWI i nie może być bezpośrednio udostępniona CPU.

- **bity 7...0 – bity danych**

zawierają następną daną do wysłania lub ostatnio odebraną daną.

Rejestr adresu slave – TWAR

Bit	7	6	5	4	3	2	1	0	
	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE	TWAR
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	1	1	1	1	1	1	1	0	

Powinien zawierać 7-bitowy adres slave gdy układ pracuje jako odbiornik lub nadajnik slave (podrzędny).
Najmłodszy bit TWAR (TWGCE) jest używany do rozpoznawania ogólnego adresu wywołania ($00). Znajduje się tu układ porównujący adres slave (lub ogólny jeżeli jest włączony) odebrany przez interfejs. Jeżeli będzie się zgadzał zostanie wygenerowane przerwanie.

- **bity 7...1 – TWA: adres slave układu TWI**
- **bit 0 – TWGCE: bit uaktywnienia adresu wywołania ogólnego**

TWGCE = 1 – włączone
TWGCE = 0 – wyłączone

TWI jest bajtowo-zorientowanym i bazującym na przerwaniach systemem. Przerwania pojawiają się w różnych operacjach linii jak np.: podczas przyjmowania danych lub wysyłania sygnału START. Ponieważ TWI bazuje na przerwaniach program w CPU jest zwolniony z zajmowania się czynnościami związanymi z wysyłaniem danych. Bit uaktywnienia przerwania TWI - TWIE w TWCR razem z I w SREG pozwalają aplikacji na decyzję czy flaga TWINT powinna generować przerwanie.

algorytm transmisji:

1. TWI wysyła sekwencję startową. Uzyskuje się to przez odpowiednią wartość do TWCR. TWI nie rozpocznie żadnej operacji dopóki TWINT w TWCR jest ustawione. Natychmiast po wyzerowaniu przez aplikację bitu TWINT, TWI rozpocznie transmisję sekwencji startowej.
2. Gdy sygnał start został wysłany flaga TWINT w TWCR zostanie ustawiona a w TWSR pojawia się na nowo stan wskazujący czy sekwencja ta została wysłana prawidłowo.
3. Aplikacja powinna sprawdzić wartość TWSR w celu stwierdzenia czy "start" zostało wysłane poprawnie. Jeśli tak się nie stało to powinna uruchomić się procedura obsługi błędów. Następnie aplikacja wysyła adres slave + bit W do TWDR. TWDR jest używany do wysyłania albo adresu albo danej. Następnie odpowiednim skonfigurowaniem TWCR, TWI wysyła zawartość TWDR.
4. Po wysłaniu adresu ustawia się flaga TWINT w TWCR a w rejestrze stanu TWSR wpisywana jest nowa wartość wskazująca czy adres został poprawnie wysłany i czy slave odpowiedział bitem potwierdzenia czy nie.
5. aplikacja powinna sprawdzić wartość TWSR czy adres został wysłany poprawnie i czy potwierdzenie było oczekiwane. następnie program powinien załadować daną do TWDR, a następnie ustawić TWCR tak aby dana została wysłana. TWINT jest ustawiana przy wpisywaniu, wpis jedynki zeruje TWINT. TWI nie rozpocznie żadnej operacji dopóki TWINT w TWCR jest ustawione. Natychmiast po wyzerowaniu przez aplikację bitu TWINT, TWI rozpocznie transmisję danej.
6. Po wysłaniu danej flaga TWINT zostaje ustawiona a w TWSR pojawia się na nowo stan wskazujący czy dana została wysłana prawidłowo i czy slave odpowiedział bitem potwierdzenia czy nie.
7. aplikacja powinna sprawdzić wartość TWSR czy adres został wysłany poprawnie i czy potwierdzenie było
8. oczekiwane. następnie program powinien załadować daną do TWDR, a następnie ustawić TWCR tak aby sekwencja stop została wysłana. TWINT jest ustawiana przy wpisywaniu, wpis jedynki zeruje TWINT. TWI nie rozpocznie żadnej operacji dopóki TWINT w TWCR jest ustawione. Natychmiast po wyzerowaniu przez aplikację bitu TWINT, TWI rozpocznie transmisję "stop". Ważne: TWINT NIE jest ustawiane po wysłaniu "stop".

    - gdy TWI zakończy działanie i oczekuje na odpowiedź, TWINT jest ustawione, SCL jest ściągnięta do masy aż do TWINT równego zero.
    - gdy TWINT jest ustawione, użytkownik musi uzupełnić wszystkie rejestry TWI związane z następnym cyklem pracy TWI, np.: nową wartość do wysłania należy wpisać do TWDR.
    - po wykonaniu zadań przez aplikację następuje zapis do TWCR a TWINT powinno być ustawione.

przykłady w asemblerze:

	Przykład w asemblerze	Przykład w C	opis
1	**ldi** r16, (1<<TWINT) | (1<< TWSTA) | (1<<TWEN) **out** TWCR , r16	TWCR = (1<<TWINT) | (1<< TWSTA) | (1<<TWEN)	Wysyła sekwencję startową
2	wait1: **in** r16,TWCR **sbrs** r16,TWINT **rjmp** wait1	**while** (!(TWCR & (1<<TWINT))) ;	Czeka na ustawienie flagi TWINT wskazującej na wysłanie “start”
3	**in** r16,TWSR **andi** r16, 0xF8 **cpi** r16, START **brne** ERROR	**if** ((TWSR & 0xF8) != START) ERROR();	Sprawdza wartość rejestru stanu TWI, zasłania bity preskalera. Jeśli jest różnica w sekwencji startowej to zgłasza błąd
	ldi r16, SLA_W **out** TWDR, r16 **ldi** r16, (1<<TWINT) | (1<<TWEN) **out** TWCR, r16	TWDR = SLA_W; TWCR = (1<<TWINT) | (1<<TWEN);	Wpisuje SLA_W do rejestru TWDR, kasuje flagę TWINT aby wysłać adres
4	wait2: **in** r16,TWCR **sbrs** r16,TWINT **rjmp** wait2	**while** (!(TWCR & (1<<TWINT))) ;	Oczekuje na ustawienie się flagi TWINT. Pokazuje ona że adres został wysłany a ACK/NACK odebrany
5	**in** r16,TWSR **andi** r16, 0xF8 **cpi** r16, MT_SLA_ACK **brne** ERROR	**if** ((TWSR & 0xF8) != MT_SLA_ACK) ERROR();	Sprawdza wartość rejestru stanu TWI, zasłania bity preskalera. Jeśli jest różnica w MT_SLA_ACK to zgłasza błąd

	Przykład w asemblerze	Przykład w C	opis
	ldi r16, DATA **out** TWDR, r16 **ldi** r16, (1<<TWINT) | (1<<TWEN) **out** TWCR, r16	TWDR = DATA; TWCR = (1<<TWINT) | (1<<TWEN);	Wpisuje daną do TWDR, zeruje TWINT w celu rozpoczęcia transmisji danej
6	wait3: **in** r16,TWCR **sbrs** r16,TWINT **rjmp** wait3	**while** (!(TWCR & (1<<TWINT))) ;	Oczekuje na ustawienie się flagi TWINT. Pokazuje ona że dana została wysłana a ACK/NACK odebrany
7	**in** r16,TWSR **andi** r16, 0xF8 **cpi** r16, MT_DATA_ACK **brne** ERROR	**if** ((TWSR & 0xF8) != MT_DATA_ACK) ERROR();	Sprawdza wartość rejestru stanu TWI, zasłania bity preskalera. Zgłasza ewentualny błąd
	ldi r16, (1<<TWINT)|(1<<TWEN)| (1<<TWSTO) **out** TWCR, r16	TWCR = (1<<TWINT)|(1<<TWEN)| (1<<TWSTO);	Wysyła sekwencję "stop"

Tryby transmisji:

TWI może pracować w czterech trybach: master nadajnik, master odbiornik, slave nadajnik, slave odbiornik.
Tryb master nadajnik może służyć do zapisu szeregowej pamięci EEPROM z interfejsem I^2C, master odbiornik – do odczytu danych z tej pamięci.
Sekwencja startowa zostanie wysłana dzięki poniższym ustawieniom TWCR:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
wartość	1	x	1	0	x	1	0	x

TWEN musi być ustawiony aby włączyć TWI, również TWSTA musi być ustawione w celu wysłania sekwencji startowej. Flagę TWINT należy ustawić, żeby ją skasować. TWI będzie testować linię i wygeneruje sekwencję startową jak tylko linie interfejsu będą zwolnione. Po starcie rozpocznie się transmisja, flaga TWINT sprzętowo zostanie ustawiona, a w rejestrze stanu TWI – TWSR pojawi się wartość $08. W trybie master-transmitt należy wysłać adres slave + bit zapisu W (SLA+W). Osiągnie się to przez wpisanie adresu slave SLA+W do TWDR. Wtedy TWINT powinno się wyzerować (przez wpisanie jedynki) w celu kontynuowania transmisji.
Do TWCR należy więc wpisać:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
wartość	1	x	0	0	x	1	0	x

Gdy SLA+W zostało wysłane a bit potwierdzenia został odebrany, TWINT jest ponownie ustawiany, a wartość w rejestrze TWSR może wynosić $18, $20 lub $38.
Po udanej transmisji adresu SLA+W powinny być wysłane dane. Osiągnie się to przez wpisanie danej do TWDR. Rejestr ten może być tylko zapisywany, gdy TWINT jest w stanie wysokim. Jeśli nie to zostanie ustawiony bit kolizji zapisu TWWC w rejestrze TWCR. Po zaktualizowaniu TWDR należy wyzerować TWINT (przez wpis „1") aby kontynuować transmisję.

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
wartość	1	x	0	0	x	1	0	x

Powyższy opis będzie powtarzany dopóki ostatni bajt będzie wysyłany a transmisja zostanie zakończona przez sekwencję STOP lub powtórzony START. STOP jest generowane przez wpisanie wartości do TWCR:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
wartość	1	x	0	1	x	1	0	x

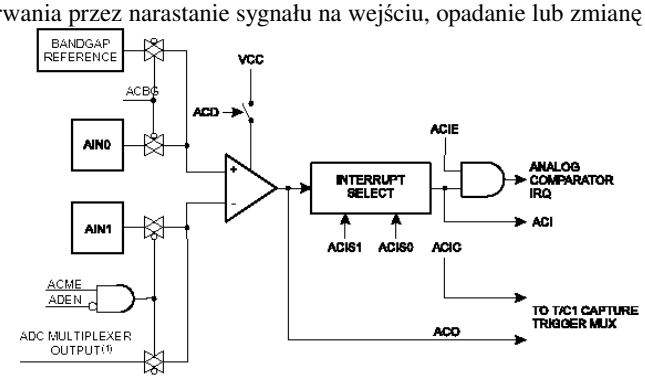
Powtórzony START:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
wartość	1	x	1	0	x	1	0	x

Po sekwencji powtórzonego startu (stan $10) TWI może połączyć się z tym samym układem Slave ponownie lub z nowym układem (gdy wysłano STOP). Powtórzony START umożliwia układowi master przełączenie między trybami – nadajnik - slave/master i - odbiornik – master bez utraty kontroli nad linią.

Komparator analogowy:

Układ ten porównuje wartość z wejść dodatniego -nie odwracającego (AIN0) i ujemnego –odwracającego (AIN1). Wyjście komparatora (ACO) przyjmie stan wysoki, gdy poziom napięcia na wejściu nieodwracającym (+) jest wyższe niż napięcie na wejściu odwracającym (-). Wyjście komparatora może służyć do wyzwalania wejścia przechwytywania Licznika TC1. Dodatkowo może generować niezależne przerwanie. Możliwy jest wybór wyzwalania przerwania przez narastanie sygnału na wejściu, opadanie lub zmianę na przeciwny.



Rejestr specjalny I/O SFIOR:

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	-	ACME	PUD	PSR2	PSR10	SFIOR
Odczyt/zapis	Z/O	Z/O	Z/O	O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 3 – ACME: uaktywnienie multipleksera komparatora analogowego**

Gdy bit jest ustawiony i przetwornik A/C (ADC) jest wyłączony, (ADEN w ADCSRA jest w stanie zero), multiplekser jest dołączony do wejścia odwracającego (-) komparatora analogowego; gdy ten bit jest wyzerowany to końcówka AIN1 jest dołączony do tego wejścia (-).

Rejestr stanu i kontroli komparatora – ACSR

Bit	7	6	5	4	3	2	1	0	
	ADC	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	ACSR
Odczyt/zapis	Z/O	Z/O	O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	Nie ustalona	0	0	0	0	0	

- **bit 7 – ACD: wyłączanie komparatora**

Jedynka logiczna w tym bicie wyłącza komparator. Może być ustawiony w dowolnej chwili. Pozwala to na redukcję poboru mocy w trybie pracy CPU i w trybie Idle. Gdy dokonuje się zmiany ustawienia bitu ACD, przerwanie od komparatora musi być wyłączone, przez wyzerowanie bitu ACIE w rejestrze ACSR. Inaczej przerwanie może zostać wyzwolone w trakcie wpisu do tego bitu.

- **bit 6 – ACBG: źródło napięcia odniesienia**

Jedynka logiczna w bicie dołącza do wejścia nieodwracającego (+) wewnętrzne źródło napięcia odniesienia, zero logiczne – wejściem nieodwracającym jest końcówka AIN0.

- **bit 5 – ACO: Wyjście komparatora**

Wyjście komparatora jest synchronizowane i dołączone bezpośrednio do ACO. Synchronizacja przebiega w 1-2 cyklach CPU.

- **bit 4 – ACI: znacznik przerwania od komparatora**

Bit ustawiany sprzętowo. Reakcja na stan komparatora ustawiana jest przez bity ACIS1 i ACIS0. Przerwanie od komparatora zostanie wykonane, gdy bity ACIE i I w rejestrze SREG zostaną ustawione na jeden. ACI będzie skasowane sprzętowo po wykonaniu programu obsługi przerwania. Zero może być wpisane programowo.

- **bit 3 – ACIE: uaktywnienie przerwań od komparatora.**

Przerwanie od komparatora jest aktywne gdy bit ten i bit I w SREG jest ustawiony na jeden.
Zero logiczne w ACIE wyłącza przerwanie od komparatora.

- **bit 2 – ACIC: uaktywnienie funkcji przechwytywania**

logiczna jedynka w tym bicie powoduje uaktywnienie wejścia przechwytywania w liczniku Timer/Counter1 przez komparator. Wyjście komparatora jest dołączone bezpośrednio do wejścia przechwytującego licznika, dzięki czemu redukuje się szumy i wybiera się sposób przerwania od przechwycenia w liczniku TC1. Po wpisaniu zera nie ma połączenia między komparatorem a wejściem przechwycającym. Żeby komparator mógł wyzwolić przerwanie od przechwycenia w liczniku TC1, musi być ustawiony bit TICIE w rejestrze TIMSK.

- **bit 1, 0 – ACIS1, ACIS0: wybór rodzaju przerwania od komparatora**

bity te określają jaki stan na wyjściu komparatora wyzwoli przerwanie.

ACIS1	ACIS0	Rodzaj przerwania
0	0	Przerwanie przy zmianie stanu wyjścia
0	1	Zarezerwowane
1	0	Przerwanie przy zboczu opadającym
1	1	Przerwanie przy zboczu narastającym (na wyjściu)

Dokonywanie zmian w tych bitach może się odbywać tylko przy wyłączonym przerwaniu od komparatora. (ACIE = 0 w rejestrze ACSR). W innym przypadku przerwanie może się pojawić w trakcie wpisu do tych bitów.

Multiplekser wejściowy komparatora:

Dowolne z wejść ADC0...7 może być dołączone do wejścia odwracającego (-) komparatora przez użycie multipleksera, jednak wtedy musi być wyłączony przetwornik analogowo – cyfrowy (ADC). Gdy bit ACME w rejestrze SFIOR jest ustawiony i przetwornik ADC wyłączony (przez wpisanie zera do bitu ADEN w rejestrze ADCSRA), bity MUX2...0 w ADMUX wybierają wejście odwracające (-) komparatora. Gdy ACME jest wyzerowany lub ADEN jest ustawiony to wejściem odwracającym jest końcówka AIN1.

Ustawienia multipleksera:

ACME	ADEN	MUX2...0	Wejście (-) komparatora
0	x	x x x	AIN1
1	1	x x x	AIN1
1	0	0 0 0	ADC0
1	0	0 0 1	ADC1
1	0	0 1 0	ADC2
1	0	0 1 1	ADC3
1	0	1 0 0	ADC4
1	0	1 0 1	ADC5
1	0	1 1 0	ADC6
1	0	1 1 1	ADC7

Przetwornik analogowo-cyfrowy

Parametry:

- rozdzielczość 10-bit
- nieliniowość 0,5 LSB
- czas konwersji 65-260us
- do 15kSPS przy najwyższej rozdzielczości
- 8 kanałów wejściowych
- 7 wejść różnicowych
- 2 wejścia różnicowe ze wzmocnieniem 10x i 200x(1)
- 0-Vcc zakres napięć wejściowych
- 2,56V wewnętrzne źródło napięcia odniesienia
- praca ciągła lub pojedyncza konwersja
- przerwanie po przetworzeniu
- tryb uśpienia z redukcją szumów

(1) – ta cecha jest możliwa do pracy tylko w układach w obudowie TQFP i MLF

10-bit przetwornik działający na zasadzie sukcesywnej aproksymacji jest połączony z 8-kanałowym analogowym multiplekserem wejściowym dołączonym do portu PORT A.
Przetwornik posiada również 16 różnicowych wejść. Dwa z tych wejść (ADC1, ADC0 i ADC3, ADC2) posiadają programowane wzmocnienie z krokiem 0dB (1x), 20dB(10x), lub 46dB(200x). 7 różnicowych analogowych wejść. Jako wejście ujemne może być wykorzystane 7 wejść, reszta jako dodatnie.
Ustawienie wzmocnienia na 1x lub 10x zmniejsza rozdzielczość do 8-bitów, przy 200x do 7-bitów.

Przetwornik posiada niezależne napięcie zasilające AVCC. Napięcie to nie może się różnić więcej niż $\pm$0,3V od Vcc. Wewnętrzne źródło odniesienia wynosi 2,56V lub Vcc. Może być również dostarczone z zewnątrz do końcówki AREF przez kondensator dla lepszej filtracji szumów.

Działanie:

Przetwornik dokonuje konwersji wejściowego napięcia analogowego na 10 bitową wartość, metodą sukcesywnej aproksymacji (kolejnego przybliżenia). Posiada wejściowy układ próbkująco - pamiętający. Minimalna wartość odpowiada GND (0V) a maksymalna AREF minus 1LSB. (najmłodszy bit). Dodatkowo AVCC lub wewnętrzne 2,56V może być dołączone do wejścia AREF przez ustawienie bitu REFSn w rejestrze ADMUX. Do wewnętrznego napięcie odniesienia dołącza się kondensator w celu redukcji szumów.
Wejścia analogowe i różnicowe są wybierane przez bity MUX rejestru ADMUX. Każde z wejść różnicowych może być wybrane jako wejście ujemne lub dodatnie wewnętrznego wzmacniacza.
Wewnętrzny wzmacniacz jest pominięty w przypadku wybrania pojedynczych wejść.
Przetwornik ADC uaktywnia się przez ustawienie bitu ADEN w rejestrze ADCSRA. Przed ustawieniem tego bitu źródło napięcia odniesienia i wybranie kanałów wejściowych nie będzie aktywne. ADC nie pobiera prądu, jeżeli jest nieaktywny (ADEN=0), stąd zaleca się jego wyłączenie przed wprowadzeniem procesora w tryb uśpienia.
Wynik konwersji przedstawiony jest w postaci 10 bitowej w rejestrach ADCH i ADCL. Normalnie wynik jest umieszczony w rejestrach od prawej strony, ale ustawienie bitu ADLAR w rejestrze ADMUX przesuwa wynik na lewą stronę.
Jeśli wynik jest dostosowany od lewej strony i nie jest wymagana rozdzielczość ponad 8-bit wystarczy odczytać wynik z rejestru ADCH. Inaczej rejestr ADCL musi być odczytany jako pierwszy, następny ADCH. Jedno odczytanie ADCL blokuje dostęp ADC do rejestrów danych. To oznacza, że gdy ADCL i następnie ADCH jest czytane zanim konwersja zostanie dokonana, żaden z rejestrów nie jest uaktualniony i rezultat konwersji jest nieprawidłowy. Gdy ADCH jest odczytywane dostęp ADC do rejestrów ADCL i ADCH jest uaktywniony.
Przetwornik ADC posiada własne przerwanie które może być wyzwolone po dokonaniu konwersji. Przerwanie będzie wyzwalane nawet gdy nieprawidłowy odczyt z rejestrów ADCL i ADCH odbędzie się zanim przetwornik dokona konwersji.

Uruchomienie konwersji:

Pojedyncza konwersja rozpoczyna się przez ustawienie bitu startu konwersji ADSC. Bit ten jest w stanie wysokim tak długo jak przetwornik przetwarza i kasowany jest sprzętowo po zakończonej konwersji. Jeżeli wyjścia ustawi się jako różnicowe w trakcie pracy przetwornika, to wynik będzie dotyczył stanu wejść przed tą zmianą.
Przetwarzanie może być wyzwalane automatycznie. Osiąga się to przez ustawienie bitu ADATE w rejestrze ADCSRA. Wybór źródła konwersji następuje po ustawieniu bitu ADTS w rejestrze SFIOR. Gdy pojawi się dodatni impuls na wybranym sygnale wyzwalającym, preskaler ADC jest zerowany i rozpoczyna się konwersja. Jest to przetwarzanie ze stałym odstępem (czasu). Jeżeli sygnał wyzwalający jest wciąż w stanie wysokim, przetwornik po dokonaniu konwersji nie zacznie przetwarzać ponownie. Jeśli dodatni (narastający) impuls

pojawi się na sygnale wyzwalającym podczas konwersji impuls ten zostanie zignorowany. Flaga przerwania zostaje ustawiona nawet gdy przerwania dla przetwornika są nie aktywne i globalny system przerwań jest wyłączony. Konwersja może zachodzić bez inicjacji przerwania. Flaga przerwania musi być wyzerowana do wyzwolenia nowej konwersji przy następnym przerwaniu.
Wybranie Flagi przerwań przetwornika ADC jako źródła wyzwalania umożliwia rozpoczęcie nowej konwersji po zakończeniu bieżącej. ADC pracuje wtedy w trybie Free Running, stale uaktualniając rejestry z wynikiem. Pierwsza konwersja musi być zapoczątkowana przez wpisanie jedynki do bitu ADSC w rejestrze ADCSRA. W tym trybie ADC dostarcza wyników przetwarzania nie zależnie od ustawienia bitu przerwań ADIF.
Rozpoczęcie pracy przetwornika w trybie automatycznego wyzwalania zostanie osiągnięte przez ustawienie bitu ADSC w rejestrze ADCSRA. Bit ten może być również wykorzystany do determinowania pracy przetwornika.
Układ sukcesywnej aproksymacji w pełnej rozdzielczości, wymaga taktowania sygnałem o częstotliwości między 50kHz a 200kHz. Przy rozdzielczości poniżej 10-bit częstotliwość może wynosić ponad 200kHz.
ADC zawiera preskaler generujący sygnał taktujący o częstotliwości akceptowanej przez przetwornik dla każdej częstotliwości systemowej procesora wyższej od 100kHz . jest on ustawiany przez bity ADPS rejestru ADCSRA. Preskaler zaczyna pracę w momencie uaktywnienia przetwornika ADC przez wpisanie jedynki do ADEN w rejestrze ADCSRA. Preskaler działa tak długo, jak bit ADEN jest ustawiony i ciągle zerowany gdy ADEN jest w stanie niskim.
Po ustawieniu bitu ADSC w ADCSRA pojedyncza konwersja rozpoczyna się w narastającym zboczu cyklu zegarowego przetwornika ADC.
Czas normalnego przetwarzania wynosi 13 cykli przetwornika ADC. Pierwsza konwersja po uruchomieniu przetwornika (ADEN=1 w ADCSRA) trwa 25 cykli.
Układ próbkująco - pamiętający S&H potrzebuje 13,5 cykli po rozpoczęciu przetwarzania w normalnej konwersji i 13,5 cykli ADC po starcie pierwszej konwersji. Gdy przetwarzanie jest zakończone wynik jest wpisywany do rejestrów danych ADC a bit ADIF jest ustawiany. W trybie pojedynczego przetwarzania ADSC zostaje jednocześnie wyzerowany. ADSC może być programowo ponownie ustawiony, co rozpocznie ponownie konwersję w narastającym zboczu impulsu zegarowego ADC.
W trybie auto – wyzwalania preskaler jest zerowany automatycznie gdy pojawia się sygnał wyzwalający przetwornik. Próbkowanie i pamiętanie pobiera 2 cykle po narastającym zboczu sygnału wyzwalającego. Dodatkowe 3 cykle systemowe CPU są przeznaczone dla synchronizacji. Tryb auto – wyzwalania potrzebuje 25 cykli na każde przetworzenie, ponieważ ADC musi być wyłączony i włączony ponownie po przetworzeniu.
W trybie Free Running nowa konwersja zostanie rozpoczęta natychmiast po zakończeniu poprzedniej podczas gdy ADSC pozostaje w stanie wysokim.

Czas konwersji:

sytuacja	Liczba cykli dla S&H	Czas przetwarzania
Pierwsza konwersja	14,5	25
Normalna konwersja, pojedynczy pomiar	1,5	13
Auto – wyzwalanie	2	13,5
Normalna konwersja, wejścia różnicowe	1,5/2,5	13/14

Wejścia różnicowe:
Różnicowa konwersja jest zsynchronizowana z wewnętrznym zegarem CK_{ADC2} równy połowie zegara przetwornika ADC. Synchronizacja jest dokonywana automatycznie. Konwersja inicjowana jest przez użytkownika (np. przy pojedynczym przetwarzaniu i przy pierwszym uruchomieniu w trybie Free Running)podczas gdy CK_{ADC2} jest w stanie niskim, będzie zajmowała taką samą ilość czasu jak pojedynczo - zakończona konwersja. (13 cykli ADC). Konwersja zainicjowana przez użytkownika gdy CK_{ADC2} jest w stanie wysokim będzie zajmować 14 cykli zegara ADC. W trybie Free Running ponowne przetwarzanie jest zainicjowane natychmiast po poprzednim i odkąd CK_{ADC2} jest w tym czasie w stanie wysokim konwersja ta zajmuje 14 cykli ADC.
Wzmocnienie jest optymalne w paśmie do 4kHz dla wszystkich wartości. Przy wyższych częstotliwościach wzmocnienie będzie nieliniowe. Gdy mierzony sygnał posiada wyższe harmoniczne powinno się stosować zewnętrzne filtry dolnoprzepustowe. Częstotliwość taktowania przetwornika jest niezależna od poziomu wzmocnienia. Np. czas przetwarzania ADC może wynosić 6us i prędkość przetwarzania osiągnie 12 kSPS bez względu na pasmo przenoszenia dla tego kanału.
Jeżeli używane są kanały różnicowe i przetwornik uruchomiony jest przez auto – wyzwalanie, musi być on wyłączony przed konwersją. Preskaler zeruje się wtedy przed rozpoczęciem przetwarzania.

Zmiana kanału lub wybór źródła odniesienia:
Bity MUXn i REFS1:0 w rejestrze ADMUX są pojedynczo buforowane przez rejestr pomocniczy, do którego CPU ma dostęp swobodny. Ich stan jest uaktualniany podczas rozpoczynania przetwarzania. Wybór kanału i odniesienia jest zablokowane podczas przetwarzania dla zapewnienia dostatecznego czasu do pomiaru. Przetwarzanie rozpoczyna się w narastającym cyklu zegara ADC po wpisie do ADSC.
Jeśli ADATE i ADEN są ustawiane, przerwanie może wystąpić w dowolnej chwili. Jeśli w tym czasie rejestr ADMUX jest zmieniany, nie da się przewidzieć czy konwersja jest dokonana na starych czy nowych ustawieniach.
ADMUX może być bezpiecznie modyfikowany zgodnie z procesem:

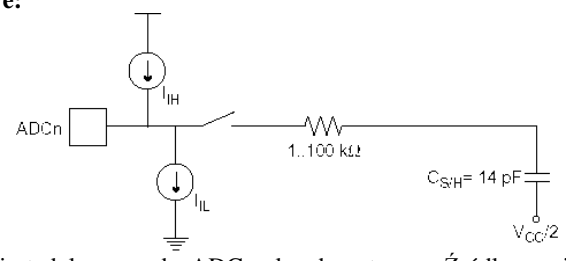
1. Gdy ADATE lub ADEN są wykasowane.
2. podczas konwersji, minimum jeden cykl ADC po pojawieniu się zbocza wyzwalającego.
3. Po konwersji, przed wyzerowaniem flagi przerwania.

Zmiana ADMUX jedną z tych metod spowoduje, że nowe ustawienia będą dotyczyć następnej konwersji.
Należy uważnie postępować przy zmianie kanałów różnicowych. Przy wybranym jednym kanale różnicowym, stan ustabilizowania się nowej wartości może trwać ponad 125 us. Proces przetwarzania nie powinien być uruchomiony przed czasem 125 us od wybrania nowego kanału różnicowego.
Taki sam okres czasu należy przyjąć przy pierwszej konwersji różnicowej, po zmianie źródła odniesienia (przez zmianę bitów REFS1:0 w rejestrze ADMUX).
Kanały wejściowe ADC:
W trybie pojedynczego przetwarzania zawsze należy ustalać kanał wejściowy przed rozpoczęciem przetwarzania. Kanał może być wybrany w jeden takt zegara ADC po wpisie jedynki do ADSC. Prostą metodą jest czekać na zakończenie pierwszej konwersji i wtedy dokonanie zmiany kanału. Jeżeli nowa konwersja zostanie uruchomiona automatycznie, następny wynik będzie skutkiem poprzedniego ustawienia kanału.
Gdy przełącza się na kanał różnicowy, wynik pierwszego przetwarzania może być mało dokładny.
Źródło napięcia odniesienia przetwornika ADC:
Wartość źródła napięcia odniesienia (Vref) decyduje o zakresie pomiarowym przetwornika ADC. Gdy napięcie na pojednyczym kanale przekracza Vref, wynikiem będzie 0x3FF. Vref można ustawić jako napięcie zasilające AVCC, wewnętrzne źródło 2,56V, lub dołączone z zewnątrz do końcówki AREF.
Wewnętrzne napięcie odniesienia jest wytwarzane z wbudowanego źródła i wzmacniacza. W pozostałych przypadkach zewnętrzna końcówka AREF jest bezpośrednio dołączona do przetwornika ADC. Należy wtedy włączyć kondensator pomiędzy końcówkę AREF a masę. Napięcie odniesienia może być też zmierzone przez woltomierz o wysokiej impedancji. Wbudowane źródło odniesienia ma wysoką impedancję i tylko obciążenia pojemnościowe mogą być do niego podłączane.
Jeżeli dołącza się stałe napięcie do końcówki AREF, nie można używać pozostałych źródeł, ponieważ będą zwarte do zewnętrznego źródła. Jeśli nie jest używane zewnętrzne napięcie dołączone do końcówki AREF, to można przełączać między AVCC a 2,56V. Wynik pierwszej konwersji po wybraniu źródła może być niedokładny i należy go pominąć.
Jeśli ustawione są wejścia różnicowe, to nie powinno wybierać się źródła dołączonego do AVCC.
Redukcja szumów w ADC:
Redukcja szumów podczas konwersji w trybie uśpienia redukcji szumów powstających w rdzeniu CPU i urządzeniach we/wy. Układ odszumiania może być używany z redukcją szumów ADC i w trybie Idle. By móc używać tych cech należy:

1. Musi być wybrana pojedyncza konwersja i uaktywnione przerwanie po zakończonym przetwarzaniu.
2. Wprowadzić tryb redukcji szumów (lub tryb Idle). ADC rozpocznie konwersję gdy CPU będzie zatrzymany.
3. Jeśli nie wystąpią inne przerwania przed zakończeniem przetwarzania, przerwanie ADC będzie wybudzać CPU i wykonywać program obsługi przerwania. Jeśli jakieś przerwanie obudzi CPU zanim przetwornik zakończy pracę, to przerwanie to będzie wykonane i przerwanie od ADC również będzie wykonane po zakończeniu przetwarzania. CPU pozostanie w stanie aktywnym dopóki nie zostanie wykonana nowa instrukcja Sleep.

ADC nie wyłączy się automatycznie po wprowadzeniu stanu Sleep, Idle i redukcji szumów. Użytkownik powinien wpisać zero do bitu ADEN zanim przejdzie się w tryb obniżonego poboru mocy. Gdy przetwornik pracuje w trybie Sleep i wejścia są ustawione jako różnicowe, to po wyłączeniu i włączeniu przetwornika po wybudzeniu procesora wynik przetwarzania będzie nieprawidłowy.

Analogowe obwody wejściowe:



Źródło sygnału analogowego jest dołączone do ADC z kondensatorem. Źródło musi wysterować kondensator układu S/H przez szeregowy rezystor.

Przetwornik może współpracować z sygnałami analogowymi o impedancji poniżej 10kΩ. Jeśli takie źródło będzie użyte, czas próbkowania będzie nieistotne. Użycie źródła o wyższej impedancji spowoduje wydłużenie czasu próbkowania o czas potrzebny na naładowanie kondensatora S/H.

Przy wejściach różnicowych, wymagana jest impedancja źródła poniżej kilkuset kΩ.

W celu uniknięcia zniekształceń najwyższa harmoniczna sygnału powinna być mniejsza od $f_{ADC}/2$. W przeciwnym przypadku należy zastosować filtr dolnoprzepustowy przed wejściem ADC.

Usuwanie szumów:

Obwody cyfrowe wytwarzają wewnątrz i na zewnątrz układu zakłócenia EMI mogące mieć wpływ na dokładność pomiaru sygnału analogowego. Jeśli wymagana jest wysoka dokładność poziom zakłóceń można zminimalizować następującymi sposobami:

1. ścieżka sygnału analogowego powinna być jak najkrótsza, poprowadzona obok masy analogowej z dala od ścieżek o wysokiej częstotliwości sygnałach cyfrowych.
2. końcówka AVCC powinna być dołączona do napięcia zasilania Vcc części cyfrowej przez dolnoprzepustowy filtr LC
3. Użycie funkcji redukcji szumów ADC podczas konwersji
4. jeśli któraś z końcówek portu ADC służy jako wyjście cyfrowe, podczas konwersji nie należy przełączać jej stanu.

Kompensacja offsetu:

W układ wbudowany jest mechanizm kompensacji napięcia offsetu podczas pomiaru różnicowego poniżej najmłodszego bitu.

Dokładność:

ADC przetwarza napięcie analogowe liniowo pomiędzy GND a V_{REF} w 2^n krokach. Najmniejsza wartość odczytywana jest jako 0, największa 2^{n-1}.

Wynik przetwarzania:

Po zakończeniu przetwarzania (ADIF w stanie wysokim) wynik jest umieszczany w rejestrach ADCL i ADCH.

Przy konwersji prostej wynik jest równy:

$$ADC = \frac{V_{IN} \cdot 1024}{V_{REF}}$$

gdzie V_{IN} jest napięciem wejściowym na wybranym wejściu, V_{REF} wybranym źródłem napięcia odniesienia. 0x000 oznacza sygnał wejściowy na poziomie masy, 0x3FF – wejście na poziomie napięcia odniesienia minus 1 LSB.

Przy konwersji różnicowej:

$$ADC = \frac{(V_{POS} - V_{NEG}) \cdot wzmocnienie \cdot 512}{V_{REF}}$$

gdzie V_{POS} – napięcie na końcówce dodatniej, V_{NEG} – napięcie na końcówce ujemnej, wzmocnienie – wybrany stopień wstępnego wzmocnienia sygnału, V_{REF} – wybrane źródło odniesienia. Wynik jest przedstawiany jako liczba ze znakiem od 0x200 (-512d) do 0x1FF (+511d). Aby szybko sprawdzić znak uzyskanego wyniku wystarczy sprawdzić ostatni bit pomiaru (ADC9 w rejestrze ADCH). Stan wysoki oznacza napięcie ujemne. Stan niski – dodatnie.

Rejestr wejściowego multipleksera – ADMUX:

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bity 7,6 – REFS1:0 – bity wyboru odniesienia**

pozwalają na wybór źródła napięcia odniesienia dla ADC. Jeśli zostaną zmienione podczas konwersji, skutek będzie dopiero po zakończeniu przetwarzania (ustawi się ADIF w ADCSRA). Nie musi być wybrane wewnętrzne odniesienie podczas gdy zewnętrzne napięcie odniesienia jest dołączone do końcówki V_{REF}.

REFS1	REFS0	Napięcie odniesienia
0	0	AREF, wewnętrzne Vref wyłączone
0	1	AVCC z zewnętrznym kondensatorem przy końcówce AREF
1	0	Zarezerwowane
1	1	Wewnętrzne 2,56V z zewnętrznym kondensatorem przy końcówce AREF

- **bit 5 – ADLAR: przesunięcie wyniku**

stan tego bitu wpływa na sposób przedstawienia wyniku w rejestrach ADC. ADLAR w stanie wysokim spowoduje przesunięcie wyniku w lewo, w niskim – w prawo. Zmiana stanu bitu daje natychmiastowy efekt.

- **bity 4:0 – MUX4:0 – kanały analogowe**

Wartość w tych bitach określa które wejścia analogowe są dołączone do ADC. określają również wzmocnienie dla kanałów różnicowych. Zmiana tych bitów będzie miała wpływ dopiero po zakończonej konwersji.

MUX4...0	Pojedyncze wejście	Dodatnie wejście różnicowe	Ujemne wejście różnicowe	wzmocnienie
00000	ADC0			
00001	ADC1			
00010	ADC2			
00011	ADC3			
00100	ADC4			
00101	ADC5			
00110	ADC6			
00111	ADC7			
01000		ADC0	ADC0	10x
01001		ADC1	ADC0	10x
01010		ADC0	ADC0	200x
01011		ADC1	ADC0	200x
01100		ADC2	ADC2	10x
01101		ADC3	ADC2	10x
01110(1)		ADC2	ADC2	200x
01111(1)		ADC3	ADC2	200x
10000		ADC0	ADC1	1x
10001		ADC1	ADC1	1x
10010		ADC2	ADC1	1x
10011		ADC3	ADC1	1x
10100		ADC4	ADC1	1x
10101		ADC5	ADC1	1x
10110		ADC6	ADC1	1x
10111		ADC7	ADC1	1x
11000		ADC0	ADC2	1x
11001		ADC1	ADC2	1x
11010		ADC2	ADC2	1x
11011		ADC3	ADC2	1x
11100		ADC4	ADC2	1x
11101		ADC5	ADC2	1x
11110	1,22V (V_{BG})			
11111	0V (GND)			

(1) – wejścia różnicowe nie były testowane w układach w obudowach DIL. Ta możliwość dostępna jest tylko w układach w obudowach TQFP i MLF.

Rejestr A kontroli i stanu ADC – ADCSRA:

Bit	7	6	5	4	3	2	1	0	
	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Odczyt/zapis	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bit 7 – ADEN: włączenie ADC**

wpisanie jedynki włącza przetwornik. Wpisanie zera wyłącza. Wyłączenie przetwornika w trakcie konwersji daje błędny wynik końcowy.

- **bit 6 – ADSC: rozpoczęcie konwersji**

W trybie pojedynczej konwersji wpis jedynki rozpoczyna konwersję. W trybie ciągłej pracy Wpisanie jedynki uruchamia pierwszą konwersję. Pierwsza konwersja po tym jak ADSC zostaje ustawiony po włączeniu ADC, lub gdy ADSC jest ustawione równocześnie z przetwornikiem ADC, potrzebuje 25 cykli zegara w stosunku do 13 przy pracy normalnej.
Odczyt z ADSC daje jedynkę tak długo jak trwa przetwarzanie. Gdy zostaje zakończone pojawia się zero.
Wpisanie ręcznie zera nie daje skutku.

- **bit 5 – ADATE: auto-wyzwalanie**

Auto-wyzwalanie zostaje włączone gdy ADATE = 1. ADC rozpoczyna pomiar wraz z dodatnim impulsem wyzwalającym. Źródło wyzwalania jest wybrane przez bit ADTS w SFIOR.

- **bit 4 – ADIF: flaga przerwania ADC**

Jest ustawiana po zakończeniu konwersji (i wpisie wyniku do rejestrów danych). Bit jest zerowany sprzętowo podczas wykonywania programu obsługi przerwania. Zerowanie programowe polega na wpisaniu jedynki do tego bitu.

- **bit 3 – ADIE: uaktywnienie przerwania ADC:**

Gdy bit ADIE i bit I w SREG są ustawione, przerwanie od przetwornika ADC są aktywne.

- **bity 2...0 – ADPS2:0 – preskaler:**

ustalają podział częstotliwości XTAL dla przetwornika:

ADPS2	ADPS1	ADPS0	Podział
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Rejestry danych ADC – ADCL i ADCH

ADLAR = 0

Bit	15	14	13	12	11	10	9	8	
	-	-	-	-	-	-	ADC9	ADC8	ADCH
	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Odczyt/zapis	O	O	O	O	O	O	O	O	
	O	O	O	O	O	O	O	O	
Wartość początkowa	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

ADLAR = 1

Bit	15	14	13	12	11	10	9	8	
	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
	ADC1	ADC0	-	-	-	-	-	-	ADCL
	7	6	5	4	3	2	1	0	
Odczyt/zapis	O	O	O	O	O	O	O	O	
	O	O	O	O	O	O	O	O	
Wartość początkowa	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

Po zakończeniu konwersji wynik znajduje się w tych rejestrach. Gdy używane były wejścia różnicowe, wynik jest liczbą ze znakiem.
Jeżeli wynik jest przesunięty w lewo (ADLAR=1) i nie jest potrzebna precyzja większa niż 8-bitów, wystarczy odczytać tylko rejestr ADCH. W każdym innym przypadku należy odczytać najpierw ADCL potem ADCH.

Rejestr specjalny – SFIOR:

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	-	ACME	PUD	PSR2	PSR10	SFIOR
Odczyt/zapis	Z/O	Z/O	Z/O	O	Z/O	Z/O	Z/O	Z/O	
Wartość początkowa	0	0	0	0	0	0	0	0	

- **bity 7...5: wybór źródła automatycznego wyzwalania ADC:**

Jeśli bit ADATE w rejestrze ADCSRA jest ustawiony, wartość tych bitów wpływa na źródło wyzwalające przetwornik ADC. Jeżeli ADATE jest wyzerowane, ustawienia ADTS2...0 nie są istotne. Przetwarzanie będzie rozpoczęte przez narastające zbocze wybranej flagi przerwania. Przełączenie ze stanu niskiego w stan wysoki źródła wyzwalania wygeneruje dodatnie zbocze. Jeżeli ADEN w ADCSRA jest ustawione, rozpocznie się przetwarzanie. W trybie Free Running (ADTS2:0 = 0) nie spowoduje wyzwolenia przetwornika nawet jeżeli flaga przerwania /ADC jest ustawiona.

ADTS2	ADTS1	ADTS0	Źródło wyzwalania
0	0	0	Tryb Free Runnung
0	0	1	Analogowy komparator
0	1	0	Przerwanie zewnętrzne 0
0	1	1	Porównanie w liczniku TC0
1	0	0	Przepełnienie licznika TC0
1	0	1	Porównanie B w liczniku TC1
1	1	0	Przepełnienie licznika TC1
1	1	1	Przechwycenie w liczniku TC1

- **bit 4 – zarezerwowany.**

Opis rejestrów

Adres	nazwa	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$3F ($5F)	SREG	I	T	H	S	V	N	Z	C
$3E ($5E)	SPH	-	-	-	-	SP11	SP10	SP9	SP8
$3D ($5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
$3C ($5C)	OCR0	Rejestr wartości porównywanej w liczniku TC0							
$3B ($5B)	GICR	INT1	INT0	INT2	-	-	-	IVSEL	IVCE
$3A ($5A)	GIFR	INTF1	INTF0	INTF2	-	-	-	-	-
$39 ($59)	TIMSK	OCIE2	TICIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0
$38 ($58)	TIFR	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0
$37 ($57)	SPMCR	SPMIE	RWWSB	-	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN
$36 ($56)	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
$35 ($55)	MCUCR	SE	SM2	SM1	SM0	ISC11	ISC10	ISC01	ISC00
$34 ($54)	MCUCSR	JTD	ISC2	-	JTRF	WDRF	BORF	EXTRF	PORF
$33 ($53)	TCCR0	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00
$32 ($52)	TCNT0	Rejestr licznika TC0 (8-bit)							
$31 ($51)	OSCCAL	Rejestr kalibracji wewnętrznego generatora							
	OCDR	Rejestr On-Chip Debug							
$30 ($50)	SFIOR	ADTS2	ADTS1	ADTS0	-	ACME	PUD	PSR2	PSR10
$2F ($4F)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10
$2E ($4E)	TCCR1B	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10
$2D ($4D)	TCNT1H	Rejestr licznika TC1 – część starsza							
$2C ($4C)	TCNT1L	Rejestr licznika TC1 – część młodsza							
$2B ($4B)	OCR1AH	Rejestr wartości porównywanej A w liczniku TC1 – część starsza							
$2A ($4A)	OCR1AL	Rejestr wartości porównywanej A w liczniku TC1 – część młodsza							
$29 ($49)	OCR1BH	Rejestr wartości porównywanej B w liczniku TC1 – część starsza							
$28 ($48)	OCR1BL	Rejestr wartości porównywanej B w liczniku TC1 – część młodsza							
$27 ($47)	ICR1H	Rejestr wartości przechwyconej w liczniku TC1 – część starsza							
$26 ($46)	ICR1L	Rejestr wartości przechwyconej w liczniku TC1 – część młodsza							
$25 ($45)	TCCR2	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20
$24 ($44)	TCNT2	Rejestr licznika TC2 (8-bit)							
$23 ($43)	OCR2	Rejestr wartości porównywanej w liczniku TC2							
$22 ($42)	ASSR	-	-	-	-	AS2	TCN2UB	OCR2UB	TCR2UB
$21 ($41)	WDTCR	-	-	-	WDTOE	WDE	WDP2	WDP1	WDP0
$20 ($40)	UBRRH	URSEL	-	-	-	UBRR[11:8]			
	UCSRC	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL
$1F ($3F)	EEARH	-	-	-	-	-	-	EEAR9	EEAR8
$1E ($3E)	EEARL	Rejestr adresu EEPROM – młodsza część							
$1D ($3D)	EEDR	Rejestr danych EEPROM							
$1C ($3C)	EECR	-	-	-	-	EERIE	EEMWE	EEWE	EERE
$1B ($3B)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0
$1A ($3A)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0
$19 ($39)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0
$18 ($38)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
$17 ($37)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
$16 ($36)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0
$15 ($35)	PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0
$14 ($34)	DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0
$13 ($33)	PINC	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0
$12 ($32)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0
$11 ($31)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0
$10 ($30)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0
$0F ($2F)	SPDR	Rejestr danych SPI							
$0E ($2E)	SPSR	SPIF	WCOL	-	-	-	-	-	SPI2X
$0D ($2D)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
$0C ($2C)	UDR	Rejestr danych I/O – USART							
$0B ($2B)	UCSRA	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM
$0A ($2A)	UCSRB	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8
$09 ($29)	UBRRL	Prędkość USART – część młodsza							
$08 ($28)	ACSR	ADC	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0
$07 ($27)	ADMUX	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0
$06 ($26)	ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0
$05 ($25)	ADCH	Rejestr danych przetwornika ADC – część starsza							
$04 ($24)	ADCL	Rejestr danych przetwornika ADC – część młodsza							
$03 ($23)	TWDR	Rejestr danych TWI							
$02 ($22)	TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
$01 ($21)	TWSR	TWS7	TWS6	TWS5	TWS4	TWS3	-	TWPS1	TWPS0
$00 ($20)	TWBR	Podział częstotliwości dla TWI							