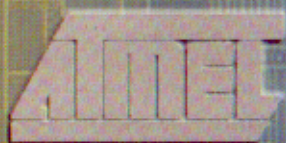


Arkadiusz Krysiak

Mikrokontrolery rodziny AVR[®]

**w obudowach
ośmiowyprowadzeniowych**



Wrocław 2000

Arkadiusz Krysiak

Mikrokontrolery rodziny AVR[®]

**w obudowach
ośmiowyprowadzeniowych**

Wrocław 2000

SPIS TREŚCI

Wstęp	9
 CHARAKTERYSTYKA OGÓLNA OŚMIOWYPROWADZENIOWYCH MIKROKONTROLERÓW RODZINY AVR	 11
 WŁAŚCIWOŚCI MIKROKONTROLERÓW	 12
Opis wyprowadzeń mikrokontrolerów	15
Rysunek 1. Rozkład wyprowadzeń mikrokontrolerów	15
Parametry elektryczne i czasowe	16
Typowe parametry pracy	17
Tabela 1. Charakterystyka stałoprądowa	17
Wykres 1. Przebieg sygnału oscylatora zewnętrznego	18
Tabela 2, 3. Parametry czasowe sygnału oscylatora zewnętrznego	19
 BUDOWA MIKROKONTROLERA	 20
Charakterystyka jednostki centralnej	20
Budowa blokowa	20
Tabela 4. Zestawienie podstawowych właściwości mikrokontrolerów	21
Rejestr stanu mikrokontrolera (STATUS REGISTER)	21
Oscylator	22
Schemat 1. Podłączenie układów taktujących mikrokontroler	23
Tabela 5. Konfiguracja układu taktującego kontroler	23
Układ taktowania i sterowania	24
Schemat 2. Układy RESET mikrokontrolera	24
Tabela 6. Parametry elektryczne układu RESET	26
Wykres 2. Przebiegi wyzwalania układu RESET	28
BOD (Brown Out Detection)	30
Watchdog	30
Schemat 3. Układ watchdog	30
Preskaler dla zegara licznika TC0	32
Schemat 4. Preskaler dla zegarów/liczników TC0	32

Układ zegara/licznika (TIMER/COUNTER) TC0	33
Schemat 5. Zegar/licznik TC0	34
Prescaler dla zegara licznika TC1	35
Schemat 6. Prescaler dla TC1	35
Układ zegara/licznika (TIMER/COUNTER 1) TC1	36
Schemat 7. Zegar/licznik TC1	37
Tryb zliczania	38
Tryb generatora PWM	38
Układ przetwornika analogowo-cyfrowego	39
Schemat 8. Przetwornik A/C	41
Tabela 7. Konfiguracja wejść przetwornika AC	43
Schemat 9. Prescaler przetwornika ADC	43
Układ komparatora analogowego	46
Schemat 10. Komparator analogowy	46
Porty wejścia/wyjścia	48
Tabela 8. Konfigurowanie portu	48
Tabela 9. Funkcje specjalne wyprowadzeń portu B	49
Rejestry robocze, obszar wejścia/wyjścia i pamięć danych	50
Stos	50
Pamięć EEPROM	51
Pamięć programu	53
Układ przerwań	53
Przyjęcie i zakończenie przerwania	57
Czas obsługi przerwania	57
Tabela 10. Wektory obsługi zdarzeń	58
TRYBY PRACY MIKROKONTROLERA Z OBNIŻONYM POBOREM MOCY	59
Idle	59
Power Down	60
ADC Tryb redukcji szumów	60
TRYBY ADRESOWANIA PAMIĘCI PROGRAMU, DANYCH I OBSZARU WEJŚCIA/WYJŚCIA	61
Tryb bezpośredniego adresowania rejestrów	61
Tryb bezpośredniego adresowania dwóch rejestrów	61
Tryb bezpośredniego adresowania obszaru wejścia/wyjścia	61
Tryb bezpośredniego adresowania pamięci danych	61

Tryb pośredniego adresowania danych z przemieszczeniem.	62
Tryb pośredniego adresowania rejestrów	62
Tryb adresowania pośredniego danych z pre-dekrementacją	62
Tryb adresowania pośredniego danych z post-inkrementacją	62
Tryb adresowania stałych z użyciem rozkazu LPM.	63
Tryb adresowania pośredniego pamięci programu. (IJMP, ICALL).	63
Tryb adresowania względnego pamięci programu. (RJMP, RCALL)	63
 REJESTRY KONTROLNE I STERUJĄCE MIKROKONTROLERA	64
Tabela 11. Sumaryczne przedstawienie rejestrów funkcji specjalnych.	64
SREG \$3F rejestr statusu	64
SPL \$3D rejestr wskaźnika wierzchołka stosu	65
GIMSK \$3B rejestr ogólnego maskowania przerwań	65
GIFR \$3A rejestr flag przerwań zewnętrznych.	66
TIMSK \$39 rejestr maskowania przerwań od TCx	66
TIFR \$38 rejestr flagi zgłoszenia przerwania od zegarów/liczników	66
MCUCR \$35 rejestr kontrolny mikrokontrolera	67
MCUSR \$34	67
TCCR0 \$33 rejestr kontrolny zegara/licznika TC0	67
TCNT0 \$32 rejestr zegara/licznika TC0	68
OSCCAL \$31 rejestr kalibracji generatora taktującego kontroler	68
TCCR1 \$30 rejestr kontrolny zegara TC1.	68
WDTCR \$21 rejestr kontrolny zegara watchdog	69
EEAR \$1E rejestr adresowy pamięci EEPROM	70
EEDR \$1D rejestr danych pamięci EEPROM.	70
EECR \$1C rejestr kontrolny pamięci EEPROM	70
PORTB \$18 rejestr danych portu B.	71
DDRB \$17 rejestr kierunku portu B.	71
PINB \$16 rejestr wejściowy portu B	71
ACSR \$08 rejestr kontrolny i statusu komparatora analogowego.	71
ADMUX \$07 rejestr sterowania multiplekserami przetwornika A/C.	72
ADCSR \$06 rejestr stanu przetwornika A/C	73
ADCH \$05 starsza część rejestru danych przetwornika A/C	73
ADCL \$04 młodsza część rejestru danych przetwornika A/C	74
 LISTA ROZKAZÓW.	75

PROGRAMOWANIE MIKROKONTROLERÓW	119
Informacje wstępne	119
Zabezpieczenie przed odczytaniem lub skasowaniem pamięci programu ..	119
Tabela 12. Bity zabezpieczające zawartość pamięci programu	119
Bity konfiguracyjne	120
Bajty sygnatur	121
Układ programowania szeregowego wysokonapięciowego	121
Interfejs fizyczny	121
Schemat 11. Układ programowania szeregowego wysokonapięciowego. .	122
Sekwencja załączenia zasilania programowanego kontrolera	122
Dodatkowe uwagi do programowania	123
Sekwencja wyłączenia zasilania po zaprogramowaniu kontrolera	123
Przebiegi sygnału w trakcie programowania wysokonapięciowego	123
Wykres 3. Przebiegi transmisji dla trybu programowania wysokonapięciowego	124
Charakterystyka przebiegów czasowych w trakcie programowania szeregowego wysokonapięciowego	124
Wykres 4. Przebiegi czasowe transmisji dla trybu programowania wysokonapięciowego	124
Tabela 13. Parametry czasowe sygnału w trakcie programowania wysokonapięciowego	125
Rozkazy trybu programowania wysokonapięciowego	125
Tabela 14. Rozkazy dla trybu programowania wysokonapięciowego	126
Opis programowania szeregowego niskonapięciowego (Downloading)	127
Interfejs fizyczny	127
Schemat 12. Układ programowania szeregowego niskonapięciowego	128
Sekwencja załączenia zasilania programowanego kontrolera	128
Dodatkowe uwagi do trybu programowania niskonapięciowego	129
Sekwencja wyłączenia zasilania programowanego kontrolera	129
Wykres 5. Przebiegi transmisji dla trybu programowania niskonapięciowego	130
Wykres 6. Przebiegi czasowe transmisji dla trybu programowania niskonapięciowego	130
Tabela 15. Charakterystyka czasowa trybu programowania niskonapięciowego	130
Tabela 16. Formaty rozkazów dla trybu programowania niskonapięciowego	131

Wstęp

Na polskim rynku znane były kontrolery w obudowach ośmiowypro-
wadzeniowych tylko jednej firmy. Firma ta (niewymieniana z nazwy) re-
klamowała swój produkt jako jedyny. W rzeczywistości już w owym
czasie istniały trzy równoległe rozwiązania różnych firm. Prócz kontro-
lerów opartych o jednostkę PIC, istniały rozwiązania oparte o jednostkę
rodziny „51” i „AVR”. Jednak możliwościami najkorzystniej prezentują
się AVR i „51”. Po raz pierwszy w języku polskim pojawia się opis tego
typu kontrolerów. Ma on szansę zapoczątkować boom na zastosowa-
nie tych niepozornych układów.

Jest to już trzecie spotkanie z książką, opisującą mikrokontrolery
z rodziny AVR. Opracowana została na podstawie materiałów firmy
ATMEL. Forma ułożenia treści została zmodyfikowana w stosunku do
poprzednich publikacji, aby usprawnić korzystanie z materiałów w niej
zawartych. Proszę o uwagi na temat publikacji, które pomogłyby mi
wyjść naprzeciw potrzebom i zainteresowaniom czytelników.

Charakterystyka ogólna ośmiowyprowadzeniowych mikrokontrolerów rodziny AVR

Jednoukładowe mikrokontrolery AVR w obudowach ośmiowyprowadzeniowych zostały opracowane w związku z rosnącym zapotrzebowaniem na małe sterowniki, o dużych możliwościach. Całkowicie statyczna budowa mikrokontrolera, niski pobór mocy i szeroki zakres napięć zasilających, rozszerzają dość znacznie zakres ich zastosowań. Budowa mikrokontrolera opiera się o architekturę harwardzką. Dużą szybkość mikrokontrolera zapewnia przetwarzanie potokowe, powodujące wykonywanie większości rozkazów mieszczące się w jednym cyklu zegarowym, oraz 32-bajtowy (bardzo duży) obszar rejestrów roboczych o natychmiastowym dostępie. Ich dodatkową zaletą jest brak ścisłego określenia akumulatora (architektura RISC). Tę funkcję może pełnić dowolnie wybrany rejestr, spośród 32-bajтового banku rejestrów roboczych. Zapewnia to bardzo znaczny wzrost szybkości implementowanych algorytmów w stosunku do mikrokontrolerów z typową budową, z określonym umiejscowieniem akumulatora. Jest to efekt rozłożenia dość znacznego „obciążenia” akumulatora przesłaniami na znacznie większą ilość rejestrów, co zapewnia eliminację wielu z nich, dokonywanych w celu zapamiętania wyników pośrednich przy większych obliczeniach. Konstruktorzy uwzględnili również wzrastającą potrzebę na wbudowaną pamięć EEPROM, służącą do zapamiętania konfiguracji urządzeń budowanych z wykorzystaniem mikrokontrolerów. Kontrolery te posiadają również układ watchdog oraz wbudowany tryb pracy z obniżonym poborem mocy, które w obecnej chwili stają się standardem w mikrokontrolerach.

Opisywanymi w tej książce mikrokontrolerami są: AT90S2323, AT90S2343. ATtiny1X, ATtiny22. Opracowano je z myślą o prostych układach sterujących, pomiarowych i nadzoru, wbudowanych w urządzenia i niewidocznych dla użytkownika, zapewniający komunikację dla innych urządzeń. Nie należy się jednak tym sugerować, możliwości tych kontrolerów są znacznie większe.

Właściwości mikrokontrolerów

Poniżej zostały przedstawione podstawowe właściwości opisujących mikrokontrolerów. Krótka charakterystyka ułatwi szybki wybór mikrokontrolera do zadania, w którym ma zostać użyty.

AT90S2323P

- 8-wyprowadzeniowa obudowa
- rozszerzona architektura RISC AVR
- 118 rozkazów wykonywanych głównie w jednym cyklu zegarowym
- 2 KB pamięci Flash reprogramowalnej w układzie za pomocą SP o trwałości 1000 cykli kasowania/zapisu
- 128 bajtów wewnętrznej pamięci RAM
- 128 bajtów wewnętrznej pamięci EEPROM o trwałości 100 000 cykli zapisu/kasowania
- programowane bity zabezpieczenia programu i pamięci EEPROM
- 32 ośmiobitowe rejestry robocze
- 3-programowalne linie wejścia/wyjścia
- w pełni statyczna budowa umożliwiająca pracę w zakresie 0-10 MHz
- cykl rozkazowy o długości 100 ns przy 10 MHz
- jeden ośmiobitowy zegar licznik z oddzielnym preskalerem (wstępny dzielnik)
- programowalny układ Watchdog z wbudowanym wewnętrznym oscylatorem
- układ przerwań z zewnętrznymi i wewnętrznymi źródłami przerwań
- dwa tryby pracy z obniżonym poborem mocy
- wyjścia portów o wydajności 20 mA zarówno w stanie wysokim, jak i niskim, umożliwiające bezpośrednie sterowanie wyświetlaczami LED i transoptorów
- bardzo niski pobór mocy
- Vcc 2,7-6,0 V

AT90S2343

- 8-wyprowadzeniowa obudowa
- rozszerzona architektura RISC AVR
- 118 rozkazów wykonywanych głównie w jednym cyklu zegarowym
- 2 KB pamięci Flash reprogramowalnej w układzie za pomocą SP o trwałości 1000 cykli kasowania/zapisu
- 128 bajtów wewnętrznej pamięci RAM

- 128 bajtów wewnętrznej pamięci EEPROM o trwałości 100 000 cykli zapisu/kasowania
- programowane bity zabezpieczenia programu i pamięci EEPROM
- 32 ośmiobitowe rejestry robocze
- 5-programowalne linie wejścia/wyjścia
- w pełni statyczna budowa umożliwiająca pracę w zakresie 0-10 MHz
- cykl rozkazowy o długości 100 ns przy 10 MHz
- jeden ośmiobitowy zegar licznik z oddzielnym preskalerem (wstępny dzielnik)
- programowalny układ Watchdog z wbudowanym wewnętrznym oscylatorem
- układ przerwań z zewnętrznymi i wewnętrznymi źródłami przerwań
- dwa tryby pracy z obniżonym poborem mocy
- wyjścia portów o wydajności 20 mA zarówno w stanie wysokim, jak i niskim, umożliwiające bezpośrednie sterowanie wyświetlaczy LED
- bardzo niski pobór mocy
- Vcc 2,7-6,0 V

ATTiny10/11/12

- 8-wyprowadzeniowa obudowa
- rozszerzona architektura RISC AVR
- 90 rozkazów wykonywanych głównie w jednym cyklu zegarowym
- 1 KB pamięci Flash reprogramowalnej w układzie za pomocą SPI o trwałości 1000 cykli kasowania/zapisu
- 64 bajty wewnętrznej pamięci EEPROM o trwałości 100 000 cykli zapisu/kasowania
- programowane bity zabezpieczenia programu i pamięci EEPROM
- 32 ośmiobitowe rejestry robocze
- 6-programowalne linie wejścia/wyjścia
- w pełni statyczna budowa, umożliwiająca pracę w zakresie 0-8 MHz (w zależności od wersji mikrokontrolera)
- wewnętrzny kalibrowany generator RC
- jeden ośmiobitowy zegar licznik z oddzielnym preskalerem
- programowalny układ Watchdog z wbudowanym wewnętrznym oscylatorem
- układ przerwań z zewnętrznymi i wewnętrznymi źródłami przerwań
- wewnętrzny komparator analogowy
- dwa tryby pracy z obniżonym poborem mocy
- wyjścia portów o wydajności 20 mA zarówno w stanie wysokim, jak i niskim, umożliwiające bezpośrednie sterowanie wyświetlaczy LED

- bardzo niski pobór mocy
przy 4 MHz, 3 V, 25 °C
praca 2,2 mA
tryb oczekiwania 0,5 mA
tryb uśpienia <1 µA
- Vcc 1,8-5,5 V (w zależności od wersji mikrokontrolera)

ATtiny15

- 8-wyprowadzeniowa obudowa
- rozszerzona architektura RISC AVR
- 90 rozkazów wykonywanych głównie w jednym cyklu zegarowym
- 1 KB pamięci Flash reprogramowalnej w układzie za pomocą SPI o trwałości 1000 cykli kasowania/zapisu
- 128 bajtów wewnętrznej pamięci RAM
- 128 bajtów wewnętrznej pamięci EEPROM o trwałości 100 000 cykli zapisu/kasowania
- programowane bity zabezpieczenia programu i pamięci EEPROM
- 32 ośmiobitowe rejestry robocze
- 6-programowalne linie wejścia/wyjścia
- w pełni statyczna budowa umożliwiająca pracę w zakresie 0-10 MHz
- cykl rozkazowy o długości 100 ns przy 10 MHz
- dwa zegary liczniki z oddzielnymi preskalerami, w tym jeden z wyjściem PWM
- czterowejściowy 10-bitowy przetwornik A/C
- programowalny układ Watchdog z wbudowanym wewnętrznym oscylatorem
- układ przerwań z zewnętrznymi i wewnętrznymi źródłami przerwań
- dwa tryby pracy z obniżonym poborem mocy
- wyjścia portów o wydajności 20 mA zarówno w stanie wysokim, jak i niskim umożliwiające bezpośrednie sterowanie wyświetlacz LED
- bardzo niski pobór mocy
- Vcc 2,7-6,0 V

ATtiny22

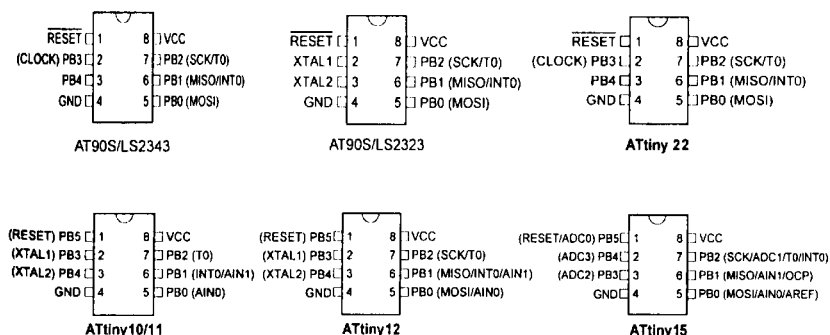
- 8-wyprowadzeniowa obudowa
- rozszerzona architektura RISC AVR
- 118 rozkazów wykonywanych głównie w jednym cyklu zegarowym
- 2 KB pamięci Flash reprogramowalnej w układzie za pomocą SPI o trwałości 1000 cykli kasowania/zapisu
- 128 bajtów wewnętrznej pamięci RAM

- 128 bajtów wewnętrznej pamięci EEPROM o trwałości 100 000 cykli zapisu/kasowania
- programowane bity zabezpieczenia programu i pamięci EEPROM
- 32 ośmiobitowe rejestry robocze
- 5-programowalne linie wejścia/wyjścia
- w pełni statyczna budowa umożliwiająca pracę w zakresie 0-8 MHz
- jeden ośmiobitowy zegar licznik z oddzielnym preskalerem (wstępny dzielnik)
- programowalny układ Watchdog z wbudowanym wewnętrznym oscylatorem
- układ przerwań z zewnętrznymi i wewnętrznymi źródłami przerwań
- dwa tryby pracy z obniżonym poborem mocy
- wyjścia portów o wydajności 20 mA zarówno w stanie wysokim, jak i niskim, umożliwiające bezpośrednie sterowanie wyświetlaczy LED
- bardzo niski pobór mocy
- Vcc 2,7-6,0 V

Opis wyprowadzeń mikrokontrolerów

Rysunek 1 przedstawia rozkład sygnałów na wyprowadzeniach mikrokontrolerów, a poniżej opis funkcji jaką pełnią.

Rysunek 1. Rozkład wyprowadzeń mikrokontrolerów



Vcc
GND

Plus zasilania
Masa

Port B

(PB0-PB5)	dwukierunkowy port wejścia/wyjścia z wewnętrznym podciąganiem wybieranym poprzez ustawienie odpowiednich bitów, indywidualnie dla każdej z linii portu w momencie ustawienia jej jako wejścia. Maksymalny prąd wejściowy każdej z linii wynosi 20 mA. W zależności od typu kontrolera port składa się z 3-6 linii
AIN0	wejście nieodwracające komparatora
AIN1	wejście odwracające komparatora
OCP	wyjście porównania licznika T/C1 (PWM)
MOSI	wejście szeregowe danych trybu programowania
MISO	wyjście szeregowe danych trybu programowania
SCK	wejście taktu zegara dla trybu programowania
INT0	wejście przerwania zewnętrznego
T0	wejście zewnętrznego sygnału zegarowego dla licznika T0
ADC0-ADC3	wejścia przetwornika analogowo-cyfrowego o rozdzielczości 10 bitów
ICP	wejście przechwytywania zegara T1
VREF	wejście napięcia odniesienia dla przetwornika A/C
RESET	wejście zerowania mikrokontrolera. Wystąpienie stanu niskiego na tym wyprowadzeniu przez okres dwóch cykli rozkazowych, gdy uruchomiony jest oscylator mikrokontrolera powoduje zerowanie.
XTAL1	wejście odwracające wzmacniacza oscylatora służące do dołączenia kwarcu. Wejście zewnętrznego sygnału taktującego
XTAL2	wyjście oscylatora służące do dołączenia kwarcu. Przy dołączeniu zewnętrznego sygnału taktującego nie wykorzystane.
CLOCK	wejście zewnętrznego sygnału taktującego mikrokontroler

Parametry elektryczne i czasowe

Przedstawione parametry zostały podane na podstawie katalogu firmy Atmel.

Parametry dopuszczalne

Temperatura pracy	-55°C do +125°C
Temperatura przechowywania	-65 °C do +150 °C

Napięcia na wyprowadzeniach z wyłączeniem wejścia RESET mierzone względem wyprowadzenia GND	-1,0V do $V_{CC}+0,5V$
Maksymalne napięcie pracy	6,6V
Prąd linii I/O	40,0 mA
Prąd wyprowadzeń V_{CC} i GND	200,0 mA

Producent nie gwarantuje całkiem poprawnych zachowań mikrokontrolera, w przypadku wystąpienia któregoś z warunków ekstremalnych.

Typowe parametry pracy

Tabela 1, 2 i 3 przedstawiają typowe parametry elektryczne i czasowe pracy mikrokontrolerów. Parametr dotyczy tylko tych bloków danego mikrokontrolera, które fizycznie w nim się znajdują. „22/15” – zapisane w tabeli oznacza, iż dany parametr dotyczy tylko kontrolerów ATtiny22 i ATtiny15.

Tabela 1. Charakterystyka stałoprądowa

Sym-bol	Parametr	warunki pomiaru	Min	Typ	Max	J.M.
V_{IL}	Wejściowe napięcie w stanie niskim	Prócz XTAL	-0,5		$0,3V_{CC}$	V
V_{IL1}	Wejściowe napięcie w stanie niskim	XTAL	-0,5		0,1	V
V_{IH}	Wejściowe napięcie w stanie wysokim	(nie dotyczy XTAL, RESET)	$0,6V_{CC}$		$V_{CC}+0,5$	V
V_{IH1}	Wejściowe napięcie w stanie wysokim	XTAL	$0,7V_{CC}$		$V_{CC}+0,5$	V
V_{IH2}	Wejściowe napięcie w stanie wysokim	RESET	$0,85V_{CC}$		$V_{CC}+0,5$	V
V_{OL}	Wyjściowe napięcie w stanie niskim (dotyczy portu B)	$I_{OL}=20mA, V_{CC}=5V$ $I_{OL}=10mA, V_{CC}=3V$			0,5 0,4	V
		$I_{OL}=20mA, V_{CC}=5V$ $I_{OL}=10mA, V_{CC}=3V$ (10/11)			0,6 0,5	V
		$I_{OL}=12mA, V_{CC}=5V$ $I_{OL}=6mA, V_{CC}=3V$ (12)			0,6 0,5	V
V_{OH}	Wyjściowe napięcie w stanie wysokim (dotyczy portu B)	$I_{OH}=-3mA, V_{CC}=5V$ $I_{OH}=-1,5mA, V_{CC}=3V$	4,2 2,4			V
		$I_{OH}=-3mA, V_{CC}=5V$ $I_{OH}=-1,5mA, V_{CC}=3V$ (10/11/12)	4,3 2,3			V

Sym-bol	Parametr	warunki pomiaru	Min	Typ	Max	J.M.
I_{IL}	Prąd wyjścia w stanie niskim (dotyczy portu B)	$V_{CC}=6V$ wyprowadzenie=L			8	μA
I_{IH}	Prąd wyjścia w stanie wysokim (dotyczy portu B)	$V_{CC}=6V$ wyprowadzenie=H			8	μA
R_{RST}	Rezystancja podciągająca do stanu wysokiego wejście RESET		100		500	$K\Omega$
R_{VO}	Rezystancja podciągająca do stanu wysokiego linie we/wy		30		150	$K\Omega$
		(10/11/12)	35		122	$K\Omega$
I_{CC}	Pobór prądu ze źródła zasilania (2343/22)	Tryb aktywności 3V, 4MHz			3	mA
		Tryb Idle 3V, 4MHz			1,1	mA
		WDT on, 3V, 4MHz			25	μA
		WDT off, 3V, 4 MHz			20	μA
	Pobór prądu ze źródła zasilania (2323)	Tryb aktywności 3V, 4MHz			4	mA
		Tryb Idle 3V, 4MHz		1	1,2	mA
		WDT on, 3V, 4MHz		9	15	μA
		WDT off, 3V, 4 MHz		<1	2	μA
	Pobór prądu ze źródła zasilania (10/11/12)	Tryb aktywności 3V, 4MHz			3	mA
		Tryb Idle 3V, 4MHz		1	1,2	mA
		WDT on, 3V, 4MHz		9	15	μA
		WDT off, 3V, 4 MHz		<1	2	μA
V_{ACIO}	Rozdzielczość komparatora analogowego	$V_{CC}=5V$			40	mV
I_{ACK}	Prąd wejścia komparatora analogowego	$V_{CC}=5V, V_{IN}=V_{CC}/2$	-50		50	nA
t_{ACPD}	Opóźnienie komparatora analogowego	$V_{CC}=2.7V$		750		ns
		$V_{CC}=4.0V$		500		ns

Wykres 1. Przebieg sygnału oscylatora zewnętrznego

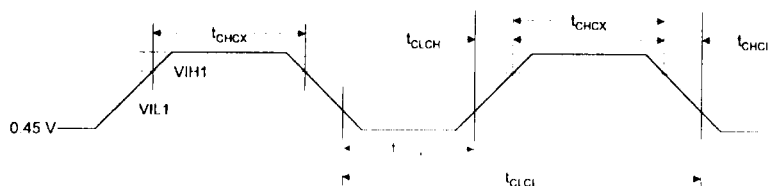


Tabela 2, 3. Parametry czasowe sygnału oscylatora zewnętrznego

Symbol	Parametr	Vcc=2,7 do 4,0V		Vcc=4,0 do 6,0V		J.M.
		Min	Max	Min	Max	
AT90S2323/2343, ATTiny22						
1/tCLCL	Częstotliwość oscylatora	0	4	0	10	MHz
tCLCL	Okres zegara	250		100		ns
tCHCX	Czas trwania stanu H	100		40		ns
tCLCX	Czas trwania stanu L	100		40		ns
tCLCH	Czas trwania zbocza narastającego		1,6		0,5	ns
tCHCL	Czas trwania zbocza opadającego		1,6		0,5	ns

Symbol	Parametr	Vcc=1,8 do 2,7V		Vcc=2,7 do 4,0V		Vcc=4,0 do 5,5V		J.M.
		Min	Max	Min	Max	Min	Max	
ATTiny 10/11/12/15								
1/tCLCL	Częstotliwość oscylatora	0	1	0	4	0	8	MHz
tCLCL	Okres zegara	1000		250		125		ns
tCHCX	Czas trwania stanu H	400		100		50		ns
tCLCX	Czas trwania stanu L	400		100		50		ns
tCLCH	Czas trwania zbocza narastającego		1,6		1,6		0,5	ns
tCHCL	Czas trwania zbocza opadającego		1,6		1,6		0,5	ns

Budowa mikrokontrolera

Charakterystyka jednostki centralnej

Mikrokontrolery AVR oparte są o jednostkę centralną, o zredukowanej liście instrukcji (RISC). Jest ona bardzo dobrze opracowana pod względem szybkości pracy. Sprzyja temu szesnastobitowe słowo rozkazu, powiązane ze wszystkimi niezbędnymi danymi zawartymi w samym kodzie rozkazu. Kolejnym rozwiązaniem przyspieszającym operacje wykonywane przez mikrokontroler, jest wykorzystanie mechanizmów przetwarzania potokowego (pipeline) – równoczesnego pobierania kolejnego rozkazu, z wykonaniem poprzedzającej go instrukcji. Efektem takiego opracowania jest wykonywanie jednej instrukcji na cykl zegarowy (W przypadku zmiany adresu-wykonania skoku następuje zburzenie kolejki i rozkaz trwa 2 cykle). Oprócz udogodnień typowo sprzętowych, na uwagę zasługuje brak sprzętowo zdefiniowanej sztywno funkcji akumulatora. Funkcję tę może pełnić dowolny z 32 rejestrów roboczych. Zastosowana arytmetyka typu rejestr-rejestr sprzyja eliminacji wielu przesłań przy zadaniach o małym i średnim stopniu złożoności, wymagających równoczesnych operacji na znacznej ilości rejestrów. Trzy (typowo kontroler posiada do dwóch) rejestry indeksowe ułatwiają implementację języków wysokiego poziomu, jak i sprzyjają eliminacji kolejnych przesłań.

Budowa blokowa

Mikrokontrolery rodziny AVR w ogólnym rozrachunku składają się z prawie identycznych jednostek centralnych i pewnego zestawu urządzeń zewnętrznych. To właśnie zestaw urządzeń zewnętrznych decyduje o możliwościach zastosowania kontrolerów. W przypadku jednostek centralnych przedstawianych mikrokontrolerów, mamy do czynienia z 90- i 118-rozkazową. Między sobą różnią się one ilością rozpoznawanych instrukcji i właściwościami elektrycznymi. Jeśli zaś mowa o urządzeniach peryferyjnych kontrolerów, to dla wyraźnego zobrazowania zostały one umieszczone wraz z podstawowymi informacjami w tabeli 4.

Tabela 4. Zestawienie podstawowych właściwości mikrokontrolerów

Blok lub funkcja kontrolera	Typ kontrolera															
	S2323	LS2323	S2343	LS2343	ATtiny10	ATtiny10L	ATtiny11	ATtiny11L	ATtiny12	ATtiny12L	ATtiny12V	ATtiny15	ATtiny15L	ATtiny22	ATtiny22L	
Liczba rozkazów	118	118	118	118	90	90	90	90	90	90	90	90	90	118	118	
Liczba linii we/wy	3	3	5	5	6	6	6	6	6	6	6	6	6	5	5	
Xtal max (MHz)	10	4	10	4	6	2	6	2	8	4	1	1,6	1,6	8	4	
Napięcie zasilania min (V)	4,0	2,7	4,0	2,7	4,0	2,7	4,0	2,7	4,0	2,7	1,8	4,0	2,7	4,0	2,7	
Napięcie zasilania max (V)	6,0	6,0	6,0	6,0	5,5	5,5	5,5	5,5	5,5	5,5	5,5	5,5	5,5	6,0	6,0	
Oscylator wewnętrzny RC	N	N	T	T	T	T	T	T	T	T	T	T	T	T	T	
Ścieżka w obszarze danych	T	T	T	T	N	N	N	N	N	N	N	N	N	T	T	
Pamięć programu (KB)	2	2	2	2	1	1	1	1	1	1	1	1	1	2	2	
Pamięć danych (B)	128	128	128	128	0	0	0	0	0	0	0	0	0	128	128	
Pamięć EEPROM (B)	128	128	128	128	0	0	64	64	64	64	64	64	64	128	128	
Zegar/licznik 8 bit TC0	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	
Zegar/licznik 8 bit TC1	N	N	N	N	N	N	N	N	N	N	N	T	T	N	N	
Rejestry indeksowe	3	3	3	3	1	1	1	1	1	1	1	1	1	3	3	
Przetwornik A/C 10 bit	N	N	N	N	N	N	N	N	N	N	N	T	T	N	N	
Komparator	N	N	N	N	T	T	T	T	T	T	T	T	T	N	N	

Rejestr stanu mikrokontrolera (STATUS REGISTER)

Z punktu widzenia programisty, jest to najważniejszy rejestr mikrokontrolera. Jako jedyny rejestr używany przez programistę jest ściśle związany z jednostką centralną. Pozwala on na określenie stanu mikrokontrolera po wykonaniu rozkazu i podjęcie decyzji co do kolejnych rozkazów. Jego bity zostały opisane poniżej.

SREG \$03F

- B7** **I** – bit zezwolenia globalnego na przerwanie. Ustawienie go w stan wysoki zezwala na przerwanie. Jest kasowany po przyjęciu obsługi przerwania, a ustawiany na jej zakończenie.
- B6** **T** – bit zachowania kopii dla instrukcji BLD i BST. Spełnia funkcję rejestru-bitu pośredniego, przy przesłaniach bitów (patrz rozkazy BST, BLD).
- B5** **H** – wskaźnik przeniesienia pomocniczego. Jest to bit tzw. przeniesienia połówkowego. Jego ustawienie następuje w momencie przeniesienia z jednej połówki bajtu do drugiej, w trakcie wykonania operacji arytmetycznej lub logicznej. Pozwala on na prostą realizację arytmetyki BCD.

- B4** **S** – bit znaku wyliczany poprzez funkcję EX-OR z bitów N i V. Sposób obliczenia wartości tego bitu wynika z właściwości liczb zapisywanych w kodzie U2. Wartość zero pozostaje jak dotychczas zapisywana w postaci 8 bitów równych 0. Liczby dodatnie liczymy tak jak w przypadku liczb binarnych bez znaku. Jednak liczby ujemne zmieniają dosyć znacznie zachowanie, gdyż stanowią uzupełnienie w stosunku do 0. Przykładowo: $-1 (U2) = 255(BIN)$, $-2(U2) = 254(BIN)$ itd. Jednakże najstarszy bit w rejestrze jest znakiem liczby. W wyniku operacji bit B7 wyniku jest przepisywany do wskaźnika N. Wskaźnik V jest ustawiany w wyniku wykrycia przekroczenia zakresu U2 tj.: wykrycia przepełnienia liczby dodatniej lub pożyczki w przypadku przekroczenia zakresu liczby ujemnej.
- B3** **V** – przepełnienie uzupełnienia do dwóch. Ustawiany w momencie przekroczenia wartości (pojemności) rejestru, na którym wykonano operację arytmetyczną.
- B2** **N** – wskaźnik wartości ujemnej ustawiany, gdy wynik operacji arytmetycznej lub logicznej przyjmuje wartość ujemną.
- B1** **Z** – wskaźnik zera. Ustawiany, gdy w wyniku operacji arytmetycznej lub logicznej wynik jest równy 0.
- B0** **C** – wskaźnik przeniesienia. Ustawiany w wyniku operacji arytmetycznych lub logicznych.

Oscylator

Mikrokontrolery posiadają wbudowany oscylator synchronizujący pracę wszystkich pozostałych bloków funkcjonalnych. Na zewnątrz mikrokontrolera użytkownik ma do swojej dyspozycji dwa wyprowadzenia oznaczone jako XTAL1 i XTAL2 lub jedno opisane jako CLOCK. Pozwalają one na dołączenie zewnętrznych elementów stabilizujących częstotliwość oscylatora, ewentualnie zewnętrznego generatora taktującego pracę mikrokontrolera. Jako element stabilizujący możemy używać nie tylko rezonator kwarcowy, ale również rezonator ceramiczny. Oscylator współpracujący z elementami zewnętrznymi to nic innego, jak wzmacniacz odwracający fazę. Schemat dołączenia zewnętrznych elementów stabilizujących częstotliwość lub zewnętrznego zegara, przedstawiono na schemacie 1.

poprzez zmianę zawartości rejestru OSCCAL (\$31). Zapisanie do rejestru OSCCAL wartości \$0FF, powoduje ustawienie maksymalnej możliwej częstotliwości. Kontroler ATtiny15 nie współpracuje z zewnętrznymi elementami stabilizującymi częstotliwość. Zawsze pracuje taktowany wewnętrznym oscylatorem o $F=1,6\text{MHz}$.

OSCCAL

B7-B0

\$31

CAL7-CAL0 tylko ATtiny15/22. Bity kalibracji częstotliwości oscylatora taktującego mikrokontroler.

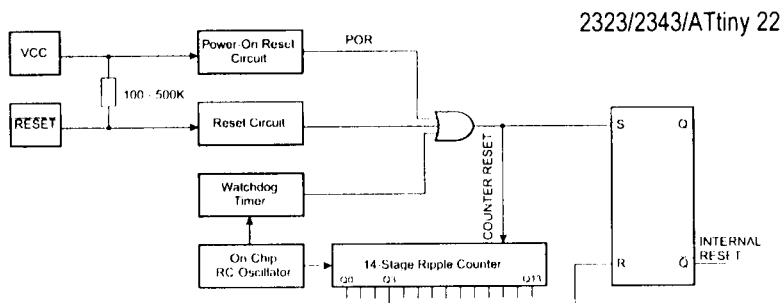
Standardowo kontrolery rozpoczynają pracę z zewnętrznym oscylatorem, jeśli nie ustawimy inaczej bitów konfiguracyjnych w trakcie programowania.

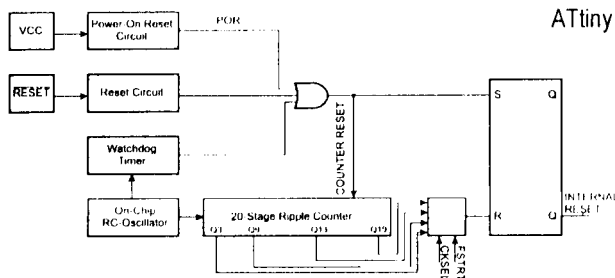
Układ taktowania i sterowania

Jest to dość złożony układ. Na podstawie sygnału z wybranego oscylatora zapewnia taktowanie pozostałych bloków mikrokontrolera. Zawartość rejestru sterowania mikrokontrolera decyduje, jakie sygnały są dostarczane podczas programowania (wybór trybu programowania).

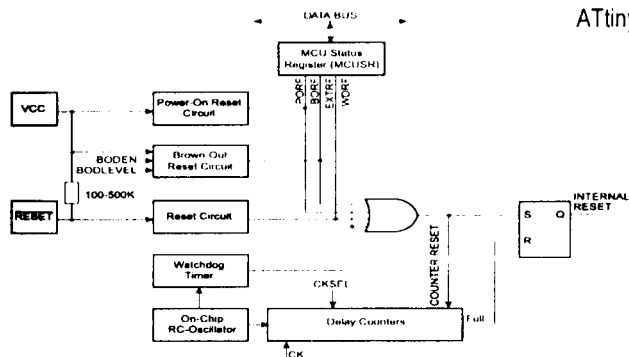
Oprócz tego na podstawie zawartości rejestru sterowania wybierany jest oscylator, taktujący mikrokontroler. Kolejną ważną funkcją jest wytworzenie sygnału RESET o odpowiednich parametrach, aby zapewnić poprawny start mikrokontrolera. Schemat 2 przedstawia układy RESET-u mikrokontrolerów ośmiowyprowadzeniowych.

Schemat 2. Układy RESET mikrokontrolera

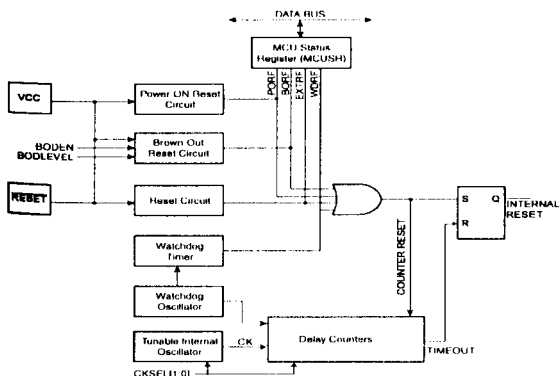




ATtiny 12



ATtiny 15



Ogólnie w układach reset-u możemy wyróżnić blok wytwarzający sygnał reset w momencie załączenia zasilania POR (power-on reset), układ formowania sygnału z wyprowadzenia RESET, blok watchdog ti-

mera oraz blok wytwarzania impulsu reset, składający się z 14-stopniowego licznika i przerzutnika R-S. W niektórych kontrolerach dodatkowo występuje układ nadzoru napięcia zasilającego BODLEVEL. Reset powoduje ustawienie wszystkich rejestrów w obszarze adresowania wejścia/wyjścia wartościami inicjującymi, a następnie rozpoczęcie wykonywania programu od adresu \$000. W kontrolerach ATtiny 10/11/12 pomiędzy licznikiem, a przerzutnikami dodatkowo znajduje się multiplekser, sterowany bitami konfiguracyjnymi mikrokontrolera. Umożliwia on wybranie tzw. szybkiego startu mikrokontrolera. Umożliwia to skrócenie wewnętrznego impulsu reset (patrz tabela 6). Tabela 6 przedstawia parametry elektryczne dla układu RESET.

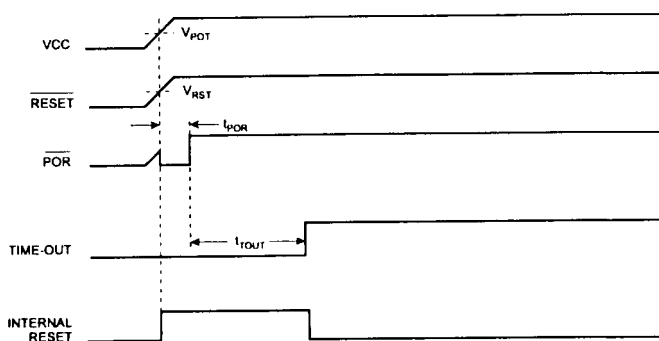
Tabela 6. Parametry elektryczne układu RESET

Symbol	Parametr	BOD	V _{cc} (V)	min	typ	max	J.M.
V _{POT}	Napięcie wyzwalania układu RESET po włączeniu zasilania		5,0	1,0	1,4	1,8	V
	Napięcie wyzwalania układu RESET po włączeniu zasilania (22)	1	5,0	1,0	1,4	1,8	V
		0	5,0	0,6	1,2	1,8	V
	Napięcie wyzwalania układu RESET po wyłączeniu zasilania		5,0	0,4	0,6	0,8	V
	Napięcie wyzwalania układu RESET po wyłączeniu zasilania (22)	1	5,0	0,4	0,6	0,8	V
		0	5,0	0,6	1,2	1,8	V
V _{BOD}	Napięcie zasilania układu BOD	0		2,6	2,7	2,8	V
		1		1,7	1,8	1,9	V
V _{RST}	Napięcie na wejściu reset wyzwalające układ RESET		5,0		0,6V _{cc}		V
t _{ROUT}	Opóźnienie od wyzwolenia do końca zerowania, FSTRT niezaprogramowany		5,0	11	16	21	ms
t _{ROUT}	Opóźnienie od wyzwolenia do końca zerowania, FSTRT zaprogramowany		5,0	1,0	1,1	1,2	ms
t _{ROUT}	Zewnętrzny rezonator kwarcowy lub ceramiczny, FSTRT niezaprogramowany (10/11)		2,7		67		ms
t _{ROUT}	Zewnętrzny rezonator kwarcowy lub ceramiczny, FSTRT zaprogramowany (10/11)		2,7		4,2		ms
t _{ROUT}	Zewnętrzny rezonator kwarcowy lub ceramiczny (10/11)		2,7		4,2		s
t _{ROUT}	Zewnętrzny lub wewnętrzny oscylator RC, FSTRT niezaprogramowany (10/11)		2,7		4,2		ms
t _{ROUT}	Zewnętrzny lub wewnętrzny oscylator RC, FSTRT zaprogramowany (10/11)		2,7		4,2		ms

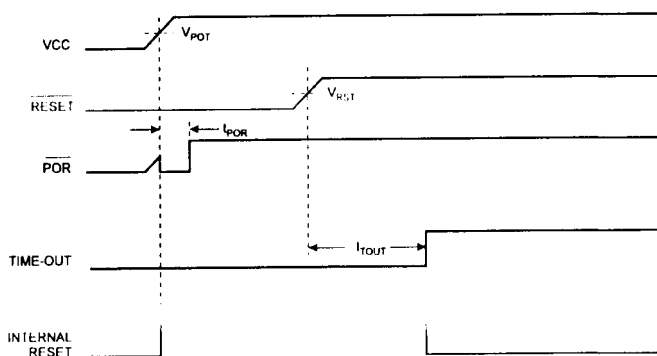
Symbol	Parametr	BOD	V _{CC} (V)	min	typ	max	J.M.
t _{ROUT}	Zewnętrzny zegar, FSTRT niezaprogramowany (10/11)		2,7		4,2		ms
t _{ROUT}	Zewnętrzny zegar, FSTRT zaprogramowany (10/11)		2,7	5 cykli z reset, 2 cykle z power down			
t _{ROUT}	CKSEL(3..0)=1111 (12)	1	1,8		1K CK		
		0	2,7		1K CK		
t _{ROUT}	CKSEL(3..0)=1110 (12)	1	1,8		3,6+1K CK		ms
		0	2,7		4,2+1K CK		ms
t _{ROUT}	CKSEL(3..0)=1101 (12)	1	1,8		57+1K CK		ms
		0	2,7		67+1K CK		ms
t _{ROUT}	CKSEL(3..0)=1100 (12)	1	1,8		16K CK		
		0	2,7		16K CK		
t _{ROUT}	CKSEL(3..0)=1011 (12)	1	1,8		3,6+16K CK		ms
		0	2,7		4,2+16K CK		ms
t _{ROUT}	CKSEL(3..0)=1010 (12)	1	1,8		57+16K CK		ms
		0	2,7		67+16K CK		ms
t _{ROUT}	CKSEL(3..0)=1001 (12)	1	1,8		57+1K CK		ms
		0	2,7		67+1K CK		ms
t _{ROUT}	CKSEL(3..0)=1000 (12)	1	1,8		57+32K CK		ms
		0	2,7		67+32K CK		ms
t _{ROUT}	CKSEL(3..0)=0111 (12)	1	1,8		6 CK		
		0	2,7		6 CK		
t _{ROUT}	CKSEL(3..0)=0110 (12)	1	1,8		3,6+6 CK		ms
		0	2,7		4,2+6 CK		ms
t _{ROUT}	CKSEL(3..0)=0101 (12)	1	1,8		57+6 CK		ms
		0	2,7		67+6 CK		ms
t _{ROUT}	CKSEL(3..0)=0100 (12)	1	1,8		6 CK		ms
		0	2,7		6 CK		ms
t _{ROUT}	CKSEL(3..0)=0011 (12)	1	1,8		3,6+6 CK		ms
		0	2,7		4,2+6 CK		ms
t _{ROUT}	CKSEL(3..0)=0010 (12)	1	1,8		57+6 CK		ms
		0	2,7		67+6 CK		ms
t _{ROUT}	CKSEL(3..0)=0001 (12)	1	1,8		6 CK		ms
		0	2,7		6 CK		ms
t _{ROUT}	CKSEL(3..0)=0000 (12)	1	1,8		3,6+6 CK		ms
		0	2,7		4,2+6 CK		ms
t _{ROUT}	CKSEL(1..0)=11 (15)	0	2,7		128+18 CK		μs
		0	5,0		32+18 CK		μs
t _{ROUT}	CKSEL(1..0)=11 (15)	1	2,7		16+18 CK		ms
		1	5,0		4+18 CK		ms
t _{ROUT}	CKSEL(1..0)=11 (15)	1	2,7		256+18 CK		ms
		1	5,0		64+18 CK		ms
t _{ROUT}	CKSEL(1..0)=11 (15)	1	2,7		256+1K CK		ms
		1	5,0		64+1K CK		ms
t _{ROUT}	Opóźnienie od wyzwolenia do końca zerowania (2343)		5,0	11	16	21	ms

Układ RESET nie pracuje poniżej napięcia zasilania układu reset – V_{POT} . W momencie wystąpienia jednego z warunków powodujących generację sygnału RESET, zostaje ustawiony przerzutnik RS (początek impulsu wewnętrznego RESET-u) i wyzerowany licznik układu RESET. Licznik ten jest taktowany przez wewnętrzny generator RC. Po odliczeniu ilości taktów zegara, powodującej przepełnienie licznika, zostaje wyzerowany przerzutnik RS (zakończenie wewnętrznego impulsu RESET). Wykres 2 przedstawia przebiegi wyzwalania układu RESET.

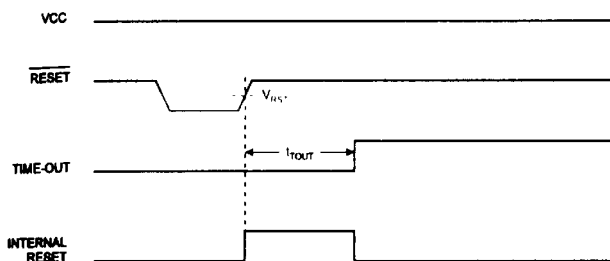
Wykres 2. Przebiegi wyzwalania układu RESET przy załączeniu napięcia zasilającego



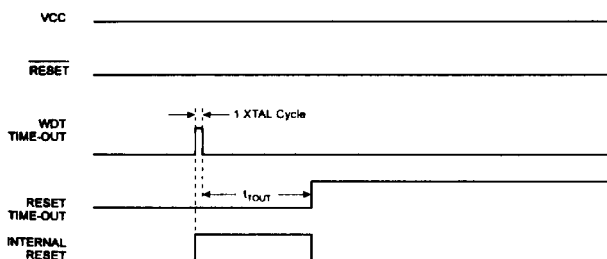
Przebiegi wyzwalania układu RESET po załączeniu zasilania i podtrzymanie wejścia RESET w stanie niskim.



Przebiegi wyzwalania układu RESET poprzez generację impulsu do stanu niskiego na wejściu RESET.



Przebiegi wyzwalania układu RESET, spowodowanego przepelnieniem licznika watchdog.



Wystąpienie sygnału reset powoduje w zależności od przyczyny jego powstania, ustawienie różnych flag informujących o przyczynie. Flagi te są umieszczone w rejestrze MCUSR przedstawionym poniżej.

MCUSR

\$34

B3

WDRF – dotyczy tylko ATtiny10/11/12/15. Bit przyjmuje wartość 1 dla reset-u spowodowanego przez watchdog

B2

BORF – dotyczy tylko ATtiny10/11/12/15. Bit przyjmuje wartość 1 dla reset-u spowodowanego przez układ Brown Out Detection

B1

EXTRF – bit przyjmuje wartość 1 dla reset-u spowodowanego zewnętrznym sygnałem

B0

PORF – bit przyjmuje wartość 1 dla reset-u spowodowanego włączeniem zasilania.

Po wystąpieniu wewnętrznego sygnału RESET, niezależnie od źródła powodującego wytworzenie go, następuje rozpoczęcie wykonywania programu począwszy od adresu \$000.

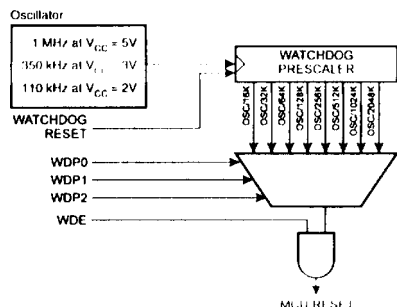
BOD (Brown Out Detection)

Układ ten odpowiada za kontrolę poziomu zasilania jednostki centralnej kontrolera. Jest to zabezpieczenie przed zejściem poniżej poziomu napięcia zapewniającego przetwarzanie, bez powstawania przekłamań. Układ ten możemy załączyć przez zaprogramowanie bitu BODEN w trakcie programowania kontrolera. Po zaprogramowaniu bitu BODLEVEL dolny próg napięcia zasilającego, kontrolowany przez układ BOD, nie powodującego wywołania procesu restartu kontrolera, zostaje przesunięty z 2,7 V na poziom 4,0 V dla ATtiny22. W przypadku kontrolera ATtiny12 następuje przesunięcie poziomu tego napięcia wyzwalania z 1,8 V do 2,7 V. Zapewnia to wymuszenie napięcia wyższego, niż kontroler wymaga do poprawnej pracy. Histereza wejścia wyzwalania układu BOD wynosi 50 mV. W przypadku zejścia poniżej granicznej wartości napięcia rozpoczynany jest proces restartu.

Watchdog

W rodzinie kontrolerów AVR występuje wewnętrzny układ watchdog umieszczony na jednej strukturze z mikrokontrolerem. Schemat 3 przedstawia schemat układu watchdog.

Schemat 3. Układ watchdog



Układ watchdog składa się z licznika zliczającego cykle zegarowe wewnętrznego oscylatora RC. Producent podaje, iż przy napięciu zasilania $V_{cc}=5\text{ V}$ częstotliwość ta wynosi około 1 MHz ($3,0\text{ V}$ 350 KHz). Stopień podziału licznika od $OSC/16$ do $OSC/2048$ wybieramy za pomocą multipleksera, sterowanego bitami $WDP0$ - $WDP2$, umieszczonymi w rejestrze $WDTCSR$ (§21). Częstotliwość z wybranego stopnia podziału trafia na bramkę zezwolenia (AND), sterowaną bitem WDE rejestru $WDRSCR$. Jeśli w trakcie pracy układu zmienimy ustawiony czas na mniejszy bez wcześniejszego wyzerowania licznika, to może nastąpić niekontrolowany reset mikrokontrolera. W związku z tym, aby nie nastąpił taki efekt uboczny należy zatrzymać licznik, a przynajmniej wyzerować jego zawartość przed dokonaniem zmiany. Opis poszczególnych bitów rejestru $WDRSCR$ przedstawiony jest poniżej.

WDTCSR**§21****B4**

WDTOE – bit zabezpiecza przed przypadkowym wyłączeniem układu czuwającego.

Aby skasować bit WDE należy w pierwszej kolejności ustawić bit $WDTOE$. Odbywa się to przez równoczesny zapis jedynki do bitów $WDTOE$ i WDE . W przeciwnym wypadku nie uda się wyzerować bitu WDE . Bit ten jest zerowany automatycznie po czterech cyklach zegarowych, od momentu jego ustawienia w stan „1”. W ciągu tych czterech cykli musimy wyzerować WDE .

B3

WDE – bit sterujący zezwoleniem pracy układu watchdog

$B3=0$ stop

$B3=1$ start

Zerowanie bitu WDE patrz bit $WDTOE$

B2-B0

WDP2, WDP1, WDP0

0	0	0	16 cykli
0	0	1	32 cykle
0	1	0	64 cykle
0	1	1	128 cykli
1	0	0	256 cykli
1	0	1	512 cykli
1	1	0	1024 cykle
1	1	1	2048 cykli

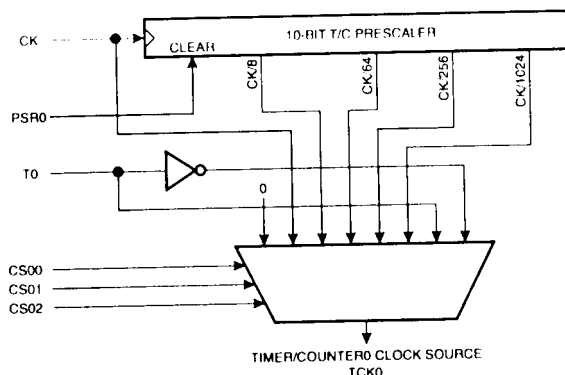
Jak widać z opisu najmniejszy okres sygnału licznika wynosi około 16 mikrosekund . Przy największym stopniu podziału, uzyskujemy okres rzę-

du 2 milisekund. Zerowania licznika dokonujemy za pomocą instrukcji WDR. W przypadku gdy program z jakichś przyczyn nie zdąży wyzerować licznika, sygnał z układu licznika powoduje wywołanie sprzętowego zerowania mikrokontrolera RESET. Wykresy czasowe w przypadku wystąpienia przepełnienia licznika watchdog, zostały przedstawione wraz z opisem układu taktowania i sterowania układu RESET mikrokontrolera.

Preskaler dla zegara licznika TC0

Preskaler o budowie podobnej do preskalera układu watchdog, rozszerzonego o dodatkowy multiplekser dla zegara/licznika TC0, stanowi wstępny dzielnik dla układu zegara/licznika TC0.

Schemat 4. Preskaler dla zegarów/liczników TC0



Uwaga: bit PSR0 – zerowanie licznika preskalera dotyczy tylko kontrolera ATtiny15.

Układ zegara licznika możemy taktować dwoma rodzajami sygnału zegarowego. Pierwszy z nich to sygnał CK aktywnego oscylatora (taktuującego mikrokontroler). Drugi, to dowolny sygnał zewnętrzny dołączony do wyprowadzenia T0 dla licznika TC0, oznaczony na schemacie jako EXT0. W przypadku licznika TC0 wyboru sygnału, stopnia podziału, jak i zbocza aktywnego dokonujemy za pomocą bitów CS00-CS02. Bity te znajdują się w rejestrze TCCR0. Wejściowy sygnał z

garowy jest synchronizowany z zegarem, taktującym mikrokontroler. Próbkowanie sygnału wejściowego odbywa się podczas zbocza wznoszącego wewnętrznego zegara. Jedynie kontroler ATtiny15 posiada dodatkowy bit, umożliwiający skasowanie zawartości dzielnika układu preskalera. Opis poszczególnych bitów i ich kombinacji sterujących preskalerem dla zegarów/liczników wygląda następująco:

TCCR0				\$33
B2-B0	CS02,	CS01,	CS00	
0	0	0	0	stop, zegar/licznik0 wyłączony
0	0	1	1	F TC0=CK
0	1	0	0	F TC0=CK/8
0	1	1	1	F TC0=CK/64
1	0	0	0	F TC0=CK/256
1	0	1	1	F TC0=CK/1024
1	1	0	0	F TC0=EXT0, reakcja na zbocze opadające
1	1	1	1	F TC0=EXT0, reakcja na zbocze narastające.

SFIOR **\$2C**
B0 **PSR0** – dotyczy tylko ATtiny15. Bit kasowania dzielnika układu preskalera dla TC0. Bit ustawiony w stan wysoki powoduje wyzerowanie dzielnika preskalera

Preskaler zapewnia wstępny podział częstotliwości, wybranego źródła sygnału. Wybrane źródło sygnału jest przyłączane przez multiplexer i układ synchronizujący z zegarem mikrokontrolera do zegara/licznika. Warunkiem synchronizacji jest, aby na czas między dwoma przejściami sygnału zewnętrznego przypadł przynajmniej jeden okres sygnału, taktującego mikrokontroler CK. Spowodowane jest to tym, iż próbkowanie sygnału wejściowego, odbywa się podczas zbocza narastającego sygnału CK.

Układ zegara/licznika (TIMER/COUNTER) TC0

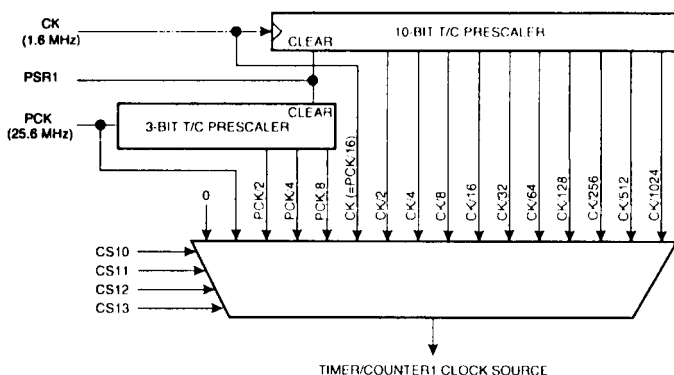
Układ zegara/licznika TC0 stanowi ośmiobitowy licznik, którego zawartość może być odczytywana, lub zmieniana poprzez dostęp do rejestru TCNT0 (\$032). Schemat 5 przedstawia budowę układu TC0 wraz z rejestrami, sterującymi jego pracą.

programu pod adresem \$006-ATtiny15, \$002-AT90S2323/43;ATtiny22, \$003-ATtiny10/11/12

Prescaler dla zegara licznika TC1

Ośmiobitowy zegar TC1 występuje tylko w kontrolerze ATtiny 15. Różni się on w stosunku do całej rodziny kontrolerów AVR. Również budowa preskalera jest odmienna. Została ona przedstawiona na schemacie 6.

Schemat 6. Preskaler dla TC1



Źródłem sygnału dla zegara TC1 jest wewnętrzny oscylator o częstotliwości około 1,6 MHz (CK) z powieleniem za pomocą PLL do 25,6 MHz (PCK). Częstotliwość silnie zależy od poziomu skalibrowania generatora podstawowego 1,6 MHz. Należy pamiętać, iż licznik TC1 przestanie spełniać swoje zadanie dla najwyższej częstotliwości taktującej, jeśli częstotliwość generatora podstawowego przekroczy 1,75 MHz. Wyboru źródła sygnału taktującego licznik, dokonujemy za pomocą bitów CS10..CS13, znajdujących się w rejestrze TCCR1 (\$30). Również ten preskaler może zostać wyzerowany. Dokonujemy tego poprzez ustawienie w stan wysoki bitu PSR1 w rejestrze SFIOR.

TCNT1
B7-B0

\$2F

Licznik/zegar TC1

TCCR1**\$30****B3-B0 CS13, CS12, CS11, CS10**

0	0	0	0	stop, zegar/licznik1 wyłączony
0	0	0	1	F TC1=PCK
0	0	1	0	F TC1=PCK/2
0	0	1	1	F TC1=PCK/4
0	1	0	0	F TC1=PCK/8
0	1	0	1	F TC1=CK=PCK/16
0	1	1	0	F TC1=CK/2
0	1	1	1	F TC1=CK/4
1	0	0	0	F TC1=CK/8
1	0	0	1	F TC1=CK/16
1	0	1	0	F TC1=CK/32
1	0	1	1	F TC1=CK/64
1	1	0	0	F TC1=CK/128
1	1	0	1	F TC1=CK/256
1	1	1	0	F TC1=CK/512
1	1	1	1	F TC1=CK/1024

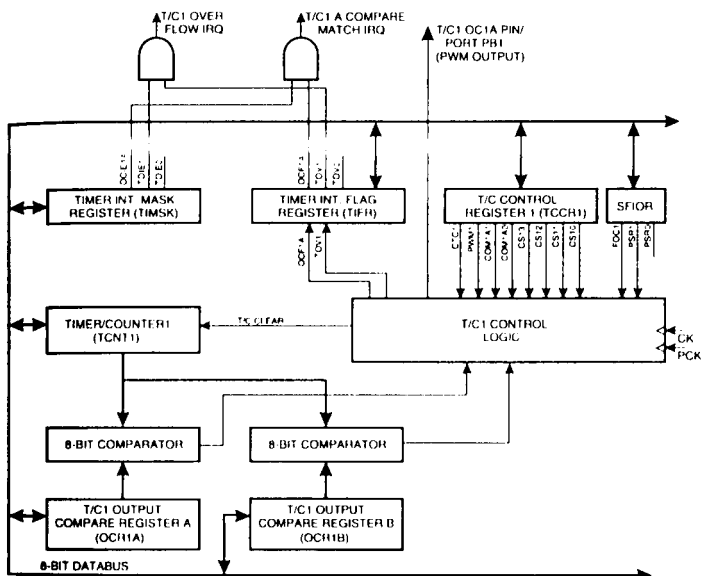
SFIOR**\$2C**

B1 PSR1 – bit kasowania dzielnika układu preskalera dla TC1. Bit ustawiony w stan wysoki powoduje wyzerowanie preskalera.

Układ zegara/licznika (TIMER/COUNTER 1) TC1

Ośmiobitowy licznik TC1 występuje tylko w kontrolerze ATtiny15. Posiada on wbudowane wsparcie dla generacji przebiegów PWM. Sygnał wejściowy dla licznika pobierany jest z zegara mikrokontrolera poprzez preskaler. Licznik TC1 może pracować w trybach: 8-bitowego licznika o możliwościach takich jak TC0, lub jako ośmiobitowy generator PWM o dużej dokładności częstotliwości. Licznik może być taktowany maksymalną częstotliwością 25,6 MHz. Przy dużych częstotliwościach działa jako autonomiczny generator PWM, o zadanych parametrach. Układ licznika może zgłaszać dwa rodzaje przerwań. Są to przerwania spowodowane przez przepełnienie licznika lub osiągnięcie wartości, zadanej przez licznik. Na schemacie 7.2 przedstawiona jest budowa licznika.

Schemat 7. Zegar/licznik TC1



W jego budowie możemy wyróżnić rejestry: maski przerw (TIMSK), znaczników zgłoszenia przerw (TIFR), sterujący pracą licznika (SFOR), licznika (TCNT1) i rejestry porównania (OCR1A, OCR1B). Wyróżniamy prócz tego blok logiki, sterującej pracą licznika (CONTROL LOGIC) oraz dwa komparatory (8-bit COMPARATOR). Wyjaśnienia wymagają dwa równocześnie występujące komparatory. Komparator dla rejestru OCR1A jest wykorzystywany standardowo do określania szerokości impulsu PWM. Przeznaczeniem dla rejestru OCR1B jest w rozszerzonym trybie PWM wyzerowanie licznika, po osiągnięciu zadanej wartości. Po osiągnięciu wartości zadanej w rejestrze OCR1B następuje w kolejnym cyklu zegarowym wpisanie do licznika wartości \$00. Umożliwia to dokładniejszą regulację częstotliwości w trybie generacji PWM. Jednak ta regulacja odbywa się kosztem rozdzielczości. Sam licznik zlicza od wartości \$00 do wartości \$FF, lub do wartości określonej w rejestrze OCR1B. Aby wartość załadowana do rejestru OCR1B miała wpływ na zliczanie, (aby ograniczyła najwyższą osiąganą wartość przez licznik), musimy ustawić w stan wysoki bit CTC1 w rejestrze TCCR1.

Tryb zliczania

Tryb ten jest bardzo podobny do trybu zliczania zegara TC0. W tym trybie sygnał wejściowy z multipleksera wyjściowego w preskalerze powoduje inkrementację licznika. W wyniku osiągnięcia maksymalnej wartości licznika, (\$FF lub wartości zadanej w OCR1B), następuje ustawienie flagi przerwania TOIE1 w rejestrze TIMSK. Zezwolenie na przerwanie w wyniku przepełnienia licznika, następuje w wyniku ustawienia w stan wysoki znacznika TOV1 w rejestrze TIFR. Jeśli bit zezwolenia na przerwanie jest ustawiony w wyniku przepełnienia, następuje zgłoszenie przerwania o wektorze nr 5 pod adresem \$004. Dodatkowo do sterowania w tym trybie mamy bit FOC1A, w rejestrze SFIOR. Za pomocą tego bitu możemy wymusić działanie powodujące ustawienie wyjścia, zgodnie z trybem zapisanym na bitach COM1A1 i COM1A0, tak jak by to się odbyło w wyniku wystąpienia warunku porównania. Bit ten w trybie PWM nie jest uwzględniany. Poniżej przedstawione jest umiejscowienie bitów, sterujących obsługą licznika w trybie zliczania.

TCNT1 \$2F

B7-B0 Licznik/zegar TC1

TIFR \$38

B6 **OCF1A** – flaga zgłoszenia przerwania w wyniku porównania rejestru OCR1A z licznikiem

1 licznik osiągnął zadaną wartość

0 wartości różne

B2 **TOV1** – flaga zgłoszenia przerwania w wyniku wystąpienia przepełnienia licznika

1 nastąpiło przepełnienia licznika TC1

0 przepełnienia licznika TC1 nie nastąpiło.

TIMSK \$39

B6 **OCIE1A** – bit zezwolenia na przerwanie w wyniku porównania zawartości licznika z rejestrem OCR1A

1 przerwania dozwolone

0 przerwanie zabronione

B2 **TOIE1** – bit zezwolenia na przerwanie spowodowane przepełnieniem TC1, lub osiągnięcia maksymalnej wartości, zadanej w rejestrze OCR1B.

1 przerwania dozwolone

0 przerwanie zabronione

SFIOR \$2C

B2 **FOC1A** ustawienie bitu wymusza ustawienie wyprowadzenia PB1, zgodnie z trybem, zapisanym na bitach COM1A1

		COM1A0. Ustawienie tego bitu nie powoduje zgłoszenia przerwania, jak to ma miejsce w przypadku porównania.	
B1	PSR1	– bit kasowania dzielnika układu preskalera dla TC1. Bit ustawiony w stan wysoki powoduje wyzerowanie preskalera.	
OCR1B	\$2D		
B7-B0		Rejestr maksymalnego pułapu licznika TC1. Dla bitu CTC=0 nie ma wpływu na licznik TC1.	
TCCR1	\$30		
B7	CTC1	bit załączenia rejestru porównania OCR1B	
	0	rejestr OCR1B nie ma wpływu na licznik TC1, rejestr TCNT1 odlicza do wartości \$FF	
	1	rejestr OCR1B określa maksymalną wartość, jaką może przyjąć rejestr TCNT1	
B6	PWM1	bit załączenia trybu pracy licznika	
	0	tryb licznika	
	1	tryb PWM	
B5, B4	COM1A1, COM1A0	bity sterujące trybem pracy wyprowadzenia PB1 dla trybu porównania rejestru OCR1A z licznikiem, lub sterowania bitem FOC1A.	
	0	0	TC1 nie steruje wyprowadzeniem PB1
	0	1	przełączanie wyjścia na stan przeciwny
	1	0	zerowanie linii PB1 w wyniku porównania
	1	1	ustawienie linii PB1 w wyniku porównania

Tryb generatora PWM

Zegar/licznik TC1 może zostać użyty, jako 8-bitowy generator impulsów o regulowanej szerokości (PWM), z dosyć płynnym ustawianiem generowanej częstotliwości. Bardzo ważny jest sposób sterowania wyprowadzeniem zewnętrznym OC1. Możemy ustawić 3 tryby obsługi tego wyprowadzenia. Wyboru sposobu obsługi dokonujemy za pomocą bitów COM1A1 i COM1A0, umiejscowionych w rejestrze TCCR1. W zależności od wyboru możemy uzyskać na wyjściu OC1 przebieg proporcjonalny do zadanej wartości, odwrotnie proporcjonalny (zanegowany) oraz brak obsługi wyjścia. Na czas dokonywania zmian tych dwóch bitów

w trybie PWM należy zablokować przerwania, aby w trakcie ich zmiany nie wywołać przypadkowo obsługi przerwania. Licznik zgłasza dwa rodzaje przerwań. Pierwsze z nich po osiągnięciu górnej wartości (\$FF lub określonej przez rejestr OCR1B) oraz przerwanie, od porównania stanu licznika z OCR1A. Przepelnienie lub porównanie rejestru OCR1B wywołuje wektor przerwania nr 5, wektor obsługi przerwania pod adresem \$004, zaś porównanie wywołuje przerwanie nr 4 z wektorem pod adresem \$003. Umieszczenie opisanych rejestrów, bitów w rejestrach i wybierane tryby pracy przedstawione są poniżej.

OCR1B \$2D

B7-B0 Rejestr wartości porównywanej z TC1, ustalający górną wartość, jaką może przyjąć licznik. Rejestr jest uwzględniany dla bitu CTC1=1

OCR1A \$2E

B7-B0 Rejestr wartości porównywanej z TC1, sterujący szerokością impulsów PWM

TCCR1 \$30

B6 **PWM1** załączenie trybu pracy licznika

0 tryb licznika

1 tryb PWM

B5, B4 **COM1A1, COM1A0**

bity sterujące trybem pracy wyprowadzenia PB1.

0 0

TC1 nie steruje wyprowadzeniem PB1

0 1

TC1 nie steruje wyprowadzeniem PB1

1 0

zerowanie linii PB1 w wyniku porównania

1 1

ustawienie linii PB1 w wyniku porównania

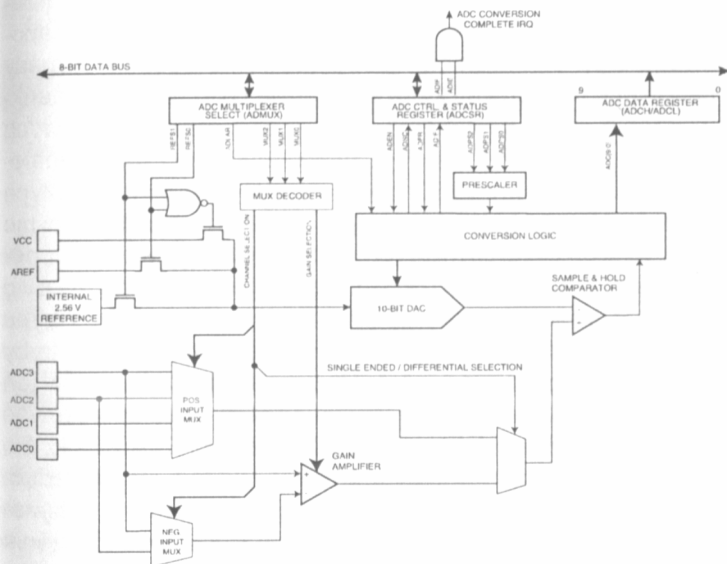
Zmiany wypełnienia przebiegów PWM, dokonujemy poprzez zmianę wartości, zapisanej w rejestrze OCR1A. Licznik pracuje z niezsynchronizowanym zapisem do rejestrów i porównaniem wartości. Oznacza to, iż nowa wartość w rejestrze OCR1A powoduje natychmiastową zmianę na wyjściu porównania.

Układ przetwornika analogowo-cyfrowego

W kontrolerze ATtiny15 jest wbudowany dużej dokładności aproksymacyjny przetwornik A/C, o rozdzielczości 10 bitów z czterema wejściami.

ściami analogowymi. Układ przetwornika składa się z 7-bitowego pre-skalera wyboru częstotliwości przetwarzania, dwóch multiplekserów sygnałów analogowych, 10 bitowego przetwornika C/A, komparatora, wzmacniacza, źródła napięcia odniesienia i logiki sterującej. Schemat 8 przedstawia budowę przetwornika.

Schemat 8. Przetwornik A/C



Załączenie przetwornika następuje po ustawieniu w stan wysoki bitu ADEN, w rejestrze ADCSR (\$06). Przetwornik może po zakończeniu przetwarzania zgłaszać przerwanie. Zezwolenie na przerwanie polega na ustawieniu w stan wysoki bitu ADIE, w rejestrze ADCSR. W wyniku zakończenia przetwarzania, zostaje ustawiona flaga ADIF w tym samym rejestrze. Przetwornik możemy uruchomić w dwóch trybach przetwarzania. Pierwszy z nich to jednokrotne przetwarzanie. Drugim jest przetwarzanie wolnobieżne. Wyboru trybu dokonujemy za pomocą bitu ADFR. Jeśli bit ten ustawimy w stan wysoki, to po zakończeniu bieżącego przetwarzania natychmiast przetwornik rozpocznie kolejne. Wynik dokonanej kwantyzacji zostaje umieszczony w rejestrach ADCH i ADCL. Sam sposób umieszczenia wyniku w tych rejestrach jest zależny od bi-

tu ADLAR, umieszczonego w rejestrze ADMUX (\$07). Po ustawieniu tego bitu w stan wysoki, 8 starszych bitów przetwarzania, zostaje umieszczonych w rejestrze ADCH, zaś dwa najmłodsze zostają zapisane na najstarszych bitach rejestru ADCL. Dla bitu ADLAR równego zero, 8 młodszych bitów wyniku zostaje umieszczone w rejestrze ADCL, a dwa najstarsze bity wyniku konwersji na najmłodszych bitach rejestru ADCH.

W przypadku przetworników A/C bardzo ważnym czynnikiem są zakłócenia od cyfrowej części układu. Podstawowym czynnikiem, zapobiegającym tego typu zakłóceniom, jest stosowanie oddzielnej masy dla części cyfrowej i analogowej, które stykają się tylko w jednym punkcie (wyprowadzenie masy kontrolera). Ścieżki sygnałów analogowych powinny być maksymalnie krótkie i być prowadzone nad masą analogową z dala od ścieżek, przewodzących sygnały cyfrowe o dużych częstotliwościach. Należy pamiętać, aby w trakcie przetwarzania nie zmieniać stanu wyprowadzeń cyfrowych portu, gdyż będzie to powodowało zwiększenie zakłóceń. Jednak sam kontroler został dodatkowo wyposażony w tryb pracy z redukcją szumów. Ustawienie tego trybu powoduje przejście kontrolera w specjalny stan uśpienia (patrz tryb pracy ze zmniejszonym poborem prądu). Aby wprowadzić kontroler w ten stan, musimy ustawić tryb jednokrotnego przetwarzania, odblokować przerwanie od przetwornika ($ADEN=1$, $ADSC=0$, $ADFR=0$, $ADIE=1$). Po ustawieniu odpowiednich bitów możemy wprowadzić tryb „Noise reduction”. Po wprowadzeniu w ten tryb start konwersji następuje automatycznie, a jednostka centralna zostaje wstrzymana, aż do zakończenia konwersji zgłoszenia przerwania zewnętrznego lub wygenerowanie sygnału reset. Maksymalne napięcie wejściowe przetwornika jest określone przez napięcie odniesienia i nie może ono przekroczyć napięcia zasilającego kontroler, zaś dla wewnętrznego źródła odniesienia (2,56 V) zakres napięć wejściowych wynosi 0-2,55 V. Wyboru źródła napięcia odniesienia dla przetwornika dokonujemy za pomocą bitów REFS1 REFS0. Do wyboru mamy napięcie zasilające V_{CC} , wewnętrzne napięcie odniesienia $V_{REF}=2,56\text{ V}$ oraz zewnętrzne napięcie odniesienia A_{REF} , podawane na linię portu PB0. Opcjonalnie można polepszyć parametry wewnętrznego źródła zasilania, przez dołączenie kondensatora między masę, a linię portu PB0. Po odpowiednim skonfigurowaniu bitów REFSx dołączony kondensator będzie filtrował wewnętrzne napięcie odniesienia. Wyboru sygnału wejściowego dla przetwornika dokonujemy za pomocą trzech bitów MUX2 MUX1 MUX0. Pozwalają one na wybranie jednego z czterech wejść ADC0-PB5, ADC1-PB2, ADC2-PB3

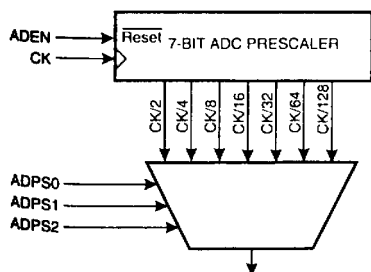
ADC3-PB4. Wyboru podłączonych wejść dokonujemy zgodnie z tabelą 7.

Tabela 7. Konfiguracja wejść przetwornika AC

MUX2..0	Pojedyncze wejście	Wejście nieodwracające	Wejście odwracające	Wzmocnienie
000	ADC0(PB5)	Nie wykorzystywane		
001	ADC1(PB2)			
010	ADC2(PB3)			
011	ADC3(PB4)			
100	Nie wykorzystywane	ADC2(PB3)	ADC2(PB3)	1x
101		ADC2(PB3)	ADC2(PB3)	20x
110		ADC2(PB3)	ADC3(PB4)	1x
111		ADC2(PB3)	ADC3(PB4)	20x

Jak widać, w tabeli możemy wyróżnić dwa tryby konfiguracji wejść dla przetwornika. Pierwszy zapewnia bezpośrednie przetwarzanie sygnału z wybranego wejścia. Drugi tryb umożliwia wykorzystanie wewnętrznego wzmacniacza, do którego na stałe jest przyłączone wyprowadzenie PB3 do wejścia nieodwracającego. Jednak praktyczne wykorzystanie w tym trybie zapewniają tylko ustawienie trybów 6 i 7. Tryby 4 i 5 służą tylko do określenia różnicowego napięcia niezrównoważenia wzmacniacza. Pozwala to na programowe skorygowanie tego błędu dla trybów 6 i 7, przez określenie rozbieżności między wejściami. Wbudowany w przetwornik prescaler o budowie, przedstawionej na schemacie 9, zapewnia taktowanie przetwornika w trakcie konwersji.

Schemat 9. Prescaler przetwornika ADC



Od szybkości taktowania zależy czas konwersji. Jednak dopuszczalne częstotliwości taktujące przetwornik muszą się zawierać

w przedziale 50-200KHz. Typowo czas przetwarzania wynosi 13 cykli zegara. Jednak, aby dokładność była na odpowiednim poziomie, to po załączeniu przetwornika, zmianie konfiguracji wejść i zmianie źródła napięcia odniesienia, konwersja będzie trwała 25 cykli (zastrzeżenie zrealizowane sprzętowo). Przyjmując, iż częstotliwość wyjściowa z pre-skalera wynosi 130 KHz, daje nam to 10 000 konwersji na sekundę (przy 200 KHz ponad 15300 konwersji na sekundę). Wejścia przetwornika charakteryzują się wysoką rezystancją, typowo wynoszącą 100 M Ω . Przy pomiarach z wykorzystaniem pojedynczego wejścia, uzyskujemy rozdzielczość pomiaru rzędu 10 bitów, jednak w przypadku wykorzystania wejść różnicowych maleje ona do 8 bitów. W przypadku taktowania przetwornika częstotliwością 2 MHz (przy $V_{ref}=4V$) błąd wynosi 16 LSB. W efekcie tracimy 4 najmłodsze bity wyniku. Jednak pozostaje nam przetwornik o rozdzielczości 6 bitów i szybkości przekraczającej 153 Kpróbk. Dla taktowania częstotliwością 1 MHz tracimy 4 LSB-2 najmłodsze bity przy 76 K próbkach. Poniżej zostały przedstawione sumarycznie wszystkie bity i rejestry, służące do obsługi przetwornika A/C.

ADMUX

B7, B6

\$07**REFS1, REFS0** – wybór napięcia odniesienia

0	0	$V_{ref} = V_{cc}$
0	1	$V_{ref} = V_{PB0}$
1	0	$V_{ref} = INTV_{ref}$ (2,56V)
1	1	$V_{ref} = INTV_{ref}$ (2,56V) + podłączenie do linii PB0, umożliwiające dodanie kondensatora, filtrującego napięcie odniesienia.

Wprowadzona zmiana nie odniesie skutku, dopóki przetwarzanie nie spowoduje ustawienia flagi ADIF. Pierwsza konwersja po wprowadzeniu i przyjęciu zmiany będzie trwała 25 cykli zegarowych

B5

ADLAR – powoduje zmianę sposobu zwracania danych w rejestrach ADCH, ADCL. Dla znacznika równego „0”, wynik jest równany do najmłodszego bitu (najstarsze bity rejestru ADCH bez znaczenia). Dla bitu równego „1”, równanie wyniku odbywa się do najstarszego bitu rejestru. (Najmłodsze bity rejestru ADCL bez znaczenia).

B2, B1, B0

MUX2, MUX1, MUX0 – wybór dołączonych wejść przetwornika

	wejście		nieodwra- cające	od- wraca- jące	wzmo- cnie nie
0	0	0	ADC0 (PB5)		x1
0	0	1	ADC1 (PB2)		x1
0	1	0	ADC2 (PB3)		x1
0	1	1	ADC3 (PB4)		x1
1	0	0	ADC2 (PB3)	ADC2 (PB3)	x1
1	0	1	ADC2 (PB3)	ADC2 (PB3)	x20
1	1	0	ADC2 (PB3)	ADC3 (PB4)	x1
1	1	1	ADC2 (PB3)	ADC3 (PB4)	x20

ADCSR**\$06**

- B7** **ADEN** – bit załącza przetwornik A/C. Wyzerowanie bitu powoduje natychmiastowe przerwanie przetwarzania.
- B6** **ADSC** – ustawienie tego bitu powoduje start konwersji. Dla trybu pojedynczej konwersji, bit zostaje wyzerowany po jej zakończeniu.
- B5** **ADFR** – bit ustawienia ciągłego (wolnobieżnego) przetwarzania. Dla bitu równego 1 kolejne cykle przetwarzania następują automatycznie. Ciąg konwersji trwa dopóki nie wyzerujemy bitu ADSC lub ADEN.
- B4** **ADIF** – znacznik (flaga) zgłoszenia przerwania. Ustawiany jest po każdym zakończeniu przetwarzania.
- B3** **ADIE** – bit zezwolenia na przerwanie od przetwornika A/C. Jeśli bit jest równy 1, to w wyniku ustawienia flagi ADIF zostaje wywołane wektora przerwania nr 9, pod adresem \$008.
- B2, B1, B0** **ADPS2, ADPS1, ADPS0** – wybór stopnia podziału preskalera dla przetwornika A/C
- | | | | |
|---|---|---|-------------------|
| 0 | 0 | 0 | $F_{ADC} = CK/2$ |
| 0 | 0 | 1 | $F_{ADC} = CK/2$ |
| 0 | 1 | 0 | $F_{ADC} = CK/4$ |
| 0 | 1 | 1 | $F_{ADC} = CK/8$ |
| 1 | 0 | 0 | $F_{ADC} = CK/16$ |
| 1 | 0 | 1 | $F_{ADC} = CK/32$ |

1	1	0	$F_{ADC}=CK/64$
1	1	1	$F_{ADC}=CK/128$

ADCH \$05

B7-B0 Starszy bajt wyniku przetwarzania. Dla bitu $ADLAR=1$ zawiera 8 najstarszych bitów wyniku. Dla $ADLAR=0$ na pozycjach 1 i 0 są zapisywane dwa najstarsze bity wyniku, zaś pozostałe bity rejestru są niewykorzystywane.

ADCL \$04

B7-B0 Młodszy bajt wyniku przetwarzania. Dla bitu $ADLAR=0$ zawiera 8 najmłodszych bitów wyniku. Dla $ADLAR=1$ na pozycjach 7 i 6 są zapisywane dwa najmłodsze bity wyniku, zaś pozostałe bity rejestru są niewykorzystywane.

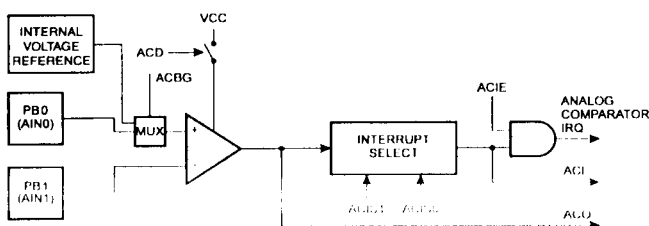
MCUCR \$35

B4, B3 SM1, SM0 – bity sterujące trybem obniżonego poboru mocy. Dla wartości tych bitów „01” kontroler po skonfigurowaniu przez twornika przystępuje do konwersji w momencie wykonania rozkazu SLEEP, przechodząc w tryb redukcji szumów, (zawieszając swoją działalność). Dzięki temu maleje tło szumów i rośnie dokładność przetwarzania. Tryb redukcji szumów jest dozwolony tylko dla pojedynczych pomiarów i nie jest uwzględniany dla cyklicznej konwersji.

Układ komparatora analogowego

Układ komparatora występuje tylko w kontrolerach ATtiny 10/11/12/15. Schemat 10 przedstawia jego budowę.

Schemat 10. Komparator analogowy



Wbudowany komparator analogowy pozwala na porównanie zewnętrznych napięć, doprowadzonych do wyprowadzeń PB0 (AIN0) i PB1 (AIN1). Opcjonalnie w układach ATtiny12 i ATtiny15 zamiast sygnału z wejścia AIN0, do porównania możemy użyć wewnętrznego źródła napięcia odniesienia $1,22\text{ V} \pm 0,05\text{ V}$. Dokonujemy tego przez ustawienie bitu ACBG w stan wysoki. Komparator pobiera stosunkowo duży prąd. Aby wyeliminować tę wadę zastosowano bit uaktywniający zasilanie komparatora. Jest on dostępny w rejestrze ACSR. Uaktywnienie uzyskujemy przez nadanie wartości 0 bitowi ACD. Odczytanie aktualnego wyniku porównania umożliwia bit ACO. Bardzo dobrze rozwiązano problem obsługi przerwań od komparatora. Za pomocą bitów ACIS1 i ACIS0 wybieramy warunek, którego spełnienie zapewni ustawienie znacznika zgłoszenia przerwania ACI. Do wyboru mamy trzy warunki. Te warunki to reakcja na zbocza opadające, narastające, lub na zmianę stanu na wejściu (reakcja na zbocza opadające i narastające). Przerwania od komparatora możemy zamaskować przez załadowanie 0 do bitu ACIE. Komparator wywołuje obsługę przerwania, wskazywaną przez wektor, znajdujący się pod adresem \$005 dla ATtiny10/11/12, lub \$007 dla ATtiny15. Poniżej przedstawiony jest opis rejestru ACSR.

ACSR	\$08
B7	ACD – bit sterujący załączeniem zasilania komparatora analogowego. Jeśli ACD=1, to komparator jest wyłączony. Przed wyłączeniem komparatora musimy zablokować przerwanie, poprzez wyzerowanie bitu ACIE.
B6	ACBG – dotyczy tylko ATtiny12/15. Bit równy 1 dołącza do porównania źródła napięcia odniesienia $1,22\text{ V} \pm 0,05\text{ V}$ zamiast napięcia z wejścia AIN0
B5	ACO – bit połączony bezpośrednio z wyjściem komparatora. Odzworowuje wynik porównania napięć wejściowych.
B4	ACI – flaga przerwania od komparatora analogowego. Ustawiana jest w momencie spełnienia warunku, wybranego za pomocą bitów ACIS1, ACIS0.
B3	ACIE – bit zezwolenia na przerwanie od komparatora. Ustawienia tego bitu w stan wysoki oznacza zezwolenie na przerwanie od komparatora.
B1, B0	ACIS1, ACIS0 – bity wybierające warunek przerwania od komparatora.

ACIS1, ACIS0

0	0	przerwanie wywoływane wystąpieniem zbocza na wyjściu komparatora
0	1	niewykorzystane
1	0	przerwanie wywoływane zboczem opadającym na wyjściu komparatora
1	1	przerwanie wywoływane zboczem narastającym na wyjściu komparatora

UWAGA: W chwili zmiany stanu bitów ACIS1, ACIS0 należy za-
blokować przerwanie od komparatora. W przeciwnym wypadku
może wystąpić niekontrolowane przerwanie od komparatora.

Porty wejścia/wyjścia

Porty wejścia/wyjścia są widziane przez programistę, jako rejestry w obszarze adresów wejścia/wyjścia. Do obsługi portu przypadają trzy rejestry o nazwach DDB, PORTB, PINB. Ustawienie odpowiedniego bitu rejestru DDB dla danej linii w stan wysoki powoduje, iż zostaje on przełączony w tryb pracy jako wyjście. Rejestr PORTB stanowi port wyjścia. Zapisane w nim dane w trybie wyjścia zostają wystawione na wyprowadzenia, a w trybie wejścia sterują podciąganiem wyprowadzeń portu (pull-up). Rejestr PINB umożliwia odczyt stanu portu bezpośrednio z jego wyprowadzeń. Należy pamiętać, iż rejestr PINB jest tylko do odczytu. Zapis do nich nie powoduje żadnej reakcji. Tabela 8 przedstawia możliwe kombinacje ustawień dla linii portu.

Tabela 8. Konfigurowanie portu

DDBn	PORTBn	Kierunek	Podciąganie (Pull-up)	Opis
0	0	wejście	nie	Wejście trójstanowe (wysoka-impedancja). Linia wejściowa będąca źródłem prądu IIL jeśli zewnętrznie zostanie podciągnięta do stanu niskiego
0	1	wejście	tak	Wyjście typu push-pull - stan niski
1	0	wyjście	nie	Wyjście typu push-pull - stan niski
1	1	wyjście	nie	Wyjście typu push-pull - stan wysoki

n – nr bitu w porcie

Cechą charakterystyczną wyprowadzeń portu jest wydajność prądowa wyjść w stanie niskim i wysokim, wynosząca 20 mA (max 40 mA). Umożliwia to bezpośrednieysterowanie wyświetlaczy LED, a nawet niektórych przekaźników. Zgodnie z opisem port obsługiwany jest przez trzy rejestry w przestrzeni adresowej danych. Rejestry te widzimy pod następującymi adresami: PINB-\$16, DDRB-\$17, PORTB-\$18. W przypadku kontrolera ATtiny15, występuje jednokierunkowa linia portu PB5, dla której nie występuje bit PORTB5. Możemy jej używać wyłącznie jako wejście. Dodatkowo w kontrolerze ATtiny15 w rejestrze MCUCR występuje bit PUD. Ustawienie tego bitu powoduje wyłączenie podciągania do stanu wysokiego wszystkich linii portu. Większość wyprowadzeń w tak małych kontrolerach spełnia dodatkowe funkcje specjalne. Tabela 9 przedstawia opis tych funkcji.

Tabela 9. Funkcje specjalne wyprowadzeń portu B

Wyprowadzenie	Kontroler/funkcja wyprowadzenia					
	AT90S2323	AT90S2343	ATtiny 10/11	ATtiny 12	ATtiny 15	ATtiny 22
PB0	MOSI	MOSI	AIN0	MOSI/AIN0	MOSI/AIN0/AREF	MOSI
PB1	MISO/INT0	MISO/INT0	INT0/AIN1	MISO/INT0/AIN1	MISO/AIN1/OCF	MISO/INT0
PB2	SCK/T0	SCK/T0	T0	SCK/T0	SCK/ADC1/TQ/INT0	SCK/T0
PB3		CLOCK	XTAL1	XTAL1	ADC2	CLOCK
PB4			XTAL2	XTAL2	ADC3	
PB5			RESET	RESET	RESET/ADC0	

- ADCx – wejście analogowe przetwornika A/C
 AIN0 – wejście nieodwracające komparatora
 AIN1 – wejście odwracające komparatora
 CLOCK – wejście zegara taktującego kontroler
 INT0 – wejście przerwania zewnętrznego
 MISO – wyjście danych układu SPI dla trybu programowania
 MOSI – wejście danych układu SPI dla trybu programowania
 OCF – wyjście porównania układu PWM
 REF – wejście napięcia odniesienia układu przetwornika A/C
 RESET – wejście zerowania kontrolera
 SCK – wejście taktujące układu SPI dla trybu programowania
 T0 – wejście zewnętrznego generatora dla układu zegara T0
 XTALx – wyprowadzenia dołączenia elementów oscylatora taktującego kontroler

MCUCR **\$035**

B6 **PUD** – dotyczy tylko ATtiny15. Bit równy 1 wyłącza podciąganie linii portu do stanu wysokiego

PORTB **\$018**

B0-B4 Rejestr wyjściowy portu B. Liczba linii zależna od typu kontrolera (patrz tab. 4).

DDB \$017

B0-B5 Rejestr kierunku linii portu B. Liczba linii zależna od typu kontrolera (patrz tab. 4).

PINB \$016

B0-B5 Rejestr wejściowy portu B. Liczba linii zależna od typu kontrolera (patrz tab. 4).

Rejestry robocze, obszar wejścia/wyjścia i pamięć danych

W przypadku mikrokontrolerów AVR mamy do dyspozycji 32 rejestry robocze adresowane od \$00 do \$1F (0-31d). Rejestry możemy podzielić na dwie grupy. Grupa o adresie \$00-\$0F bez możliwości wykonania rozkazów z wartością bezpośrednią (stałą zawartą w rozkazie), oraz grupa o adresie \$10-\$1f z możliwością wykonania rozkazów, zawierających wartość bezpośrednią. W zależności od listy rozkazów, mamy również 1 (ATtiny10/11/12/15), lub 3 (2323/2343/ATtiny22), szesnastobitowe rejestry indeksowe X(R26, R27), Y(R28, R29), Z(R30, R31). Możemy ich używać jako 16-bitowych wskaźników, pozwalających na adresowanie pośrednio rejestrów roboczych. Przy czym należy pamiętać, iż w trybie adresowania indeksowego możemy zaadresować zarówno rejestry robocze, jak i rejestry wejścia/wyjścia. Rejestry robocze umieszczone są od adresu \$000 do \$01F. Zaraz za nim począwszy od adresu \$020, umieszczony jest obszar rejestrów wejścia/wyjścia. Adres do tego obszaru, tworzymy poprzez dodanie do normalnego adresu rejestru, przesunięcie o wartości \$020. Za obszarem wejścia/wyjścia jest umieszczony obszar pamięci danych, zaczynający się od adresu \$60, którego adres pośredni, wyznaczamy poprzez dodanie do adresu bezpośredniego, wartości przesunięcia równej \$60. Pamięć danych w ilości 128 B zawierają kontrolery 2323/2343, ATtiny22.

Stos

W mikrokontrolerach w obudowach 8-wyprowadzeniowych, mamy do dyspozycji sprzętowy stos, umieszczony w pamięci danych, gdy ta kontroler posiada, lub stos odseparowany od pamięci danych dla kontrolerów, posiadających tylko rejestry danych (ATtiny10/11/12/15).

Stos odseparowany ma głębokość trzech poziomów. Aby określić jaki rodzaj stosu posiada dany kontroler, patrz do tabeli 4. Wskaźnik stosu występuje tylko w kontrolerach z pamięcią danych. Wierzchołek stosu jest wskazywany przez rejestr-wskaźnik stosu SPL (\$3D). Jako, iż pierwsze 96 bajtów pamięci wewnętrznej to rejestry robocze i obszar wejścia/wyjścia, stos możemy umieścić w obszarze od \$060 do \$0DF. Wierzchołek stosu trzeba na samym początku programu zainicjalizować. Stos ma tak zwaną organizację malejącą. Oznacza to, iż odłożenie danych lub adresu powoduje dekrementację wskaźnika wierzchołka, a zdejmowanie ze stosu powoduje inkrementację. Dla odłożenia na stos następuje załadowanie odkładanego bajtu pod adres, zawarty we wskaźniku stosu SPL, a następnie dekrementacja wskaźnika wierzchołka stosu. Dla zdjęcia ze stosu wykonywana jest inkrementacja wskaźnika wierzchołka stosu, i w następnej kolejności podniesienie bajtu. Dla stosu w obszarze danych istnieją dodatkowe dwa rozkazy jego obsługi – PUSH, POP.

Pamięć EEPROM

Ilość pamięci EEPROM zawarta w kontrolerach jest różna, w zależności od typu kontrolera (patrz tabela 4, wiersz 10). Stanowi ona oddzielny obszar danych, do którego dostęp realizowany jest z wykorzystaniem rejestrów: sterującego danych i rejestru adresowego, mieszczących się w przestrzeni adresowej wejścia/wyjścia. Pamięć EEPROM wytrzymuje 100 000 cykli zapisu/odczytu. Zapis do pamięci trwa od 2.5 do 4 ms i jest uzależniony od napięcia zasilającego VCC. Przy napięciu zasilającym równym 5 V, czas zapisu trwa około 2,5 ms i rośnie on do około 4 ms dla napięcia równego 2,7 V. Wykrywanie, kiedy może nastąpić kolejny zapis do pamięci, spoczywa na oprogramowaniu użytkowym. Wewnętrzne układy mikrokontrolera zabezpieczają przed próbą zapisu do pamięci, gdy napięcie VCC jest poniżej wartości, zapewniającej właściwy zapis. Jeżeli jest dokonywana operacja zapisu lub odczytu pamięci EEPROM, to praca mikrokontrolera zostaje wstrzymana na dwa cykle zegarowe, przed wykonaniem kolejnego rozkazu, następującego po tej operacji. Aby dokonać zapisu przy ustawionym adresie i zapisanej do rejestru danych bajtu informacji, musimy odczekać do momentu, aż bit EEMWE zostanie wyzerowany przez mikrokontroler. Zapisać jedynkę do EEMWE, a następnie w ciągu czterech cykli rozkazowych ustawić w stan wysoki bit EEMWE. Inno-

wacją jest wprowadzenie dodatkowego przerwania, wywoływanego w momencie, gdy pamięć jest gotowa do następnej operacji dostępu. Pozwala to na efektywne wykorzystanie czasu kontrolera. Ustawiony bit EERIE zezwala na przerwanie w momencie, gdy pamięć EEPROM jest gotowa do kolejnego zapisu (bit EERWE=0), w efekcie sprzętowego wyzerowania bitu EERWE, zostaje wywoływany wektor przerwania znajdujący się pod adresem \$004 dla ATtiny12, lub \$006 dla ATtiny15. Bajt sterujący dostępem do pamięci EEPROM o nazwie EECR oraz pozostałe rejestry, sterujące dostępem do pamięci EEPROM wyglądają następująco:

EECR		\$01C
B3	EERIE	– dotyczy tylko ATtiny12/15. Jeśli bit jest równy 1, to jest zezwolenie na przerwanie spowodowane gotowością pamięci EEPROM. Przerwanie jest wywoływane przez sprzętowe wyzerowanie bitu EERWE. Wywoływany wektor przerwania znajduje się pod adresem \$004 dla ATtiny12 i \$006 dla ATtiny15
B2	EEMWE	– bit głównego zabezpieczenia przed zapisem do pamięci EEPROM. Zerowany automatycznie po czterech cyklach rozkazowych
	0	zmiana bitu EERWE nie powoduje żadnej reakcji
	1	zezwolenie na zapis do pamięci EEPROM
		Zapis do pamięci musi nastąpić w ciągu 4 cykli, od momentu ustawienia bitu zezwolenia. Po tym czasie mikrokontroler automatycznie blokuje zezwolenie na zapis do pamięci, przez sprzętowe wyzerowanie bitu EEMWE
B1	EERWE	– bit zezwolenia na zapis. Po załadowaniu właściwego adresu i danych, aby został wykonany zapis, należy nadać temu bitowi wartość „1”. Po dokonaniu zapisu bit zostaje wyzerowany sprzętowo. Program użytkownika przed kolejnym zapisem musi go testować i oczekiwać na jego wyzerowanie
B0	EERE	– bit zezwolenia na odczyt. Po zapisaniu do rejestru adresowego adresu komórki, którą chcemy odczytać, musimy nadać temu bitowi wartość jeden. Bit ten jest zerowany sprzętowo po dokonaniu odczytu z pamięci EEPROM, a odczytana dana znajduje się w rejestrze danych EEDR

FEEDR \$01D

37-B0 bajt danych do zapisania, lub odczytane z pamięci EEPROM.

EEAR \$01E

36-B0 bity adresu, wskazujące komórkę pamięci EEPROM, której ma dotyczyć operacja zapisu lub odczytu danych.

128B – 2323/2343/ATTiny22, 64B – ATTiny12/15

UWAGA!

Jeżeli w trakcie zapisu do pamięci EEPROM wystąpi RESET mikrokontrolera, to zapis do pamięci nie zostanie przerwany, jednak zostanie wyzerowany rejestr adresu EEAR. W efekcie zapis nastąpi pod adres \$00, zamiast pod wskazany adres. Gdy może wystąpić sygnał RESET w budowanym przez Ciebie układzie, to jeśli jest możliwe, używaj pamięci EEPROM od adresu \$01, pozostawiając niewykorzystaną komórkę o adresie \$00. Warunkiem bezpiecznego korzystania z adresu \$00 jest niemożność wywołania zewnętrznego RESET-u, oraz zablokowanie na czas zapisu pamięci układu Watchdog.

Pamięć programu

Zorganizowana jest w słowa o rozmiarze 16 bitów i tak jest traktowana, podczas pobierania rozkazów do wykonania. Jednakże podczas programowania dane są przesyłane, jako pojedyncze bajty, ze wskazaniem adresu słowa i połówki, której dotyczy zapis lub odczyt przy weryfikacji. Początkowy obszar pamięci zajmują komórki pamięci, w których umieszcza się wektory (rozказы) skoków do podprogramów obsługi zdarzeń (przerwań), w tym również obsługi RESET. Typowo umieszcza się w tym obszarze rozказы skoków do procedur, obsługujących dane zdarzenia. Dla programu użytkownika pozostaje obszar pamięci, począwszy od pierwszego adresu powyżej najwyższego wektora. Należy pamiętać, iż rozkaz LPM adresuje kolejne bajty pamięci programu, nie słowa.

Układ przerwań

Przerwania są zgłaszane przez urządzenia zewnętrzne do jednostki centralnej. Musimy pamiętać, aby każdy blok kontrolera, którego rejestry

leżą w przestrzeni adresowej wejścia/wyjścia (I/O) utożsamiać z urządzeniem zewnętrznym, mimo iż znajduje się w obudowie kontrolera. Mikrokontrolery AVR posiadają jeden bit globalnego sterowania przerwaniami „I”. Oprócz niego za sterowanie przerwaniami odpowiadają bity, znajdujące się w rejestrach urządzeń, których pracą one sterują. Bit sterujący przerwaniami globalnymi o nazwie I, znajduje się w rejestrze SREG(\$3F). Zezwolenia na wybrane przerwanie, jak i na przerwania globalne, dokonujemy poprzez ustawienie bitu zezwolenia w stan wysoki. Ogólną zasadą jest, iż zmiana trybu pracy urządzenia zewnętrznego powinna odbywać się przy zablokowanych przerwaniach. Niespełnienie tego warunku może być przyczyną wywoływania nieprzewidzianych przerwań, w trakcie ustawiania tych zmian. Dodatkowo kontrolery ATtiny10/11/12/15 został wyposażony w mechanizm przerwań od dowolnego wyprowadzenia, skonfigurowanego jako wejście. Przerwaniami tym sterują bity PCIE – zezwolenie i PCIF – znacznik. Jeśli zezwolimy na przerwanie, to w przypadku wystąpienia na dowolnym z aktualnych wejść, (zdefiniowanych w rejestrze DDRB), zmiany stanu na czas co najmniej jednego cyklu, spowoduje zgłoszenie przerwania nr 3 o wektorze pod adresem \$002.

Z przerwaniami zewnętrznymi jest ściśle powiązany rejestr MCUCR, znajdujący się pod adresem \$035. Zawiera on bity, sterujące sposobem wyzwalania przerwań od urządzeń zewnętrznych, poprzez wyprowadzenie INTO. Jest to bardzo praktyczne i nowatorskie rozwiązanie w stosunku do innych mikrokontrolerów. Dzięki temu wyzwalanie może się odbywać zarówno z boczem opadającym, poziomem niskim, jak i z boczem narastającym, zaś w przypadku kontrolerów ATtiny10/11/12/15, również dowolnym z boczem. Należy pamiętać, iż w przypadku przerwania zgłaszanego poziomem niskim, przerwanie jest zgłaszane tak długo, jak długo występuje ten poziom. Opis bitów rejestru MCUCR, powiązanych z przerwaniami, przedstawiony został poniżej.

GIMSK \$03B

- B6 INTO** – bit zezwolenia na przerwania od układów zewnętrznych na wejściu INTO.
- B5 PCIE** – dotyczy tylko ATtiny10/11/12/15. Bit zezwolenia na przerwanie od dowolnych wyprowadzeń portu, skonfigurowanych jako wejścia.

GIFR \$03B

- B6 INTF0** – bit zgłoszenia przerwania od układów zewnętrznych na wejściu INTO.

B5 **PCIF** – dotyczy tylko ATtiny10/11/12/15. Bit zgłoszenia przerwania od układów zewnętrznych, dołączonych do dowolnego wyprowadzenia skonfigurowanego jako wejście.

MCUCR **\$035**

B1,B0 **ISC01, ISC00**

0	0	Wyzwalanie przerwania poziomem niskim na wejściu INT0
0	1	Dotyczy tylko ATtiny10/11/12/15. Wyzwalanie przerwania dowolnym zboczem
1	0	Wyzwalanie przerwania zboczem opadającym
1	1	Wyzwalanie przerwania zboczem narastającym

Flaga zgłoszenia przerwania od układów zewnętrznych jest niedostępna programowo. Jest ona ustawiana w momencie wykrycia warunku przerwania. Zerowanie tej flagi odbywa się automatycznie, w momencie rozpoczęcia obsługi wektora przerwania. W wyniku przerwania od INT0 jest wywoływana procedura, wskazywana przez wektor nr 2, znajdujący się pod adresem \$001.

Zezwoleniem na przerwania od zegara/licznika TC0 i TC1 steruje rejestr TIMSK, znajdujący się pod adresem \$039.

TIMSK **\$039**

B6 **OCIE1A** – tylko ATtiny15. Bit zezwolenia na przerwanie w wyniku wystąpienia równości podczas porównania licznika TC1

0 przerwania w wyniku osiągnięcia wartości zadanej zabronione

1 przerwania dozwolone

B2 **TOIE1** – tylko ATtiny15. Zezwolenie na przerwania od układu zegara/licznika TC1 wywołanego przepełnieniem, lub osiągnięciem wartości zadanej w rejestrze OCR1B

B1 **TOIE0** – zezwolenie na przerwania od układu zegara/licznika TC0, spowodowane przepełnieniem

TIFR **\$38**

B6 **OCF1A** – tylko ATtiny15. Flaga zgłoszenia przerwania w wyniku wystąpienia przechwycenia wartości licznika

B2 **TOV1** – flaga zgłoszenia przerwania w wyniku przepełnienia zegara/licznika TC1

- B1 TOV0** – flaga przepelnienia licznika ustawiana sprzętowo. Należy pamiętać, iż bit ten jest również zerowany sprzętowo po wywołaniu wektora przerwania dla zegara/licznika (nie wykonania rozkazu RETI). Możemy również sterować wartością tego bitu programowo.

Obsługa przerwań nd przetwornika A/C jest realizowana z użyciem rejestru statusu przetwornika ADCSR i dotyczy tylko kontrolera ATtiny15.

ADCSR \$06

- B4 ADIF** – znacznik (flaga) zgłoszenia przerwania. Ustawiany jest po każdym zakończeniu przetwarzania.
- B3 ADIE** – bit zezwolenia na przerwania od przetwornika A/C. Jeśli bit jest równy 1 to w wyniku ustawienia flagi ADIF zostaje wywołane wektora przerwania nr 9 pod adresem \$008.

Za zezwolenie na przerwania od komparatora analogowego odpowiada rejestr ACSR, znajdujący się pod adresem 08H.

ACSR \$08

- B4 ACI** – bit zgłoszenia przerwania od komparatora. Ustawiany sprzętowo po spełnieniu zadanego warunku, ustawianego za pomocą bitów ACIS1, ACIS0. Zerowanie tego bitu odbywa się sprzętowo, w momencie wywołania wektora przerwania dla zegara/licznika.
- B3 ACIE** – bit zezwolenia na przerwania od komparatora analogowego. Ustawiany dla zezwolenia na przerwania

B1, B0 ACIS1, ACIS0

0	0	przerwanie następuje podczas zmian stanu na wyjściu komparatora.
0	1	zarezerwowane
1	0	przerwanie następuje podczas zbocza, narastającego na wyjściu komparatora.
1	1	przerwanie następuje podczas zbocza, opadającego na wyjściu komparatora

Dodatkowe rozszerzenie obsługi pamięci EEPROM powoduje wywołanie przerwania, gdy pamięć jest gotowa do następnej operacji zapisu. Jest to bardzo wygodne przerwanie, jeśli uwzględnimy czas zapisu do pamięci na 2,5-4 ms.

EECR**B3****\$01C**

EERIE – dotyczy tylko ATtiny12/15. Jeśli bit jest równy 1, to jest zezwolenie na przerwanie spowodowane gotowością pamięci EEPROM. Przerwanie jest wywoływane przez sprzętowe wyzerowanie bitu EEW. Wywołany wektor przerwania znajduje się pod adresem \$004 dla ATtiny12 i \$006 dla ATtiny15.

Przyjęcie i zakończenie przerwania

Przyjęcia przerwania przebiega następująco:

- zostaje ustawiona flaga odpowiednia dla danego źródła przerwań
- następuje kasowanie znacznika zezwolenia na przerwanie globalne I (SREG)
- następuje skok pod adres wektora przerwania dla danego urządzenia
- zostaje wyzerowana flaga przerwania dla danego urządzenia.

Taki sposób obsługi przerwań gwarantuje obsługę tylko jednego przerwania w danym momencie. Jednak przy stosie o tak dużej głębokości jaką możemy uzyskać, nic nie stoi na przeszkodzie, aby w obsłudze przerwania ustawić znacznik I. W ten sposób zezwalamy na inne przerwania w trakcie obsługi przerwania. Możemy wykorzystać nawet całą pamięć danych na stos. Przerwanie kończymy poprzez wykonanie instrukcji RETI. Wykonanie tej instrukcji powoduje:

- podniesienie adresu powrotu ze stosu.
- ustawienie znacznika I(SREG).

Czas obsługi przerwania

Bardzo ważny w przypadku krytycznych czasowo procedur obsługi zdarzeń jest czas przyjęcia i zakończenia przerwania. Od momentu pojawienia się zdarzenia wywołującego przerwanie, muszą upłynąć 4 cykle zegarowe. Jeśli po upływie czterech cykli zegarowych jednostka centralna jest w trakcie wykonywania rozkazu o czasie trwania większym niż 1 cykl zegarowy, obsługa przerwania zostaje wstrzymana do zakończenia tego rozkazu. Po tym czasie jednostka centralna przystępuje do wykonania obsługi przerwania. W czasie 2 cykli rozkazowych na stos zostaje odłożony adres powrotu. Następnie zostaje wykonana

instrukcja, znajdująca się pod adresem wektora przerwania odpowiedniego dla zdarzenia, powodującego przerwanie. Łączny czas przyjęcia przerwania (razem z wykryciem go), trwa minimum 8 cykli rozkazowych. Powrót z obsługi przerwania trwa 4 cykle zegarowe, podczas których zostaje podniesiony ze stosu adres powrotu oraz ustawiany znacznik zezwolenia na przerwania globalne I(SREG). Po wyjściu z przerwania zostaje wykonana przynajmniej jedna instrukcja z programu głównego, zanim zostanie przyjęte kolejne przerwanie, pod warunkiem wykonania 4 cykli zegarowych, podczas których jednostka centralna wykrywa zaistnienie zdarzenia zgłoszenia przerwania. Jeśli występuje więcej niż jedno przerwanie równocześnie, w pierwszej kolejności zostaje przyjęte zgłoszenie o najwyższym priorytecie (przerwanie o najniższym numerze). Priorytety obsługi zdarzeń wraz z adresami wektorów przerw, przedstawione zostały dla wszystkich kontrolerów w tabeli 10.

Tabela 10. Wektory obsługi zdarzeń

Rodzaj przerwania	Kontroler				
	AT90S2323/2343	ATtiny10/11	ATtiny12	ATtiny15	ATtiny22
RESET	1/\$000	1/\$000	1/\$000	1/\$000	1/\$000
INT0	2/\$001	2/\$001	2/\$001	2/\$001	2/\$001
I/O Pins		3/\$002	3/\$002	3/\$002	
TC1 COMPA				4/\$003	
TC1 OVF				5/\$004	
TC0 OVF	3/\$002	4/\$003	4/\$003	6/\$005	3/\$002
EE READY			5/\$004	7/\$006	
COMPARATOR		5/\$004	6/\$005	8/\$007	
ADC				9/\$008	

Tryby pracy mikrokontrolera z obniżonym poborem mocy

Układy mikrokontrolerów z serii AVR wyposażono w instrukcję przejścia w tryb obniżonego poboru mocy SLEEP. Mimo, iż istnieje jedna instrukcja przełączająca mikrokontroler w tryb obniżonego poboru mocy, mamy do dyspozycji dwa tryby uśpienia, a w przypadku ATtiny15 trzy. Wybieramy je poprzez ustawienie odpowiednich bitów w rejestrze MCUCR, umieszczonym pod adresem 035 h.

Wprowadzenie w tryb obniżonego poboru mocy wykonujemy ustawiając bit SE, a następnie wykonując rozkaz SLEEP. Jeżeli w trybie SLEEP wystąpi przerwanie, mikrokontroler budzi się z trybu uśpienia. Przystępuje do wykonania procedury obsługi przerwania, a następnie po jej wykonaniu podejmuje wykonywanie programu od rozkazu, następującego po poleceniu SLEEP. Jeśli podczas trybu SLEEP wystąpi RESET mikrokontrolera, przystępuje on do wykonania odpowiadającego mu wektora obsługi zdarzenia. Poniżej został przedstawiony opis bitów, sterujących wyborem trybu pracy z obniżonym poborem mocy.

MCUCR	\$35		
B5	SE	– ustawienie tego bitu zezwala na wejście w tryb obniżonego poboru mocy (SLEEP)	
B4	SM	– bit nie dotyczy Attiny15. Wybór trybu obniżonego poboru mocy	
	0	wybrany tryb Idle	
	1	wybrany tryb Power Down	
B4,B3	SM1, SM0	– bity dotyczą tylko Attiny15. Wybór trybu obniżonego poboru mocy	
	0	0	wybrany tryb Idle
	0	1	wybrany tryb redukcji szumów przetwor- nika AC
	1	0	wybrany tryb Power Down
	1	1	zarezerwowane

Idle

Wprowadzenie w tryb Idle wymaga wyzerowania bitu SM (ATtiny15 SM1, SM0=01), a następnie wykonania rozkazu SLEEP. Tryb Idle cha-

rakteryzuje się tym, iż jednostka centralna zostaje wprowadzona w tryb bezczynności, a układy peryferyjne pracują dalej. Jeśli nie jest wymagane zgłoszenie przerwania od komparatora analogowego, wskazane jest jego wyłączenie programowe, celem zminimalizowania poboru mocy. Z tego trybu mikrokontroler może zostać wyprowadzony poprzez dowolne urządzenie wejścia/wyjścia, lub poprzez RESET.

Power Down

Ten tryb pracy uzyskujemy po ustawieniu bitu SM (ATtiny15 SM1, SM0=10) i wykonaniu rozkazu SLEEP. Wprowadzenie mikrokontrolera w ten tryb powoduje zablokowanie zewnętrznego oscylatora. Wyprowadzić z tego trybu uśpienia mikrokontroler może jedynie timer watchdog, przerwania zewnętrzne INT0, INT1 wyzwalane poziomem oraz RESET. Jeśli wykorzystujemy zewnętrzne przerwanie wyzwalane poziomem niskim do rozbudzenia mikrokontrolera, poziom niski musimy utrzymywać na wejściu INTx przez czas tTOUT (patrz układ RESET). Jeśli podany czas nie będzie występował odpowiednio długo, mikrokontroler pozostanie w stanie uśpienia.

ADC Tryb redukcji szumów

Ten tryb dotyczy tylko kontrolera ATtiny15. Jest on ściśle powiązany z przetwornikiem A/C. Aby kontroler wprowadzić w ten tryb, należy nadać bitom SM1, SM0 wartość 01. Wykonanie rozkazu SLEEP powoduje wejście jednostki centralnej w tryb redukcji zakłóceń przetwornika i start przetwarzania. Po zakończeniu przetwarzania wyjście z tego trybu następuje poprzez zgłoszenie przerwania od przetwornika A/C, przerwanie zewnętrzne wyzwalane poziomem, przerwanie zgłaszane zmianą na dowolnym wejściu kontrolera lub poprzez reset.

Tryby adresowania pamięci programu, danych i obszaru wejścia/wyjścia

Od dostępnych trybów adresowania, zależy szybkość wykonywania wielu procedur. Jednakże mnogość trybów adresowania wcale nie musi powodować znacznego przyspieszenia wykonywanych działań. Wręcz odwrotnie, powoduje zwiększenie komplikacji budowy mikrokontrolera i niejednokrotnie zmniejszenie szybkości wykonywania samych rozkazów mikrokontrolera. W AT90S2313 możemy wyróżnić:

Tryb bezpośredniego adresowania rejestrów

Charakteryzuje się on bezpośrednim wskazaniem rejestru, na którym ma zostać wykonana operacja. Wskazanie rejestru umieszczone jest w samym kodzie rozkazu.

Tryb bezpośredniego adresowania dwóch rejestrów

Charakteryzuje się on bezpośrednim wskazaniem rejestrów, na których ma zostać wykonana operacja. Wskazanie rejestrów umieszczone jest w samym kodzie rozkazu. Wynik operacji zostaje umieszczony w rejestrze Rd.

Tryb bezpośredniego adresowania obszaru wejścia/wyjścia

Charakteryzuje się on bezpośrednim wskazaniem rejestru wejścia/wyjścia, na którym ma zostać wykonana operacja. Wskazanie rejestru umieszczone jest w samym kodzie rozkazu.

Tryb bezpośredniego adresowania pamięci danych

W tym trybie szesnastobitowy adres jest zawarty w rozkazie. Rd/Rr wskazuje na rejestr źródłowy lub docelowy operacji.

Tryb pośredniego adresowania danych z przemieszczeniem

Bardzo wygodny tryb dla przetwarzania rekordów, itp. Pozwala na zaadresowanie struktury danych za pomocą rejestru Y lub Z i odczytanie konkretnej danej ze struktury, za pomocą przesunięcia zapisanego na sześciu bitach w słowie rozkazu.

Tryb pośredniego adresowania rejestrów

Charakteryzuje się wskazaniem rejestru, na którym ma zostać wykonana operacja za pomocą adresu, zawartego w tzw. rejestrze indeksowym. Wskazanie rejestru polega na załadowaniu adresu rejestru (podmiotu) do rejestru wskaźnikowego X, Y, Z (wskazującego podmiot działania). Często ten adres nazywany jest wskaźnikiem, gdyż wskazuje rejestr, na którym wykonywana jest operacja. Wskaźnik do danych jest bardzo wygodnym sposobem przekazywania parametrów i danych do wywoływanych procedur.

Tryb adresowania pośredniego danych z pre-dekrementacją

Ten tryb adresowania jest bardzo wygodny dla przetwarzania długich struktur danych. Charakteryzuje się automatyczną dekrementacją adresu wskaźnikowego przed wykonaniem operacji, na wskazywanej komórce pamięci.

Tryb adresowania pośredniego danych z post-inkrementacją

Ten tryb adresowania jest podobny do przedstawionego powyżej. Jego zastosowanie jest podobne. Umożliwia on wykonanie operacji na wskazanej przez wskaźnik komórce pamięci, a następnie zostaje automatycznie zwiększany adres o 1.

Tryb adresowania stałych z użyciem rozkazu LPM

Bardzo często zachodzi konieczność tworzenia tablic stałych, potrzebnych w programie. Adresowanie odbywa się z wykorzystaniem rejestru Z. Najmłodszy bit w słowie rozkazu decyduje, czy starszy (1), czy młodszy (0) bajt słowa programu jest odczytywany. Odczytany bajt danej jest ładowany do rejestru R0.

Tryb adresowania pośredniego pamięci programu. (IJMP, ICALL)

Tryb ten pozwala na skok bezwzględny ze śladem (ICALL), lub bez śladu (IJMP) do dowolnego adresu pamięci programu. Adres celu znajduje się w rejestrze Z.

Tryb adresowania względnego pamięci programu. (RJMP, RCALL)

W przypadku skoków i wywołań procedur w pamięci programu, możemy wykorzystać tryb adresowania względnego. Tryb ten charakteryzuje się tym, iż skok jest wykonywany względem aktualnej zawartości licznika programu PC. Skok względny może być wykonany w przód lub wstecz. Oznacza to, iż przesunięcie może mieć wartość dodatnią lub ujemną. Przemieszczenie (skok) jest zapisywany w kodzie U2. Ze względu na oszczędność pamięci jest wskazane, aby głównie na tym trybie oprzeć wywołania procedur i skoki.

Rejestry kontrolne i sterujące mikrokontrolera

Mikrokontrolery AVR posiadają 32 rejestry funkcji specjalnych, umieszczonych w przestrzeni adresowej wejścia/wyjścia. W tabeli 11 przedstawiono sumarycznie dostępne rejestry wraz z oznaczeniami poszczególnych bitów w tych rejestrach. Tabela przedstawia sumarycznie rejestry za wszystkie wersje kontrolerów. Czy dany bit lub rejestr występuje w konkretnym kontrolerze, należy sprawdzić w opisie kontrolera.

Tabela 11. Sumaryczne przedstawienie rejestrów funkcji specjalnych

Adres	Nazwa	B7	B6	B5	B4	B3	B2	B1	B0
\$3F (\$5F)	SREG	I	T	H	S	V	N	Z	C
\$3D (\$5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
\$3B (\$5B)	GIMSK		INT0	PCIE					
\$3A (\$5A)	GIFR		INTF0	PCIF					
\$39 (\$59)	TIMSK		OCIE1A				TOIE1	TOIE0	
\$38 (\$58)	TIFR		OCF1A				TOV1	TOV0	
\$35 (\$55)	MCUCR		PUD	SE	SM/SM1	SM0		ISC01	ISC00
\$34 (\$54)	MCUSR					WDRF	BORF	EXTRF	PORF
\$33 (\$53)	TCCR0						CS02	CS01	CS00
\$32 (\$52)	TCNT0	Timer/Counter0 (8 bit)							
\$31 (\$51)	OSCCAL	Rejestr Kalibracji Oscylatora							
\$30 (\$50)	TCCR1	CTC1	PWM1	COM1A1	COM1A0	CS13	CS12	CS11	CS10
\$21 (\$41)	WDTCR				WDTOE	WDE	WDP2	WDP1	WDP0
\$1E (\$3E)	EEAR	EEPROM Adres							
\$1D (\$3D)	EEDR	EEPROM Dane							
\$1C (\$3C)	EEDR					EERIE	EEMWE	EEWE	EERE
\$18 (\$38)	PORTB				PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
\$17 (\$37)	DDRB			DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
\$16 (\$36)	PINB			PINB5	PINB4	PINB3	PINB2	PINB1	PINB0
\$08 (\$28)	ACSR	ACD	ACBG	ACO	ACI	ACIE		ACIS1	ACIS0
\$07 (\$27)	ADMUX	REFS1	REFS0	ADLAR			MUX2	MUX1	MUX0
\$06 (\$26)	ADCSR	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0
\$05 (\$25)	ADCH	Rejestr danych przetwornika H							
\$04 (\$24)	ADCL	Rejestr danych przetwornika L							

SREG \$3F rejestr statusu

- B7 I – bit zezwolenia globalnego na przerwania. Kasowany sprzętowo w momencie przyjęcia obsługi przerwania. Ustawiany w chwili jej zakończenia, podczas wykonania rozkazu RETI. Bit dostępny programowo do odczytu i zapisu
- 0 przerwania zablokowane
- 1 przerwania odblokowane

- B6** **T** – bit kopii bitu dla instrukcji BST i BLD. Bit ten pośredniczy przy przesłaniach bit-bit. Ominięto w ten sposób nadmierne wykorzystanie znacznika CARRY, jak w przypadku innych mikrokontrolerów. Po wykonaniu rozkazu BST bit przyjmuje taką wartość, jak wskazany bit
- B5** **H** – wskaźnik przeniesienia pomocniczego. Jest to bit tzw. przeniesienia połówkowego. Jego ustawienie następuje w momencie przeniesienia z jednej połówki bajtu do drugiej, w trakcie wykonania operacji arytmetycznej lub logicznej. Pozwala on na prostą realizację arytmetyki BCD
- B4** **S** – bit znaku wyliczany poprzez funkcję EX-OR z bitów N i V
- B3** **V** – przepełnienie uzupełnienia do dwóch. Ustawiany w momencie przekroczenia wartości (pojemności) rejestru, na którym wykonano operację arytmetyczną
- B2** **N** – wskaźnik wartości ujemnej ustawiany, gdy wynik operacji arytmetycznej lub logicznej przyjmuje wartość ujemną. Bit stanowi zawsze kopię najstarszego bitu liczby (w kodzie U2 najstarszy bit stanowi tzw. bit znaku). Wyjątkiem są rozkazy zmieniające bezpośrednio wartość tego bitu
- B1** **Z** – wskaźnik zera. Ustawiany, gdy w wyniku wykonania operacji arytmetycznej lub logicznej wynik jest równy 0
- B0** **C** – wskaźnik przeniesienia. Ustawiany w wyniku operacji arytmetycznych lub logicznych

SPL \$3D rejestr wskaźnika wierzchołka stosu

- B7-B0** dotyczy AT90S2323/2343, ATtiny22. Wskaźnik wierzchołka stosu. Wskazuje na aktualnie wolne miejsce, na które może nastąpić odłożenie adresu lub danych. Stos ma tzw. odwrotny kierunek. Oznacza to, iż rośnie w kierunku malejących adresów

GIMSK \$3B rejestr ogólnego maskowania przerwań

- B6** **INT0** – bit zezwolenia na przerwanie od układów zewnętrznych na wejściu INT0
 0 przerwanie zabronione
 1 przerwanie dozwolone
- B5** **PCIE** – dotyczy ATtiny10/11/12/15. Zezwolenie na przerwanie od dowolnej linii portu, skonfigurowanej jako wejście

GIFR		\$3A rejestr flag przerwań zewnętrznych
B6	INTF0	– flaga zgłoszenia przerwania od układów zewnętrznych na wejściu INT0 INTF0=1 wystąpił warunek przerwania
B5	PCIF	– dotyczy ATtiny10/11/12/15. Flaga zgłoszenia przerwania, wywołanego zmianą na dowolnej linii portu, skonfigurowanej jako wejście
TIMSK		\$39 rejestr maskowania przerwań od TCx
B6	OCIE1A	– bit zezwolenia na przerwanie w wyniku porównania zawartości licznika z rejestrem OCR1A 0 przerwanie zabronione 1 przerwania dozwolone
B2	TOIE1	– bit zezwolenia na przerwanie spowodowane przepelnieniem TC1, lub osiągnięcia maksymalnej wartości, zadanej w rejestrze OCR1B. 0 przerwanie zabronione 1 przerwania dozwolone
B1	TOIE0	– zezwolenie na przerwania od układu zegara/licznika TC0 spowodowane przepelnieniem. Należy pamiętać, iż bit ten jest również zerowany sprzętowo po wywołaniu wektora przerwania TC0 OVF, (nie wykonania rozkazu RETI). Możemy również sterować wartością tego bitu programowo przez zapis logicznej jedynki 0 przerwania zabronione 1 przerwania dozwolone
TIFR		\$38 rejestr flagi zgłoszenia przerwania od zegarów/liczników
B6	OCF1A	– flaga zgłoszenia przerwania w wyniku porównania rejestru OCR1A z licznikiem 0 wartości różne 1 licznik osiągnął zadaną wartość
B2	TOV1	– flaga zgłoszenia przerwania w wyniku wystąpienia przepelnienia licznika 0 przepelnienia licznika TC1 nie nastąpiło 1 nastąpiło przepelnienia licznika TC1
B1	TOV0	– flaga zgłoszenia przerwania w wyniku przepelnienia licznika TC0

- 0 przerwanie niezgłoszone
 1 nastąpił warunek przepełnienia licznika T0

MCUCR \$35 rejestr kontrolny mikrokontrolera

B6 PUD – bit dotyczy tylko ATtiny15. Ustawienie tego bitu wyłącza podciąganie do stanu wysokiego wszystkich linii portu

B5 SE – ustawienie tego bitu zezwala na wejście w tryb obniżonego poboru mocy (SLEEP)

B4 SM – bit nie dotyczy ATtiny15. Wybór trybu obniżonego poboru mocy

0 wybrany tryb Idle

1 wybrany tryb Power Down

B4, B3 SM1, SM0 – bity dotyczą tylko ATtiny15. Wybór trybu obniżonego poboru mocy.

0 0 wybrany tryb Idle

0 1 wybrany tryb redukcji szumów przetwornika AC

1 0 wybrany tryb Power Down

1 1 zarezerwowane

B1- B0 ISC11, ISC10

0 0 wyzwalanie przerwania poziomem niskim na wejściu INT1

0 1 -

1 0 wyzwalanie przerwania zboczem opadającym

1 1 wyzwalanie przerwania zboczem wznoszącym

MCUSR \$34

B3 WDRF – dotyczy tylko ATtiny10/11/12/15. Bit przyjmuje wartość 1 dla resetu spowodowanego przez watchdog

B2 BORF – dotyczy tylko ATtiny10/11/12/15. Bit przyjmuje wartość 1 dla reset-u spowodowanego przez układ Brown Out Detection

B1 EXTRF – bit przyjmuje wartość 1 dla reset-u, spowodowanego zewnętrznym sygnałem

B0 PORF – bit przyjmuje wartość 1 dla reset-u, spowodowanego włączeniem zasilania

TCCR0 \$33 rejestr kontrolny zegara/licznika TC0

B2-B0 CS02, CS01, CS00 – bity wybierają stopień podziału preskalera dla TC0 oraz źródło sygnału, taktującego ten licznik

0	0	0	stop, zegar/licznik wyłączony
0	0	1	F licznika=CK
0	1	0	F licznika=CK/8
0	1	1	F licznika=CK/64
1	0	0	F licznika=CK/256
1	0	1	F licznika=CK/1024
1	1	0	F licznika=T0, reakcja na zbocze opadające
1	1	1	F licznika=T0, reakcja na zbocze narastające

TCNT0 \$32 rejestr zegara/licznika TC0

B7-B0 ośmiobitowy rejestr zegara licznika inkrementowany sprzętowo

OSCCAL \$31 rejestr kalibracji generatora taktującego kontroler

B7-B0 rejestr dotyczy tylko ATtiny15. Ośmiobitowy rejestr kalibracji wewnętrznego oscylatora (1,6 MHz), taktującego kontroler. Zapisanie \$FF powoduje ustawienie maksymalnej częstotliwości oscylatora

TCCR1 \$30 rejestr kontrolny zegara TC1

B7 **CTC1** – bit załączenia rejestru porównania OCR1B
 0 rejestr OCR1B nie ma wpływu na licznik TC1 rejestr TCNT1 odlicza do wartości \$FF
 1 rejestr OCR1B określa maksymalną wartość, jaką może przyjąć rejestr TCNT1

B6 **PWM1** – bit załączenia trybu pracy licznika
 0 tryb licznika
 1 tryb PWM

B5, B4 **COM1A1, COM1A0** – bity sterujące trybem pracy wyprowadzenia PB1 dla trybu porównania rejestru OCR1A z licznikiem, lub sterowania bitem FOC1A
 0 0 TC1 nie steruje wyprowadzeniem PB1
 0 1 przełączanie wyjścia na stan przeciwny
 1 0 zerowanie linii PB1 w wyniku porównania

1 1 ustawienie linii PB1 w wyniku porównania

B3-B0 CS13, CS12, CS11, CS10 – dotyczy tylko ATtiny15. Bity wyboru częstotliwości taktującej licznik TC1

0	0	0	0	stop, zegar/licznik1 wyłączony
0	0	0	1	F TC1=PCK
0	0	1	0	F TC1=PCK/2
0	0	1	1	F TC1=PCK/4
0	1	0	0	F TC1=PCK/8
0	1	0	1	F TC1=CK=PCK/16
0	1	1	0	F TC1=CK/2
0	1	1	1	F TC1=CK/4
1	0	0	0	F TC1=CK/8
1	0	0	1	F TC1=CK/16
1	0	1	0	F TC1=CK/32
1	0	1	1	F TC1=CK/64
1	1	0	0	F TC1=CK/128
1	1	0	1	F TC1=CK/256
1	1	1	0	F TC1=CK/512
1	1	1	1	F TC1=CK/1024

WDTCR \$21 rejestr kontrolny zegara watchdog

B4 WDTOE – bit zezwolenia na wyłączenie układu watchdog. Przed wyzerowaniem WDE, należy ustawić w stan wysoki WDTOE. Jeśli tego nie zrobimy, zerowanie bitu WDE zostanie zignorowane. Bit WDTOE po 4 cyklach zegarowych automatycznie zostaje wyzerowany. Operacja na WDE musi zostać wykonana w ciągu tych 4 cykli

B3 WDE – bit sterujący zezwoleniem pracy układu watchdog

0	stop watchdog timer
1	start watchdog timer

B2- B0 WDP2, WDP1, WDP0 – bity sterujące okresem czasu nadzoru przez układ watchdog

0	0	0	16 cykli
0	0	1	32 cykle
0	1	0	64 cykle
0	1	1	128 cykli
1	0	0	256 cykli
1	0	1	512 cykli

1	1	0	1024 cykle
1	1	1	2048 cykli

EEAR \$1E rejestr adresowy pamięci EEPROM

B6-B0 EEAR6-EEAR0 – rejestr adresu komórki pamięci EEPROM, do której żądany jest dostęp. Patrz tabela 4

EEDR \$1D rejestr danych pamięci EEPROM

B7-B0 EEDR7-EEDR0 – rejestr danych pamięci EEPROM. W przypadku zapisu do pamięci EEPROM, umieszczamy w nim wartość do zapamiętania. W cyklu odczytu wartość z pamięci EEPROM jest dostępna w tym rejestrze

EECR \$1C rejestr kontrolny pamięci EEPROM

B3 EERIE – dotyczy tylko ATtiny12/15. Jeśli bit jest równy 1 to jest zezwolenie na przerwanie spowodowane osiągnięciem gotowości pamięci EEPROM do powtórnej operacji zapisu. Przerwanie jest wywoływane przez sprzętowe wyzerowanie bitu EEWE. Wywołany wektor przerwania znajduje się pod adresem \$004 dla ATtiny12 i \$006 dla ATtiny15.

B2 EEMWE – bit zabezpieczający przed dokonaniem przypadkowego zapisu do pamięci EEPROM. Aby dokonać zapisu należy ustawić w stan wysoki bit EEMWE, w ciągu 4 cykli zegarowych ustawić bit EEWE, aby zapisać pod adres zawarty w EEAR bajt umieszczony w EEDR. Jeśli nie ustawimy bitu EEMWE, to zmiana bitu EEWE zostanie zignorowana

B1 EEWE – bit-znacznik zezwolenia na zapis do pamięci EEPROM. Po załadowaniu adresu i danej do rejestrów EEAR i EEDR, ustawienie bitu EEWE powoduje uruchomienie cyklu zapisu do pamięci. Po zakończeniu cyklu zapisu bit zostaje sprzętowo wyzerowany i można przeprowadzić kolejny cykl dostępu. Dopóki bit nie zostanie wyzerowany przez mikrokontroler nie można przeprowadzić kolejnej operacji zapisu

B0 EERE – bit-znacznik zezwolenia na odczyt z pamięci EEPROM. Po załadowaniu adresu do rejestru EEAR

ustawienie bitu EEWE powoduje uruchomienie cyklu odczytu z pamięci. Po zakończeniu cyklu odczytu, bit zostaje sprzętowo wyzerowany i można odczytać wartość wskazanej komórki pamięci EEPROM z rejestru EEDR

PORTB \$18 rejestr danych portu B

B4-B0 PORTB4-PORTB0 – bity rejestru danych wyjściowych portu B.
W trybie wyjścia steruje poziomem na linii portu

1	wyjście x w stanie wysokim
0	wyjście x w stanie niskim

W trybie wejścia steruje podciąganiem linii portu

1	wejście x z podciąganiem do stanu H
0	wejście x bez podciągania

DDRB \$17 rejestr kierunku portu B

B5-B0 DDRB5-DDRB0 – bity sterujące kierunkiem linii portu.

1	linia portu pracuje jako wyjście
0	linia portu pracuje jako wejście

PINB \$16 rejestr wejściowy portu B

B5-B0 PINB5-PINB0 – bity rejestru odczytu stanu linii wejściowych. Odczyt następuje bezpośrednio z wyprowadzenia przez bramkę z przerzutnikiem schmidta. W przypadku pracy linii portu jako wyjście, odczyt nie musi być zgodny z wartościami, zapisanymi w rejestrze danych wyjściowych

ACSR \$08 rejestr kontrolny i statusu komparatora analogowego

B7 ACD – bit sterujący załączeniem zasilania komparatora analogowego. Jeśli ACD=1, to komparator jest wyłączony. Przed wyłączeniem komparatora musimy zablokować przerwanie, poprzez wyzerowanie bitu ACIE

B6 ACBG – dotyczy tylko ATtiny12/15. Bit równy 1 dołącza do porównania źródła napięcia odniesienia 1,22 V +/- 0,05 V zamiast napięcia z wejścia AIN0

B5 ACO – bit połączony bezpośrednio z wyjściem komparatora. Odzworowuje wynik porównania napięć wejściowych

- B4 **ACI** – flaga przerywania od komparatora analogowego. Ustawiana jest w momencie spełnienia warunku wybranego za pomocą bitów ACIS1, ACIS0
- B3 **ACIE** – bit zezwolenia na przerywania od komparatora. Ustawienia tego bitu w stan wysoki oznacza zezwolenie na przerywanie od komparatora
- B1, B0 **ACIS1, ACIS0** – bity wybierające warunek przerywania od komparatora
- | | | |
|---|---|--|
| 0 | 0 | przerywanie wywoływane wystąpieniem zbocza na wyjściu komparatora |
| 0 | 1 | nie wykorzystane |
| 1 | 0 | przerywanie wywoływane zboczem opadającym na wyjściu komparatora |
| 1 | 1 | przerywanie wywoływane zboczem narastającym na wyjściu komparatora |

ADMUX \$07 rejestr sterowania multiplekserami przetwornika A/C

B7, B6 **REFS1, REFS0** Wybór napięcia odniesienia

0	0	Vref= Vcc
0	1	Vref=VPB0
1	0	Vref=INTVref (2,56V)
1	1	Vref=INTVref (2,56V)+ podłączenie do linii PB0 umożliwiające dodanie kondensatora, filtrującego napięcie odniesienia

Wprowadzona zmiana nie odniesie skutku, dopóki przetwarzanie nie spowoduje ustawienia flagi ADIF. Pierwsza konwersja po wprowadzeniu i przyjęciu zmiany będzie trwała 25 cykli zegarowych

B5 **ADLAR** – powoduje zmianę sposobu zwracania danych w rejestrach ADCH, ADCL. Dla znacznika równego „0” wynik jest równany do najmłodszego bitu (najstarsze bity rejestru ADCH bez znaczenia). Dla bitu równego „1” równanie wyniku odbywa się do najstarszego bitu rejestru (Najmłodsze bity rejestru ADCL bez znaczenia)

B2-B0 **MUX2, MUX1, MUX0** – Wybór dołączonych wejść przetwornika

	wejście		nieodwracające	odwracające	wzmocnienie
	0	0	0	ADC0 (PB5)	x1
	0	0	1	ADC1 (PB2)	x1

0	1	0	ADC2 (PB3)	x1
0	1	1	ADC3 (PB4)	x1
1	0	0	ADC2 (PB3) ADC2 (PB3)	x1
1	0	1	ADC2 (PB3) ADC2 (PB3)	x20
1	1	0	ADC2 (PB3) ADC3 (PB4)	x1
1	1	1	ADC2 (PB3) ADC3 (PB4)	x20

ADCSR \$06 rejestr stanu przetwornika A/C

- B7** **ADEN** – bit załącza przetwornik A/C. Wyzerowanie bitu powoduje natychmiastowe przerwanie przetwarzania
- B6** **ADSC** – ustawienie tego bitu powoduje start konwersji. Dla trybu pojedynczej konwersji bit zostaje wyzerowany po jej zakończeniu.
- B5** **ADFR** – bit ustawienia ciągłego (wolnobieżnego) przetwarzania. Dla bitu równego 1 kolejne cykle przetwarzania następują automatycznie. Ciąg konwersji trwa dopóty, dopóki nie wyzerujemy bitu ADSC lub ADEN
- B4** **ADIF** – znacznik (flaga) zgłoszenia przerwania. Ustawiany jest po każdym zakończeniu przetwarzania
- B3** **ADIE** – bit zezwolenia na przerwania od przetwornika A/C. Jeśli bit jest równy, 1 to w wyniku ustawienia flagi ADIF zostaje wywołane wektora przerwania nr 9 pod adresem \$008

B2-B0 **ADPS2, ADPS1, ADPS0** – wybór stopnia podziału preskalera dla przetwornika A/C

0	0	0	$F_{ADC}=CK/2$
0	0	1	$F_{ADC}=CK/2$
0	1	0	$F_{ADC}=CK/4$
0	1	1	$F_{ADC}=CK/8$
1	0	0	$F_{ADC}=CK/16$
1	0	1	$F_{ADC}=CK/32$
1	1	0	$F_{ADC}=CK/64$
1	1	1	$F_{ADC}=CK/128$

ADCH \$05 starsza część rejestru danych przetwornika A/C

B7-B0 Starszy bajt wyniku przetwarzania. Dla bitu ADLAR=1 zawiera 8 najstarszych bitów wyniku. Dla ADLAR=0 na pozycjach 1 i 0 są zapisywane dwa najstarsze bity wyniku, zaś pozostałe bity rejestru są niewykorzystywane.

ADCL **\$04 młodsza część rejestru danych przetwornika A/C**
B7-B0 Młodszy bajt wyniku przetwarzania. Dla bitu ADLAR=0 zawiera 8
najmłodszych bitów wyniku. Dla ADLAR=1 na pozycjach 7 i 6 są
zapisywane dwa najmłodsze bity wyniku, zaś pozostałe bity reje-
stru są niewykorzystywane.

Lista rozkazów

Lista rozkazów mikrokontrolerów w zależności od budowy, zawiera 90 lub 118 instrukcji. W pierwszej grupie kontrolerów o liście, składającej się z 90 rozkazów znajdują się ATtiny10/11/12/15. Do grupy kontrolerów o liście 118 rozkazów należą AT90S2323/2343, ATtiny22. Większość rozkazów jest wykonywana w jednym okresie zegarowym. Poniżej przedstawiony jest dokładny opis dostępnych instrukcji. Zostały one zestawione alfabetycznie wraz z opisem.

Opis:

(118) – Rozkaz wykonywany tylko przez AT90S2323/2343 i ATtiny22.

Rd – Rejestr R0-R31 lub R16-R31

Rr – Rejestr R0-R31

b – Stała 0-7 (adres bitu w bajcie)

s – Stała 0-7 (adres bitu w bajcie statusu)

P – Stała 0-31 lub 0-63 (adres portu)

K – Stała wartość 0-255 lub 0-63 (wartość bezpośrednia)

k – Stała, zakres zależy od instrukcji (przesunięcie względem bieżącego adresu)

ADC – dodaj z przeniesieniem dwa rejestry

Do zawartości rejestru Rd dodawana jest zawartość rejestru Rr i znacznika C. Wynik jest umieszczany w rejestrze Rd. Zgodnie z wynikiem operacji są ustawiane znaczniki: Z, C, N, V, H.

ADC Rd(0-31),Rr(0-31)

Operacja: $Rd \leftarrow Rd + Rr + C$

Liczba cykli: 1

Kod:

0	0	0	1	1	1	Rr	Rd	Rr
---	---	---	---	---	---	----	----	----

I	T	H	S	V	N	Z	C
-	-	\Leftrightarrow	\Leftrightarrow	\Leftrightarrow	\Leftrightarrow	\Leftrightarrow	\Leftrightarrow

ADD – dodaj dwa rejestry

Do zawartości rejestru Rd dodawana jest zawartość rejestru Rr. Wynik jest umieszczany w rejestrze Rd. Zgodnie z wynikiem operacji są ustawiane znaczniki: Z, C, N, V, H.

ADD Rd(0-31),Rr(0-31)

Operacja: $Rd \leftarrow Rd + Rr$

Liczba cykli: 1

Kod:

0	0	0	0	1	1	Rr	Rd	Rr
---	---	---	---	---	---	----	----	----

I	T	H	S	V	N	Z	C
-	-	\leftrightarrow	\leftrightarrow	\leftrightarrow	\leftrightarrow	\leftrightarrow	\leftrightarrow

ADIW – (118) dodaj stałą do słowa

Do zawartości szesnastobitowego rejestru Rd dodawana jest wartość bezpośrednia, z zakresu 0-63. Wynik jest umieszczany w rejestrze Rdh:Rdl. Bity „d” wskazują parę rejestrów, której dotyczy rozkaz. Rozkaz dotyczy tylko czterech par rejestrów: (d=0) R24:R25, (d=1) R26:R27, (d=2) R28:R29, (d=3) R30:R31. Zgodnie z wynikiem operacji są ustawiane znaczniki: S, V, N, Z, C.

ADIW Rdh:Rdl(0-31),K(0-63)

Operacja: $Rdh:Rdl \leftarrow Rdh:Rdl + K$

Liczba cykli: 2

Kod:

1	0	0	1	0	1	1	0	K	d	K
---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	\leftrightarrow	\leftrightarrow	\leftrightarrow	\leftrightarrow	\leftrightarrow

AND – iloczyn logiczny dwóch rejestrów

Wykonywany jest iloczyn logiczny rejestrów Rd i Rr. Wynik jest umieszczany w rejestrze Rd. Zgodnie z wynikiem operacji są ustawiane znaczniki: Z, N, V.

AND Rd(0-31),Rr(0-31)

Operacja: $Rd \leftarrow Rd \cdot Rr$

Liczba cykli: 1

Kod:

0	0	1	0	0	0	Rr	Rd	Rr
---	---	---	---	---	---	----	----	----

I	T	H	S	V	N	Z	C
-	-	-	\leftrightarrow	0	-	-	-

ANDI – iloczyn logiczny rejestru i stałej

Wykonywany jest iloczyn logiczny rejestru Rd i stałej K. Wynik jest umieszczany w rejestrze Rd. Zgodnie z wynikiem operacji są ustawiane znaczniki: Z, N, V. Operacja jest wykonywana tylko dla rejestrów R16-R31.

ANDI Rd(16-31),K(0-255)

Operacja: $Rd \leftarrow Rd \cdot Rr$

Liczba cykli: 1

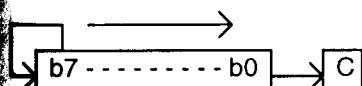
Kod:

0	1	1	1	K(7-4)				Rd				K(3-0)			
---	---	---	---	--------	--	--	--	----	--	--	--	--------	--	--	--

I	T	H	S	V	N	Z	C
-	-	-	\leftrightarrow	0	\leftrightarrow	\leftrightarrow	-

ASR – arytmetyczne przesunięcie w prawo

Powoduje przesunięcie zawartości rejestru w prawo. Na najstarszą pozycję rejestru (B7) zostaje nie zmieniony, wartość z najmłodszej pozycji zmienia wartość znacznika C. Instrukcja zmienia znaczniki Z, N, C, V.



ASR Rd(0-31)

Operacja: $Rd(n) \leftarrow Rd(n+1)$, $n=0..6$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	Rd				0	1	0	1
---	---	---	---	---	---	---	----	--	--	--	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	\leftrightarrow	\leftrightarrow	\leftrightarrow	\leftrightarrow	\leftrightarrow

BCLR – zeruj wskaźnik

Rozkaz zeruje wybrany wskaźnik w rejestrze SREG.

BCLR s(0-7)

Operacja: $SREG(s) \leftarrow 0$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	1	s	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
↔	↔	↔	↔	↔	↔	↔	↔

BLD – zapis bitu T do rejestru

Przepisuje wartość bitu T do bitu „b” w rejestrze „Rd”.

BLD Rd(0-31),b(0-7)

Operacja: $Rd(b) \leftarrow T$

Liczba cykli: 1/2

Kod:

1	1	1	1	1	0	0	Rd	0	B
---	---	---	---	---	---	---	----	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRBC – skocz względnie jeśli wskaźnik stanu jest wyzerowany

Jeśli wskaźnik stanu s w rejestrze SREG jest wyzerowany, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle.

BRBC s(0-7),k((+63) - (-64))

Operacja: jeśli $(SREG(s) = 0)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	1	K	s
---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRBS – skocz względnie, jeśli wskaźnik stanu jest ustawiony

Jeśli wskaźnik stanu s w rejestrze SREG jest ustawiony, to następuje skok względny (przemieszczenie względem bieżącego adresu).

Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle.

BRBS $s(0-7), k((+63) - (-64))$

Operacja: jeśli $(SREG(s) = 1)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	0	k	s
---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRCC – skocz względnie, jeśli znacznik przeniesienia wyzerowany

Jeśli wskaźnik stanu C jest wyzerowany, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby zapisanej, za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle.

Rozkaz jest pochodną rozkazu BRBC

BRCC $k((+63) - (-64))$

Operacja: jeśli $(C = 0)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	1	k	0	0	0
---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRCS – skocz względnie, jeśli znacznik przeniesienia ustawiony

Jeśli wskaźnik stanu C jest ustawiony, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle.

Rozkaz jest pochodną rozkazu BRBS

BRCS $k((+63) - (-64))$

Operacja: jeśli $(C = 1)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	0	k				0	0	0
---	---	---	---	---	---	---	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BREQ – skocz, jeśli równe

Jeśli wskaźnik stanu Z jest ustawiony, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 4 2 cykle. Rozkaz jest pochodną rozkazu BRBS

BREQ k((+63) - (-64))

Operacja: jeśli (Z = 1) to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	0	k				0	0	1
---	---	---	---	---	---	---	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRGE – skocz, jeśli większy lub równy ze znakiem

Jeśli wskaźnik stanu S jest wyzerowany, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBC

BRGE k((+63) - (-64))

Operacja: jeśli (S = 0) to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	1	k				1	0	0
---	---	---	---	---	---	---	--	--	--	---	---	---

I	T	H	S	V	N	Z	C

BRHC – skocz, jeśli wskaźnik przeniesienia pomocniczego wyzerowany

Jeśli wskaźnik stanu H jest wyzerowany, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBC

BRHC $k((+63) - (-64))$

Operacja: jeśli $(H = 0)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	1	k							1	0	1
---	---	---	---	---	---	---	--	--	--	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRHS – skocz, jeśli wskaźnik przeniesienia pomocniczego ustawiony

Jeśli wskaźnik stanu H jest ustawiony, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBS

BRHS $k((+63) - (-64))$

Operacja: jeśli $(H = 1)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	0	k							1	0	1
---	---	---	---	---	---	---	--	--	--	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRID – skocz, jeśli przerwania zablokowane

Jeśli wskaźnik zezwolenia na przerwania I jest wyzerowany, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, za-

pisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBC

BRID $k((+63) - (-64))$

Operacja: jeśli $(I = 0)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	1	k				1	1	1
---	---	---	---	---	---	---	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRIE – skocz, jeśli przerwania odblokowane

Jeśli wskaźnik zezwolenia na przerwania I jest ustawiony, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle.

Rozkaz jest pochodną rozkazu BRBS

BRIE $k((+63) - (-64))$

Operacja: jeśli $(I = 1)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	0	k				1	1	1
---	---	---	---	---	---	---	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRLO – skocz, jeśli mniejszy

Jeśli wskaźnik przeniesienia C jest ustawiony, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBS

BRLO $k((+63) - (-64))$

Operacja: jeśli $(C = 1)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	0	k	0	0	0
---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRLT – skocz, jeśli mniejszy niż zero, ze znakiem

Jeśli wskaźnik S jest wyzerowany, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBC

BRLT $k((+63) - (-64))$ Operacja: jeśli $(S = 0)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	0	k	1	0	0
---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRMI – skocz, jeśli ujemny

Jeśli wynik ostatniego działania jest ujemny, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBC

BRMI $k((+63) - (-64))$ Operacja: jeśli $(I = 0)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	0	k	0	1	0
---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRNE – skocz, jeśli różne

Jeśli wskaźnik Z jest wyzerowany, to następuje skok względny (przeszyczenie względem bieżącego adresu). Przeszyczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBC

BRNE k((+63) - (-64))

Operacja: jeśli (Z = 0) to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	1	k					0	0	1
---	---	---	---	---	---	---	--	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRPL – skocz, jeśli dodatni

Jeśli wskaźnik N jest wyzerowany, to następuje skok względny (przeszyczenie względem bieżącego adresu). Przeszyczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBC

BRPL k((+63) - (-64))

Operacja: jeśli (C = 0) to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	1	k					0	1	0
---	---	---	---	---	---	---	--	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRSB – skocz, jeśli nie mniejszy

Jeśli wskaźnik C jest wyzerowany, to następuje skok względny (przeszyczenie względem bieżącego adresu). Przeszyczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBC

BRSH $k((+63) - (-64))$

Operacja: jeśli $(C = 0)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	1	k				0	0	0
---	---	---	---	---	---	---	--	--	--	---	---	---

I	T	II	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRTC – skocz, jeśli wskaźnik T wyzerowany

Jeśli wskaźnik T jest wyzerowany, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBC

BRTC $k((+63) - (-64))$

Operacja: jeśli $(T = 0)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	1	k				1	1	0
---	---	---	---	---	---	---	--	--	--	---	---	---

I	T	II	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRTS – skocz, jeśli wskaźnik T ustawiony

Jeśli wskaźnik T jest ustawiony, to następuje skok względny (przemieszczenie względem bieżącego adresu). Przemieszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBS

BRTS $k((+63) - (-64))$

Operacja: jeśli $(T = 1)$ to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	0	k						1	1	0
---	---	---	---	---	---	---	--	--	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRVC – skocz, jeśli nie nastąpiło przepełnienie

Jeśli wskaźnik V jest wyzerowany, to następuje skok względny (przeszczenie względem bieżącego adresu). Przeszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego, czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBC

BRVC k((+63) - (-64))

Operacja: jeśli (V = 0) to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	1	k						0	1	1
---	---	---	---	---	---	---	--	--	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BRVS – skocz, jeśli nastąpiło przepełnienie

Jeśli wskaźnik V jest ustawiony, to następuje skok względny (przeszczenie względem bieżącego adresu). Przeszczenie jest zapisane na siedmiu bitach w postaci liczby, zapisanej za pomocą kodu U2. Dla warunku spełnionego czas trwania rozkazu wynosi 2 cykle. Rozkaz jest pochodną rozkazu BRBS

BRVS k((+63) - (-64))

Operacja: jeśli (V = 1) to $PC \leftarrow PC + k + 1$

Liczba cykli: 1/2

Kod:

1	1	1	1	0	0	k						0	1	1
---	---	---	---	---	---	---	--	--	--	--	--	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

BSET – ustaw wskaźnik

Ustawia wybrany wskaźnik z rejestru SREG.

BSET s(0-7)Operacja: $s \leftarrow 1$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	0	s	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
↔	↔	↔	↔	↔	↔	↔	↔

BST – załaduj bit do znacznika T

Ładuje do bitu znacznika T SREG(6) wartość bitu b z rejestru Rr.

BST Rr(0-31),b(0-7)Operacja: $T \leftarrow Rr(b)$

Liczba cykli: 1

Kod:

1	1	1	1	1	0	1	Rr	0	B
---	---	---	---	---	---	---	----	---	---

I	T	H	S	V	N	Z	C
-	↔	-	-	-	-	-	-

CBI – zeruj bit w rejestrze I/O

Zeruje bit b w porcie P, znajdującym się w przestrzeni adresowej wejścia/wyjścia.

CBI P(0-31),b(0-7)Operacja: $P(b) \leftarrow 0$

Liczba cykli: 2

Kod:

1	0	0	1	1	0	0	0	P	B
---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

CBR – zeruj bity w rejestrze

Zeruje bity w rejestrze Rd, zgodnie z maską K. Zerowane bity wskazujemy poprzez ustawienie bitów, które mają zostać wyzerowane.

CBR Rd(16-31),K(0-255)

Operacja: $P(b) \leftarrow 0$

Liczba cykli: 1

Kod:

0	1	1	1	K	Rd	K
---	---	---	---	---	----	---

I	T	H	S	V	N	Z	C
-	-	-	\leftrightarrow	0	\leftrightarrow	\leftrightarrow	-

CLC – zeruj wskaźnik przeniesienia

Rozkaz zeruje wskaźnik C w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BCLR.

CLC

Operacja: $C \leftarrow 0$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	1	0	0	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	0

CLH – zeruj wskaźnik przeniesienia pomocniczego

Rozkaz zeruje wskaźnik H w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BCLR.

CLH

Operacja: $H \leftarrow 0$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	1	1	0	1	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	0	0	0	1	-

CLI – zablokuj przerwania

Rozkaz zeruje wskaźnik I w rejestrze SREG przez co blokuje przerwania. Rozkaz stanowi pochodną rozkazu BCLR.

CLIOperacja: $I \leftarrow 0$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	1	1	1	1	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
0	-	-	-	-	-	-	-

CLN – zeruj wskaźnik wartości ujemnej

Rozkaz zeruje wskaźnik N w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BCLR.

CLNOperacja: $N \leftarrow 0$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	1	0	1	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	0	-	-

CLR – zeruj rejestr

Instrukcja zeruje wskazany rejestr Rd. Stanowi on pochodną rozkazu EOR (EX-OR).

CLR Rd(0 - 31)

Operacja: $Rd \leftarrow 0$

Liczba cykli: 1

Kod: dla Rd=Rr

0	0	1	0	0	1	Rd	Rr	Rd
---	---	---	---	---	---	----	----	----

I	T	H	S	V	N	Z	C
-	-	-	0	0	0	1	-

CLS – zeruj wskaźnik znaku

Rozkaz zeruje wskaźnik S w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BCLR.

CLSOperacja: $S \leftarrow 0$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	1	1	0	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	0	-	-	-	-

CLT – zeruj wskaźnik kopii bitu T

Rozkaz zeruje wskaźnik T w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BCLR.

CLTOperacja: $T \leftarrow 0$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	1	1	1	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	0	-	-	-	-	-	-

CLV – zeruj wskaźnik przepelnienia uzupełnienia do dwóch

Rozkaz zeruje wskaźnik V w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BCLR.

CLVOperacja: $V \leftarrow 0$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	1	0	1	1	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	0	-	-	-

CLZ – zeruj wskaźnik zera

Rozkaz zeruje wskaźnik Z w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BCLR.

Kod:

1	0	0	1	0	1	0	0	1	0	0	1	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	0	-

Kod:

1	0	0	1	0	1	0	Rd	0	0	0	0
---	---	---	---	---	---	---	----	---	---	---	---

I	T	II	S	V	N	Z	C
-	-	-	\Leftrightarrow	0	\Leftrightarrow	\Leftrightarrow	1

Kod:

0	0	0	1	0	1	Rr	Rd	Rr
---	---	---	---	---	---	----	----	----

[illegible]

CPC – porównaj rejestr z rejestrem i przeniesieniem

Rozkaz porównuje rejestr z rejestrem i znacznikiem C. Porównanie polega na odjęciu wartości rejestru Rr i znacznika C od Rd, oraz na ustawieniu bitów znaczników, odpowiednio do wyniku tej operacji. Stanowi ona odpowiednik rozkazu SBC, w którym wynik operacji odejmowania nie zostaje nigdzie zapisany. Operacja ustawia znaczniki Z, N, V, C, H.

CPC Rd(0-31),Rr(0-31)

Operacja: Rd - Rr - C

Liczba cykli: 1

Kod:

0	0	0	0	0	1	Rr	Rd	Rr
---	---	---	---	---	---	----	----	----

I	T	II	S	V	N	Z	C
-	-	<>	<>	<>	<>	<>	<>

CPI – porównaj rejestr ze stałą

Rozkaz porównuje rejestr ze stałą. Porównanie polega na odjęciu wartości stałej K od rejestru Rd i ustawieniu bitów znaczników, odpowiednio do wyniku tej operacji. Stanowi ona odpowiednik rozkazu SUB, w którym wynik operacji odejmowania nie zostaje nigdzie zapisany. Operacja ustawia znaczniki Z, N, V, C, H.

CPI Rd(0-31),K(0-255)

Operacja: Rd - K

Liczba cykli: 1

Kod:

0	0	1	1	K	Rd	K
---	---	---	---	---	----	---

I	T	II	S	V	N	Z	C
-	-	<>	<>	<>	<>	<>	<>

CPSE – (118) porównaj, skocz jeśli równe

Rozkaz porównuje dwa rejestry. Porównanie polega na odjęciu wartości rejestru Rr od rejestru Rd. Nie zostają ustawione żadne bity znaczników. Jeśli argumenty są sobie równe, to adres licznika rozkazów PC jest zwiększany o 2 lub 3. Następuje ominięcie następnego rozkazu, niezależnie czy jest on o rozmiarze jednego, czy dwu słów (dotyczy ko-

lejszych wersji mikrokontrolerów). Czas trwania rozkazu dla argumentów równych wynosi 2 cykle zegarowe.

CPSE Rd(0-31),Rr(0-31)

Operacja: jeśli (Rd = Rr) to $PC \leftarrow PC + 2$ lub 3

Liczba cykli: 1/2

Kod:

0	0	0	1	0	0	Rr		Rd		Rr
---	---	---	---	---	---	----	--	----	--	----

I	T	II	S	V	N	Z	C
-	-	-	-	-	-	-	-

DEC – zmniejsz o jeden

Rozkaz zmniejsza o jeden zawartość wskazanego rejestru. Rozkaz modyfikuje wskaźniki Z, N, V.

DEC Rd(0-31)

Operacja: $Rd \leftarrow Rd - 1$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0		Rd		1	0	1	0
---	---	---	---	---	---	---	--	----	--	---	---	---	---

I	T	II	S	V	N	Z	C
-	-	-	↔	↔	↔	↔	-

EOR – oblicz różnicę symetryczną (EX-OR)

Rozkaz wykonuje operację EX-OR na bitach wskazanych rejestrów.

Rozkaz zmienia wskaźniki Z, N, V.

EOR Rd(0-31),Rr(0-31)

Operacja: $Rd \leftarrow Rd \text{ EX-OR } Rr$

Liczba cykli: 1

Kod:

0	0	1	0	0	1	Rr		Rd		Rr
---	---	---	---	---	---	----	--	----	--	----

I	T	H	S	V	N	Z	C
-	-	-	↔	0	↔	↔	-

ICALL (118) – skocz ze śladem pośrednio pod adres, zawarty w rejestrze Z

Rozkaz skoku ze śladem pod adres bezwzględny, zawarty w szesnastobitowym rejestrze Z (R30:R31). Instrukcja powoduje odłożenie na stos 16-bitowego adresu bieżącego, zmniejszenie wskaźnika wierzchołka stosu o 2, a następnie przejście do wywoływanej procedury.

ICALL K(0-65535)

Operacja: $STACK \leftarrow PC+1$, $SP \leftarrow SP-2 \leftarrow PC \leftarrow Z$

Liczba cykli: 3

Kod:

1	0	0	1	0	1	0	1	X	X	X	X	1	0	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

IJMP – (118) skocz pod adres pośredni, zawarty w rejestrze Z

Rozkaz skoku pod adres bezwzględny zawarty w szesnastobitowym rejestrze Z (R30:R31).

IJMP K(0-65535)

Operacja: $PC \leftarrow Z(15-0)$

Liczba cykli: 2

Kod:

1	0	0	1	0	1	0	0	X	X	X	X	1	0	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

IN – odczytaj port

Rozkaz przeladowuje zawartość rejestru z obszaru adresowego wejścia/wyjścia do rejestru roboczego.

IN Rd(0-31),P(0-63)

Operacja: $Rd \leftarrow P$

Liczba cykli: 1

Kod:

I	0	I	I	0	P	Rd	P
---	---	---	---	---	---	----	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

INC – zwiększ o jeden

Rozkaz zwiększa o jeden zawartość wskazanego rejestru. Rozkaz modyfikuje wskaźniki Z, N, V.

INC Rd(0-31)

Operacja: $Rd \leftarrow Rd + 1$

Liczba cykli: 1

Kod:

I	0	0	I	0	I	0	Rd	0	0	I	I
---	---	---	---	---	---	---	----	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	↔	↔	↔	↔	-

LD – załaduj rejestr pośrednio z wykorzystaniem rejestru X

Rozkaz ładuje do rejestru Rd wartość rejestru wskazywanego adresem, zawartym w rejestrze indeksowym. Pamięć rejestrów roboczych i rejestrów wejścia/wyjścia od strony adresowania indeksowego, przedstawia się jako obszar ciągły. Pod adresami \$0-\$1F umieszczone są rejestry robocze, a od adresu \$20 do \$5F, umieszczony jest obszar rejestrów wejścia/wyjścia. Od adresu \$60 rozpoczyna się obszar pamięci RAM. Aby obliczyć adres indeksowy rejestru wejścia/wyjścia, do adresu w przestrzeni adresowej I/O dodajemy \$20. W przypadku pamięci danych RAM do adresu pierwotnego dodajemy \$60. Rozkaz ten może być użyty w trzech trybach adresowania dla rejestru X i w czterech dla rejestrów Y i Z.

Tryby te to:

- adresowanie pośrednie (rejestr roboczy jest ładowany zawartością pamięci, wskazywaną przez rejestr indeksowy)
- adresowanie pośrednie z postinkrementacją (rejestr roboczy jest ładowany zawartością pamięci, wskazywaną przez rejestr indeksowy, a następnie adres w rejestrze indeksowym jest zwiększany o 1)
- adresowanie pośrednie z predekrementacją (adres w rejestrze indeksowym jest zmniejszany o 1, a następnie jest ładowany rejestr roboczy zawartością pamięci, wskazywaną przez rejestr indeksowy)

– adresowanie pośrednie z przemieszczeniem (rejestr roboczy jest ładowany zawartością pamięci, wskazywaną przez adres wyznaczany rejestrem roboczym i stałą przemieszczenia, zawartą w rozkazie)
W żadnym z rozkazów nie są zmieniane znaczniki.

(118) Tryb adresowania pośredniego zawartością rejestru X:

LD Rd(0-31), X

Operacja: $Rd \leftarrow (X)$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	0		Rd		1	1	0	0
---	---	---	---	---	---	---	--	----	--	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru X z postinkrementacją:

LD Rd(0-31), X+

Operacja: $Rd \leftarrow (X)$, $X \leftarrow X + 1$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	0		Rd		1	1	0	1
---	---	---	---	---	---	---	--	----	--	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru X z predekrementacją:

LD Rd(0-31), -X

Operacja: $X \leftarrow X - 1$, $Rd \leftarrow (X)$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	0		Rd		1	1	1	0
---	---	---	---	---	---	---	--	----	--	---	---	---	---

LD (LDD) – załaduj rejestr pośrednio z wykorzystaniem rejestru Y

(118) Tryb adresowania pośredniego zawartością rejestru Y:

LD Rd(0-31), Y

Operacja: $Rd \leftarrow (Y)$

Liczba cykli: 2

Kod:

1	0	0	0	0	0	0	0	Rd	1	0	0	0
---	---	---	---	---	---	---	---	----	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru Y z postinkrementacją:

LD Rd(0-31), Y+

Operacja: $Rd \leftarrow (Y)$, $Y \leftarrow Y + 1$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	0	0	Rd	1	0	0	1
---	---	---	---	---	---	---	---	----	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru Y z predekrementacją:

LD Rd(0-31), -Y

Operacja: $Y \leftarrow Y - 1$, $Rd \leftarrow (Y)$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	0	0	Rd	1	0	1	0
---	---	---	---	---	---	---	---	----	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru Y z przemieszczeniem:

LDD Rd(0-31), Y+q(0-63)

Operacja: $Rd \leftarrow (Y+q)$

Liczba cykli: 2

Kod:

1	0	q	0	q	0	0	0	Rd	1	q		
---	---	---	---	---	---	---	---	----	---	---	--	--

LD (LDD) – załaduj rejestr pośrednio z wykorzystaniem rejestru Z

Tryb adresowania pośredniego zawartością rejestru Z:

LD Rd(0-31), Z

Operacja: $Rd \leftarrow (Z)$

Liczba cykli: 2

Kod:

1	0	0	0	0	0	0	0	Rd	0	0	0	0
---	---	---	---	---	---	---	---	----	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru Z z postinkrementacją:

LD Rd(0-31), Z+

Operacja: $Rd \leftarrow (Z)$, $Z \leftarrow Z + 1$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	0		Rd		0	0	0	1
---	---	---	---	---	---	---	--	----	--	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru Z z predekrementacją:

LD Rd(0-31), -Z

Operacja: $Z \leftarrow Z - 1$, $Rd \leftarrow (Z)$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	0		Rd		0	0	1	0
---	---	---	---	---	---	---	--	----	--	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru Z z przemieszczeniem:

LDD Rd(0-31), Z+q(0-63)

Operacja: $Rd \leftarrow (Z+q)$

Liczba cykli: 2

Kod:

1	0	q	0	q	0		Rd		0		q
---	---	---	---	---	---	--	----	--	---	--	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

LDI – załaduj stałą do rejestru

Rozkaz ładuje wartość bezpośrednią K do wskazanego rejestru Rd.

LDI Rd(16-31), K

Operacja: $Rd \leftarrow K$

Liczba cykli: 1

Kod:

1	1	1	0	K				Rd				K			
I				T				H				S			
-				-				-				-			

LDS – (118) załaduj bajt z pamięci RAM do rejestru

Rozkaz ładuje spod podanego bezpośrednio 16 bitowego adresu bajt danej do rejestru roboczego. Rozkaz nie zmienia znaczników.

LDS Rd(16-31),k(0-65535)

Operacja: $Rd \leftarrow (k)$

Liczba cykli: 3

Kod:

1	0	0	1	0	0	0	Rd			0	0	0	0		
K				k			k			K					
I		T		H		S		V		N		Z		C	
-		-		-		-		-		-		-		-	

LPM – ładuj bajt pamięci programu

Rozkaz ładuje jeden bajt pamięci programu, wskazywany przez adres, zawarty w rejestrze Z do rejestru R0. Adres wskazuje kolejne bajty (nie słowa). Rozkaz nie zmienia znaczników.

LPM

Operacja: $R0 \leftarrow (Z)$

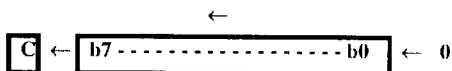
Liczba cykli: 3

Kod:

1	0	0	1	0	1	0	1	1	1	0	X	1	0	0	0
I				T				H				S			
-				-				-				-			

LSL – przesun logicznie w lewo

Rozkaz przesuwa logicznie zawartość rejestru Rd w lewo. Na najmłodszą pozycję wpisywane jest 0. W wyniku wykonania ustawiane są znaczniki H, Z, C, N, V. Rozkaz jest pochodną rozkazu dodawania ADD.



LSL Rd(0-31)

Operacja: $Rd(n+1) \leftarrow Rd(n)$, $Rd(0) \leftarrow 0$

Liczba cykli: 1

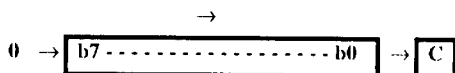
Kod: dla $Rd=Rr$

0	0	0	0	1	1	Rr	Rd				Rr			
---	---	---	---	---	---	----	----	--	--	--	----	--	--	--

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

LSR – przesun logicznie w prawo

Rozkaz przesun logicznie zawartość rejestru Rd w prawo. Na najstarszą pozycję wpisywane jest 0. W wyniku wykonania ustawiane są znaczniki Z, C, N, V.



LSR Rd(0-31)

Operacja: $Rd(n) \leftarrow Rd(n+1)$, $Rd(7) \leftarrow 0$

Liczba cykli: 1

Kod:

I	0	0	1	0	1	0	Rd				0	1	1	0
---	---	---	---	---	---	---	----	--	--	--	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	↔	↔	0	↔	↔

MOV – przepis zawartość z rejestru do rejestru

Rozkaz dokonuje przesłania między rejestrami roboczymi.

MOV Rd(0-31),Rr(0-31)

Operacja: $Rd \leftarrow Rr$

Liczba cykli: 1

Kod

0	0	1	0	1	1	Rr	Rd	Rr
---	---	---	---	---	---	----	----	----

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

NEG – oblicz uzupełnienie do dwóch

Oblicza wartość uzupełnienia kodu do dwóch.

Ustawia znaczniki Z, N, C, V.

NEG Rd(0-31)

Operacja: $Rd \leftarrow \$00 - Rd$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	Rd	0	0	0	1
---	---	---	---	---	---	---	----	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

NOP – nic nie rób

Rozkaz pusty (bez operacji).

NOP

Operacja:

Liczba cykli: 1

Kod:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

OR – oblicz sumę logiczną rejestrów

Rozkaz bitowy obliczający sumę logiczną dwóch rejestrów roboczych.

Wykonanie rozkazu zmienia znaczniki Z, N, V.

OR Rd(0-31), Rr(0-31)

Operacja: $Rd \leftarrow Rd \vee Rr$

Liczba cykli: 1

Kod:

0	0	1	0	1	0	Rr	Rd	Rd
---	---	---	---	---	---	----	----	----

I	T	H	S	V	N	Z	C
-	-	-	↔	0	↔	↔	-

ORI – oblicz sumę logiczną rejestru i stałej

Rozkaz bitowy obliczający sumę logiczną rejestru i stałej K. Wykonanie rozkazu zmienia znaczniki Z, N, V.

ORI Rd(16-31),K(0-255)

Operacja: $Rd \leftarrow Rd \vee K$

Liczba cykli: 1

Kod:

0	1	1	0	K	Rd	K
---	---	---	---	---	----	---

I	T	H	S	V	N	Z	C
-	-	-	↔	0	↔	↔	-

OUT – zapisz do portu

Rozkaz przeladowuje zawartość rejestru roboczego do rejestru z obszaru adresowego wejścia/wyjścia.

OUT P(0-63),Rd(0-31)

Operacja: $P \leftarrow Rd$

Liczba cykli: 1

Kod:

1	0	1	1	1	P	Rd	P
---	---	---	---	---	---	----	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

POP – (118) podnieś rejestr ze stosu

Rozkaz odtwarza zawartość rejestru ze stosu, zachowaną rozkazem PUSH. Podczas wykonania rozkazu najpierw zostaje zwiększony wskaźnik wierzchołka stosu, a następnie zostaje podniesiona zawartość wierzchołka stosu i załadowana do wskazanego rejestru.

POP Rd(0-31)

Operacja: $SP \leftarrow SP+1$, $Rd \leftarrow STACK$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	0		Rd		1	1	1	1
---	---	---	---	---	---	---	--	----	--	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

PUSH – (118) zapamiętaj rejestr na stosie

Rozkaz zachowuje zawartość wskazanego rejestru na stosie. Podczas wykonania rozkazu najpierw zostaje zapamiętana zawartość rejestru, a następnie zmniejszona o 1 zawartość wskaźnika wierzchołka stosu.

PUSH Rd(0-31)

Operacja: $STACK \leftarrow Rd$, $SP \leftarrow SP-1$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	1		Rd		1	1	1	1
---	---	---	---	---	---	---	--	----	--	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

RCALL – względne wywołanie podprogramu

Skok ze śladem. Rozkaz odkłada adres powrotu na stos (1 cykl zegara), a następnie „skacze” pod sumę adresu bieżącego, znajdującego się w liczniku programu PC i przemieszczenia zawartego w kodzie rozkazu (2 cykle zegarowe). Przemieszczenie zapisane jest w postaci kodu U2.

RCALL k(-32767 - 32768)

Operacja: $PC \leftarrow PC + k + 1$

Liczba cykli: 3

Kod:

1	1	0	1					k					
---	---	---	---	--	--	--	--	---	--	--	--	--	--

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

RET – powrót z podprogramu

Rozkaz powoduje zdjęcie ze stosu adresu powrotu, odłożonego za pomocą rozkazu RCALL, lub w trakcie wywołania obsługi przerwania.

RET

Operacja: $PC \leftarrow \text{STACK}$

Liczba cykli: 4

Kod:

1	0	0	1	0	1	0	1	0	0	0	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

RETI – powrót z obsługi przerwania

Rozkaz powoduje zdjęcie ze stosu adresu powrotu, odłożonego w trakcie wywołania obsługi przerwania, a następnie odblokowanie przerw poprzez ustawienie bitu I w rejestrze SREG.

RETI

Operacja: $PC \leftarrow \text{STACK}$

Liczba cykli: 4

Kod:

1	0	0	1	0	1	0	1	0	X	X	1	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

RJMP – skok względny

Skok względny. Rozkaz wykonuje skok pod sumę adresu bieżącego, znajdującego się w liczniku programu PC i przemieszczenia, zawartego w kodzie rozkazu (2 cykle zegarowe). Przemieszczenie zapisane jest w postaci kodu U2.

RJMP k(-32767 - 32768)

Operacja: $PC \leftarrow PC + k + 1$

Liczba cykli: 3

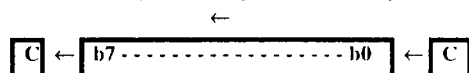
Kod:

1	1	0	0	k														i
---	---	---	---	---	--	--	--	--	--	--	--	--	--	--	--	--	--	---

I	T	II	S	V	N	Z	C
-	-	-	-	-	-	-	-

ROL – obróć w lewo wraz ze znacznikiem przeniesienia

Wykonywany jest obrót logiczny w lewo wraz ze znacznikiem C. Wszystkie bity zostają przesunięte o jedną pozycję w lewo, na najmłodszą pozycję rejestru jest wpisywana wartość znacznika C, a bit z najstarszej pozycji trafia do znacznika C. Rozkaz ustawia znaczniki Z, C, N, V. Rozkaz pochodny od instrukcji ADC



ROL Rd(0-31)

Operacja: $Rd(0) \leftarrow C$, $Rd(n+1) \leftarrow Rd(n)$, $C \leftarrow Rd(7)$

Liczba cykli: 1

Kod: dla Rd=Rr

0	0	0	1	1	1	Rr	Rd				Rr			
---	---	---	---	---	---	----	----	--	--	--	----	--	--	--

I	T	II	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

ROR – obróć w prawo wraz ze znacznikiem przeniesienia

Wykonywany jest obrót logiczny w prawo wraz ze znacznikiem C. Wszystkie bity zostają przesunięte o jedną pozycję w prawo, na najstarszą pozycję rejestru jest wpisywana wartość znacznika C, a bit z najmłodszej pozycji trafia do znacznika C. Rozkaz ustawia znaczniki Z, C, N, V. Rozkaz pochodny od instrukcji ADC



ROR Rd(0-31)

Operacja: $Rd(7) \leftarrow C$, $Rd(n-1) \leftarrow Rd(n)$, $C \leftarrow Rd(0)$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0		Rd		0	1	1	1
---	---	---	---	---	---	---	--	----	--	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	↔	↔	↔	↔	↔

SBC – odejmij z przeniesieniem dwa rejestry

Obliczana jest różnica między rejestrami z uwzględnieniem znacznika C. Rozkaz ustawia znaczniki Z, C, N, V, H.

SBC Rd(0-31),Rr(0-31)

Operacja: $Rd \leftarrow Rd - Rr - C$

Liczba cykli: 1

Kod:

0	0	0	0	1	0	Rr		Rd			Rr
---	---	---	---	---	---	----	--	----	--	--	----

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

SBCI – odejmij z przeniesieniem stałą od rejestru

Odejmowana jest od rejestru stała z uwzględnieniem znacznika C. Rozkaz ustawia znaczniki Z, C, N, V, H.

SBCI Rd(16-31),K

Operacja: $Rd \leftarrow Rd - Rr - C$

Liczba cykli: 1

Kod:

0	1	0	0		K		Rd		K
---	---	---	---	--	---	--	----	--	---

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

SBI – ustaw bit w rejestrze I/O

Ustawia bit b w porcie P, znajdującym się w przestrzeni adresowej wejścia/wyjścia. Rozkaz dotyczy obszaru I/O o adresach 0-31.

SBI P(0-31),b(0-7)

Operacja: $P(b) \leftarrow 1$

Liczba cykli: 2

Kod:

I	0	0	I	I	0	I	0		P		b
---	---	---	---	---	---	---	---	--	---	--	---

I	T	II	S	V	N	Z	C
-	-	-	-	-	-	-	-

SBIC – skocz, jeśli bit w rejestrze I/O wyzerowany

Rozkaz wykonuje skok, jeśli bit b w porcie P, znajdującym się w przestrzeni adresowej wejścia/wyjścia ma wartość 0. Jeśli zostaje spełniony warunek, to następuje zwiększenie adresu licznika programu PC o 2 lub 3, aby ominąć rozkaz umieszczony bezpośrednio za rozkazem skoku. Czas trwania rozkazu dla testowanego bitu równego 0 wynosi 2 cykle zegarowe. Rozkaz dotyczy obszaru I/O o adresach 0-31.

SBIC P(0-31),b(0-7)

Operacja: jeśli $(P(b) = 0)$ to $PC \leftarrow PC + 2$ lub 3

Liczba cykli: 1/2

Kod:

I	0	0	I	I	0	0	I		P		b
---	---	---	---	---	---	---	---	--	---	--	---

I	T	II	S	V	N	Z	C
-	-	-	-	-	-	-	-

SBIS – skocz, jeśli bit w rejestrze I/O ustawiony

Rozkaz wykonuje skok, jeśli bit b w porcie P, znajdującym się w przestrzeni adresowej wejścia/wyjścia ma wartość 1. Jeśli zostaje spełniony warunek, to następuje zwiększenie adresu licznika programu PC o 2 lub 3, aby ominąć rozkaz umieszczony bezpośrednio za rozkazem skoku. Czas trwania rozkazu dla testowanego bitu równego 1 wynosi 2 cykle zegarowe. Rozkaz dotyczy obszaru I/O o adresach 0-31.

SBIS P(0-31),b(0-7)

Operacja: jeśli $(P(b) = 1)$ to $PC \leftarrow PC + 2$ lub 3

Liczba cykli: 1/2

Kod:

I	0	0	I	I	0	I	I		P		b
---	---	---	---	---	---	---	---	--	---	--	---

I	T	II	S	V	N	Z	C
-	-	-	-	-	-	-	-

SBIW – (118) odejmij wartość bezpośrednią od pary rejestrów

Rozkaz odejmuje wartość bezpośrednią z zakresu 0-63 od jednej z czterech par rejestrów (rejestr 16-bitowego), umieszczonych w ostatnich bajtach rejestrów roboczych

SBIW RdI(0-3),K(0-63)

Operacja: $RdI : RdI \leftarrow RdI(24,26,28,30) - K(0-63)$

Liczba cykli: 2

Kod:

I	0	0	I	0	I	I	I	K	d	K
---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	<->	<->	<->	<->	<->

SBR – ustaw bity w rejestrze

Rozkaz bitowy obliczający sumę logiczną rejestru i stałej K. Wykonanie rozkazu zmienia znaczniki Z, N, V.

SBR Rd(16-31),K (0-255)

Operacja: $Rd \leftarrow Rd \vee K$

Liczba cykli: 1

Kod:

0	I	I	0	K	Rd	K
---	---	---	---	---	----	---

I	T	H	S	V	N	Z	C
-	-	-	<->	0	<->	<->	-

SBRC – skocz, jeśli bit w rejestrze jest wyzerowany

Rozkaz wykonuje skok, jeśli bit b w rejestrze Rd banku rejestrów roboczych ma wartość 0. Jeśli zostaje spełniony warunek, to następuje zwiększenie adresu licznika programu PC o 2 lub 3, aby ominąć rozkaz umieszczony bezpośrednio za rozkazem skoku. Czas trwania rozkazu dla testowanego bitu równego 0 wynosi 2 cykle zegarowe.

SBRC Rd(0-31),b(0-7)

Operacja: jeśli $(Rd(b) = 0)$ to $PC \leftarrow PC + 2$ lub 3

Liczba cykli: 1/2

Kod:

1	1	1	1	1	1	0	Rd	X	b
---	---	---	---	---	---	---	----	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

SBRS – skocz, jeśli bit w rejestrze jest ustawiony

Rozkaz wykonuje skok, jeśli bit b w rejestrze Rd banku rejestrów roboczych ma wartość 1. Jeśli zostaje spełniony warunek, to następuje zwiększenie adresu licznika programu PC o 2 lub 3, aby ominąć rozkaz umieszczony bezpośrednio za rozkazem skoku. Czas trwania rozkazu dla testowanego bitu równego 1 wynosi 2 cykle zegarowe.

SBRS Rd(0-31),b(0-7)

Operacja: jeśli $(Rd(b) = 1)$ to $PC \leftarrow PC + 2$ lub 3

Liczba cykli: 1/2

Kod:

1	1	1	1	1	1	1	Rd	X	b
---	---	---	---	---	---	---	----	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

SEC – ustaw wskaźnik przeniesienia

Rozkaz ustawia wskaźnik C w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BSET.

SEC

Operacja: $C \leftarrow 1$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	1

SEH – ustaw wskaźnik przeniesienia pomocniczego

Rozkaz ustawia wskaźnik H w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BSET.

SEHOperacja: $H \leftarrow 1$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	0	1	0	1	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	1	-	-	-	-	-

SEI – odblokuj przerwania

Rozkaz ustawia wskaźnik I w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BSET.

SEIOperacja: $I \leftarrow 1$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	0	1	1	1	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
1	-	-	-	-	-	-	-

SEN – ustaw wskaźnik wartości ujemnej

Rozkaz ustawia wskaźnik N w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BSET.

SENOperacja: $N \leftarrow 1$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	0	0	1	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	1	-	-

SER – ustaw rejestr

Rozkaz ładuje do rejestru Rd wartość \$FF (255 dec). Rozkaz stanowi pochodną od rozkazu LDI

SER Rd(16-31)

Operacja: $Rd \leftarrow \$FF$

Liczba cykli: 1

Kod:

1	1	1	0	1	1	1	1		Rd		1	1	1	1
---	---	---	---	---	---	---	---	--	----	--	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

SES – ustaw wskaźnik znaku

Rozkaz ustawia wskaźnik S w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BSET.

SES

Operacja: $S \leftarrow 1$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	0	1	0	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	1	-	-	-	-

SET – ustaw bit pomocniczy T

Rozkaz ustawia wskaźnik T w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BSET.

SET

Operacja: $T \leftarrow 1$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	0	1	1	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	1	-	-	-	-	-	-

SEV – ustaw wskaźnik przepełnienia uzupełnienia do dwóch

Rozkaz ustawia wskaźnik V w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BSET.

SEVOperacja: $V \leftarrow 1$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	0	0	1	1	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	1	-	-	-

SEZ – ustaw wskaźnik zera

Rozkaz ustawia wskaźnik Z w rejestrze SREG. Rozkaz stanowi pochodną rozkazu BSET.

SEZOperacja: $Z \leftarrow 1$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	1	-

SLEEP – przejdź w tryb pracy z obniżonym poborem mocy

Rozkaz wprowadza jednostkę centralną w tryb obniżonego poboru mocy. Wyprowadzenie może odbyć się tylko poprzez przerwanie, lub reset mikrokontrolera.

SLEEP

Operacja: uśpienie

Liczba cykli: 3

Kod:

1	0	0	1	0	1	0	1	1	0	0	X	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

ST – zapamiętaj pośrednio rejestr w pamięci RAM

Rozkaz zapisuje zawartość wskazanego rejestru pod adresem, wskazywanym zawartością jednego z trzech rejestrów indeksowych. Pamięć rejestrów roboczych i rejestrów wejścia/wyjścia od strony adresowania indeksowego, przedstawia się jako obszar ciągły. Pod adresami \$0-\$1F umieszczone są rejestry robocze, a od adresu \$20 do \$5F umieszczony jest obszar rejestrów wejścia/wyjścia. Od adresu \$60 rozpoczyna się obszar pamięci RAM. Aby obliczyć adres indeksowy rejestru wejścia/wyjścia, do adresu w przestrzeni adresowej I/O dodajemy \$20. W przypadku pamięci danych RAM do adresu pierwotnego dodajemy \$60. Rozkaz możemy użyć w czterech trybach adresowania w przypadku rejestrów Y i Z, oraz trzech dla rejestru X. Rozkaz nie zmienia znaczników.

Tryby te to:

- adresowanie pośrednie (zawartość rejestru roboczego jest ładowana do pamięci, wskazywanej przez rejestr indeksowy)
- adresowanie pośrednie z postinkrementacją (zawartość rejestru roboczego jest ładowana do pamięci, wskazywanej przez rejestr indeksowy, a następnie adres w rejestrze indeksowym jest zwiększany o 1)
- adresowanie pośrednie z predekrementacją (adres w rejestrze indeksowym jest zmniejszany o 1, a następnie zawartość rejestru roboczego jest ładowana do pamięci, wskazywanej przez rejestr indeksowy)
- adresowanie pośrednie z przemieszczeniem (zawartość rejestru roboczego jest ładowana do pamięci, wskazywanej przez sumę zawartości rejestru indeksowego i stałej przemieszczenia, zawartej w rozkazie)

W żadnym z rozkazów znaczniki nie są zmieniane.

ST (STD) – (118) Tryb adresowania pośredniego z wykorzystaniem rejestru X

Tryb adresowania pośredniego zawartością rejestru X:

ST X, Rd(0-31)

Operacja: (X) \leftarrow Rd

Liczba cykli: 2

Kod:

1	0	0	1	0	0	1		Rd		1	1	0	0
---	---	---	---	---	---	---	--	----	--	---	---	---	---

Tryb adresowania pośredniego zawartością rejestru X z postinkrementacją:

ST X+, Rd(0-31)

Operacja: (X) \leftarrow Rd ; X \leftarrow X + 1

Liczba cykli: 2

Kod:

1	0	0	1	0	0	1	Rd	1	1	0	1
---	---	---	---	---	---	---	----	---	---	---	---

Tryb adresowania pośredniego zawartością rejestru X z predekrementacją:

ST -X , Rd(0-31)

Operacja: $X \leftarrow X - 1$; $(X) \leftarrow Rd$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	1	Rd	1	1	1	0
---	---	---	---	---	---	---	----	---	---	---	---

ST (STD) – (118) zapamiętaj rejestr pośrednio z wykorzystaniem rejestru Y

Tryb adresowania pośredniego zawartością rejestru Y:

ST Y , Rd(0-31)

Operacja: $(Y) \leftarrow Rd$

Liczba cykli: 2

Kod:

1	0	0	0	0	0	1	Rd	1	0	0	0
---	---	---	---	---	---	---	----	---	---	---	---

Tryb adresowania pośredniego zawartością rejestru Y z postinkrementacją:

ST Y+ , Rd(0-31)

Operacja: $(Y) \leftarrow Rd$; $Y \leftarrow Y + 1$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	1	Rd	1	0	0	1
---	---	---	---	---	---	---	----	---	---	---	---

Tryb adresowania pośredniego zawartością rejestru Y z predekrementacją:

ST -Y , Rd(0-31)

Operacja: $Y \leftarrow Y - 1$; $(Y) \leftarrow Rd$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	1		Rd		1	0	1	0
---	---	---	---	---	---	---	--	----	--	---	---	---	---

Tryb adresowania pośredniego zawartością rejestru Y z przemieszczeniem:

STD Y+q(0-63) , Rd(0-31)

Operacja: $(Y+q) \leftarrow Rd$

Liczba cykli: 2

Kod:

1	0	q	0	Q	1		Rd		1		q
---	---	---	---	---	---	--	----	--	---	--	---

ST (STD) – zapamiętaj rejestr pośrednio z wykorzystaniem rejestru Z

Tryb adresowania pośredniego zawartością rejestru Z:

ST Z , Rd(0-31)

Operacja: $(Z) \leftarrow Rd$

Liczba cykli: 2

Kod:

1	0	0	0	0	0	1		Rd		0	0	0	0
---	---	---	---	---	---	---	--	----	--	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru Z z postinkrementacją:

ST Z+ , Rd(0-31)

Operacja: $(Z) \leftarrow Rd$; $Z \leftarrow Z + 1$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	1		Rd		0	0	0	1
---	---	---	---	---	---	---	--	----	--	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru Z z predekrementacją:

ST -Z , Rd(0-31)

Operacja: $Z \leftarrow Z - 1$; $(Z) \leftarrow Rd$

Liczba cykli: 2

Kod:

1	0	0	1	0	0	1		Rd		0	0	1	0
---	---	---	---	---	---	---	--	----	--	---	---	---	---

(118) Tryb adresowania pośredniego zawartością rejestru Z z przemieszczeniem:

STD Z+q(0-63) , Rd(0-31)

Operacja: $(Z+q) \leftarrow Rd$

Liczba cykli: 2

Kod:

1	0	q	0	q	1	Rd	0	q
---	---	---	---	---	---	----	---	---

I	T	II	S	V	N	Z	C
-	-	-	-	-	-	-	-

STS (118) – załaduj bezpośrednio rejestr do pamięci danych

Rozkaz ładuje zawartość rejestru Rd do pamięci, wskazywanej adresem bezpośrednim, zawartym w kodzie rozkazu. Pamięć rejestrów roboczych i rejestrów wejścia/wyjścia od strony adresowania indeksowego, przedstawia się jako obszar ciągły. Pod adresami \$0-\$1F umieszczone są rejestry robocze, a od adresu \$20 do \$5F umieszczony jest obszar rejestrów wejścia/wyjścia. Aby obliczyć adres indeksowy rejestru wejścia/wyjścia, do adresu w przestrzeni adresowej I/O dodajemy \$20.

STS k(0-65535) , Rd(0-31)

Operacja: $(Z) \leftarrow Rd$

Liczba cykli: 3

Kod:

1	0	0	1	0	0	1	Rd		0	0	0	0			
K				k			k		k						
I		T		II		S		V		N		Z		C	
-		-		-		-		-		-		-		-	

SUB – odejmij dwa rejestry

Od zawartości rejestru Rd odejmowana jest zawartość rejestru Rr. Wynik jest umieszczany w rejestrze Rd. Zgodnie z wynikiem operacji są ustawiane znaczniki: Z, C, N, V, H.

SUB Rd(0-31),Rr(0-31)

Operacja: $Rd \leftarrow Rd - Rr$

Liczba cykli: 1

Kod:

0	0	0	1	1	0	Rr		Rd		Rr
---	---	---	---	---	---	----	--	----	--	----

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

SUBI – odejmij stałą od rejestru

Wykonywana jest suma rejestru Rd i stałej K. Wynik jest umieszczany w rejestrze Rd. Zgodnie z wynikiem operacji są ustawiane znaczniki Z, N, C, V, H. Operacja jest wykonywana tylko dla rejestrów R16-R31.

SUBI Rd(16-31),K(0-255)

Operacja: $Rd \leftarrow Rd - K$

Liczba cykli: 1

Kod:

0	1	0	1		K		Rd		K
---	---	---	---	--	---	--	----	--	---

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

SWAP – zamień półbajty rejestru

Rozkaz zamienia starsze 4 bity z młodszymi 4 bitami w podanym rejestrze Rd.

SWAP Rd(0-31)

Operacja: $Rd(3..0) \leftarrow Rd(7..4)$, $Rd(7..4) \leftarrow Rd(3..0)$

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0		Rd		0	0	1	0
---	---	---	---	---	---	---	--	----	--	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

TST – sprawdź zero lub minus

Rozkaz wykonuje test ustawiając znaczniki Z, N, V. Test polega na wykonaniu mnożenia logicznego. Rozkaz oparty został na instrukcji AND.

TST Rd(0-31)

Operacja: $Rd \leftarrow Rd \cdot Rd$

Liczba cykli: 1

Kod: dla Rd=Rr

0	0	1	0	0	0	Rr	Rd			Rr		
---	---	---	---	---	---	----	----	--	--	----	--	--

I	T	H	S	V	N	Z	C
-	-	-	↔	0	↔	↔	-

WDR – zeruj rejestr zegara watchdog

Instrukcja zeruje zawartość rejestru licznika watchdog.

WDR

Operacja:

Liczba cykli: 1

Kod:

1	0	0	1	0	1	0	1	1	0	1	X	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

Programowanie mikrokontrolerów

Informacje wstępne

Ze względu na liczbę wyprowadzeń, kontrolery ośmiowywyprowadzeniowe programowane są tylko poprzez interfejs SPI. Wewnętrzna pamięć programu FLASH, jak i pamięć EEPROM może być programowana w trybie niskonapięciowym lub wysokonapięciowym. Największą zaletą szeregowej metody programowania, jest możliwość przeprogramowania mikrokontrolera po zamontowaniu w układzie. Program oraz dane po zaprogramowaniu i weryfikacji, możemy zabezpieczyć, poprzez ustawienie odpowiednich bitów na dwóch poziomach zabezpieczenia. Programowanie wysokonapięciowe wymaga napięcia programującego $V_{pp}=12\text{ V}$. Jednak producent, przewidując problemy z takim napięciem w przypadku reprogramowania po zamontowaniu mikrokontrolera w układzie, tak skonstruował mikrokontroler, aby do przeprogramowania wystarczyło $V_{pp}=5\text{ V}$. Pamięć programu jest kasowana automatycznie w trakcie programowania. Skasowana pamięć przyjmuje wartości 0FFh (255d). Producent gwarantuje trwałość pamięci Flash na 1000 cykli kasowania-zapisu. Uwaga: Dokonano syntezy parametrów czasowych w tabelach. W efekcie wyznaczono zakresy częstotliwości i czasy programowania wspólne dla wszystkich kontrolerów.

Zabezpieczenie przed odczytaniem lub skasowaniem pamięci programu

Kontrolery rodziny AVR posiadają dwa poziomy zabezpieczenia programu. Pierwszy poziom nie pozwala na weryfikację. Drugi, oprócz zakazu weryfikacji, nie pozwala przeprogramować mikrokontrolera. Jest to skuteczne zabezpieczenie przed skasowaniem oprogramowania, zapisanego w pamięci programu. Bity zabezpieczające i tryby zabezpieczenia przedstawione zostały w tabeli 12.

Tabela 12. Bity zabezpieczające zawartość pamięci programu

Tryb	LB1	LB2	Typ zabezpieczenia
1	1	1	Program niezabezpieczony
2	0	1	Programowanie zabronione
3	0	0	Tak jak tryb drugi plus zakaz weryfikacji

Jak widać z powyższej tabeli, bity programowane są wartością „0” tzw. logika ujemna. Bity te, są wymazywane podczas operacji kasowania pamięci programu i przyjmują wartość „1”.

Bity konfiguracyjne

Mikrokontrolery posiadają bity konfiguracyjne. Umożliwiają one skonfigurowanie niektórych właściwości kontrolera, wyłącznie w trakcie programowania. Bity zostają zaprogramowane poprzez zapisanie na nich wartości „0” – tzw. logika ujemna. Dostępne są następujące bity konfiguracyjne:

- SPIEN** dotyczy: ATtiny12/15/22, AT90S2323/2343
Jeśli bit=0, to dozwolone jest reprogramowanie kontrolera w trybie niskonapięciowym. Bit można przeprogramować tylko w trybie programowania wysokonapięciowego. Domyślnie bit ma wartość równą 0.
- BODLEVEL** dotyczy: ATtiny12/15.
Dla bitu niezaprogramowanego napięcie wyzwalania układu BOD wynosi 1,8 V dla Attiny12 i 2,7 V dla ATtiny15. Po zaprogramowaniu napięcie wyzwalania wynosi 2,7 V dla Attiny12 i 4,0 V dla ATtiny15.
- BODEN** dotyczy: ATtiny12/15.
Zaprogramowanie tego bitu powoduje załączenie układu Brown Out Detection, a co za tym idzie powoduje również zmianę czasów restartu kontrolera.
- RSTDISBL** dotyczy: ATtiny12/15.
Bit powoduje zablokowanie wyzwalania resetu z zewnętrznego wyprowadzenia. Dzięki temu wyprowadzenie może zostać użyte jako linia portu PB5. Bit reprogramować można tylko w trybie wysokonapięciowym. Domyślnie niezaprogramowany.
- RCEN** dotyczy: AT90S2323/2343, ATtiny22.
Bit powoduje załączenie taktowania kontrolera z wewnętrznego generatora RC. Bit ten nie jest zmieniany w cyklu kasowania układu. Domyślnie zaprogramowany.
- CKSEL3, CKSEL2, CLSEL1, CKSEL0**
Dotyczy Attiny10/11/12/15.
Bity sterują wyborem źródła sygnału taktującego kontroler, oraz decydują o czasach restartu kontrolerów.

W tabelach 5 i 6 zostały przedstawione szczegółowe informacje o opcjach wyboru.

dotyczy: AT90S2323/2343, ATtiny10/11

Bit FSTRT po zapisaniu wartości 0 uaktywnia szybki start mikrokontrolera. Zmianę czasów spowodowaną przeprogramowaniem tego bitu można znaleźć w tabeli 6.

FSTRT

Bajty sygnatur

Bajtami sygnatur nazywamy wartości zwracane przez mikrokontroler w trybie odczytu sygnatur. Służą one do pełnej identyfikacji danego typu mikrokontrolera, włącznie z wartością napięcia programującego. Pozwala to, przy dobrze skonstruowanym oprogramowaniu programatora, na uniknięcie pomyłek przy programowaniu. Mikrokontroler przechowuje je w oddzielnej części pamięci. Kontrolery zwracają trzy bajty sygnatur, które przyjmują następujące wartości:

1. \$000:	\$1E	– Produkt firmy Atmel
2. \$001:	\$90	– 1 kB pamięci Flash
	\$91	– 2 kB pamięci Flash
3. \$002:	\$02 dla \$001=91	– AT90S2323
	\$03 dla \$001=91	– AT90S2343
	\$03 dla \$001=90	– ATtiny10
	\$04 dla \$001=90	– ATtiny11
	\$05 dla \$001=90	– ATtiny12
	\$06 dla \$001=90	– ATtiny15
	\$06 dla \$001=91	– ATtiny22

Układ programowania szeregowego wysokonapięciowego

Interfejs fizyczny

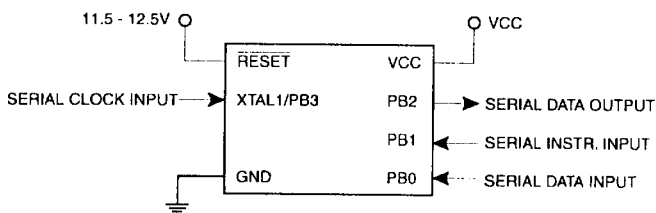
Interfejs programowania w trybie wysokonapięciowym składa się z czterech linii:

- SDI (dane wejściowe – do kontrolera)
- SDO (dane wyjściowe – z kontrolera)

- SI (wejście rozkazu – do kontrolera)
- CLK (zegar synchronizujący – do kontrolera)

Kontroler w trakcie programowania pracuje jako podrzędne urządzenie SPI (jest taktowany zewnętrznym sygnałem zegarowym). Schemat 11 przedstawia układ podłączenia mikrokontrolera do programowania w trybie szeregowym wysokonapięciowym.

Schemat 11. Układ programowania szeregowego wysokonapięciowego



Sekwencja załączenia zasilania programowanego kontrolera

Aby nie wywoływać dodatkowych cykli kasowania lub programowania, oraz zapewnić poprawne rozpoczęcie programowania, należy wykonać szereg czynności, które to zapewnią. Poniżej przedstawiono sekwencje załączenia i wyłączenia zasilania, która będzie poprawnie współpracowała ze wszystkimi opisywanymi kontrolerami. Sekwencja załączenia zasilania (Power-up) dla trybu wysokonapięciowego wygląda następująco:

- załączyć napięcie zasilające $V_{cc}=4,5-5,5$ V między wyprowadzenie V_{cc} a GND
- ustawić linie portu PB5(RESET) i PB0(SDI)
- odczekać co najmniej 30 μ s
- wygenerować przynajmniej 4 okresy przebiegu na linii PB3(SCK) o czasie trwania nie krótszym, niż 100ns (proponowana $f=1..4$ MHz)
- ustawić PB3 w poziom niski „0”
- odczekać co najmniej 100 ns
- podać napięcie $V_{pp}=12$ V na linię portu PB5(RESET)
- odczekać co najmniej 100 ns przed dokonaniem zmiany stanu linii PB0
- odczekać co najmniej 8 μ s przed wysłaniem jakiegokolwiek rozkazu

Dodatkowe uwagi do programowania

W trakcie programowania należy pamiętać o tym, iż:

- pamięć Flash jest programowana: jedna komórka w danym cyklu poprzez wysłanie adresu, młodszego bajtu danych, starszego bajtu danych
- zapis rozkazu odbywa się w tym samym czasie co zapis danych
- przesłanie danych i rozkazów do kontrolera jest dozwolone, pod warunkiem, iż linia PB2(RDY/BSY) znajduje się w stanie wysokim
- pamięć EEPROM jest programowana: jedna komórka w danym cyklu poprzez wysłanie adresu oraz bajtu danych
- dowolna komórka pamięci może zostać zweryfikowana poprzez dokonanie jej odczytu. Odczytana wartość jest zwracana poprzez linię portu PB2(SDO)

Sekwencja wyłączenia zasilania po zaprogramowaniu kontrolera

Sekwencja wyłączenia zasilania polega na:

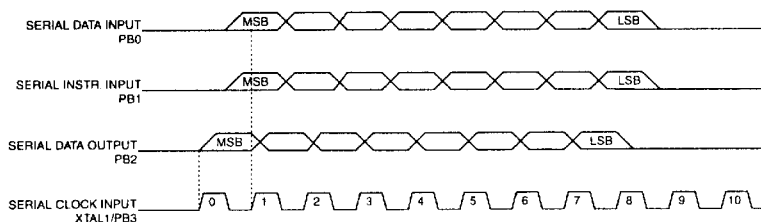
- ustawieniu linii portu PB3 w stan niski
- ustawieniu linii portu PB5
 - w stan niski dla: ATtiny15
 - w stan wysoki dla: ATtiny10/11/12/22, AT90S2323/2343
- wyłączeniu zasilania

Przebiegi sygnału w trakcie programowania wysokonapięciowego

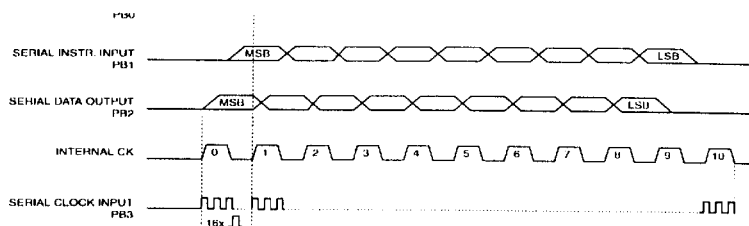
Na wykresie 3 przedstawione zostały przebiegi sygnału transmisji danych w trakcie programowania wysokonapięciowego, z zaznaczeniem miejsc wystawienia i akwizycji danych przez kontroler (linie przerywane).

Wykres 3. Przebiegi transmisji dla trybu programowania wysokonapięciowego

Dotyczy ATtiny15



Dotyczy pozostałych kontrolerów

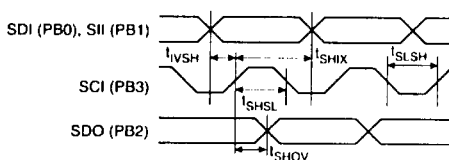


Charakterystyka przebiegów czasowych w trakcie programowania szeregowego wysokonapięciowego

Na wykresie 4 przedstawione zostały przebiegi czasowe transmisji danych w trakcie programowania wysokonapięciowego, wraz z zaznaczeniem najważniejszych czasów przebiegów elektrycznych. W tabeli 13 mamy pełny opis parametrów czasowych do wykresu 4.

Wykres 4. Przebiegi czasowe transmisji dla trybu programowania wysokonapięciowego

Dotyczy ATtiny15



Dotyczy pozostałych kontrolerów

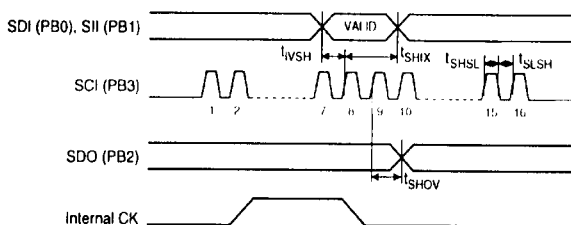


Tabela 13. Parametry czasowe sygnału w trakcie programowania wysokonapięciowego

Symbol	Parametr	Min	Typ	Max	J.m.
Dotyczy ATtiny15					
t_{SHSL}	SCI (PB3) Szerokość impulsu w stanie wysokim	25	-	-	ns
t_{SLSH}	SCI (PB3) Szerokość impulsu w stanie niskim	25	-	-	ns
t_{VSH}	SDI (PB0), SII (PB1) Minimalny czas potrzebny na ustalenie stanów wejść SCI i SII	50	-	-	ns
t_{SHIX}	SDI (PB0), SII (PB1) Minimalny czas przytrzymania stanów wejść SCI i SII potrzebny na poprawną akwizycję danych	50	-	-	ns
t_{SHOV}	SCI (PB3), Czas zmiany danych wyjściowych względem zbocza narastającego linii SCI.	10	16	32	ns
Dotyczy pozostałych kontrolerów					
t_{SHSL}	SCI (PB3) Szerokość impulsu w stanie wysokim	100	-	-	ns
t_{SLSH}	SCI (PB3) Szerokość impulsu w stanie niskim	100	-	-	ns
t_{VSH}	SDI (PB0), SII (PB1) Minimalny czas potrzebny na ustalenie stanów wejść SCI i SII	50	-	-	ns
t_{SHIX}	SDI (PB0), SII (PB1) Minimalny czas przytrzymania stanów wejść SCI i SII potrzebny na poprawną akwizycję danych	50	-	-	ns
t_{SHOV}	SCI (PB3), Czas zmiany danych wyjściowych względem zbocza narastającego linii SCI.	10	16	32	ns
t_{WLWH_CE}	Czas oczekiwania po przesłaniu rozkazu kasowania pamięci	5	10	15	ms
t_{WLWH_PFB}	Czas oczekiwania po przesłaniu rozkazu zapisu bitów konfiguracyjnych	1,0	1,5	1,8	ms

Rozkazy trybu programowania wysokonapięciowego

W tabeli 14 przedstawiony został komplet rozkazów dla trybu programowania wysokonapięciowego.

Tabela 14. Rozkazy dla trybu programowania wysokonapięciowego

Rozkaz	Doryczy	Format rozkazu				Uwagi
		Bajt 1	Bajt 2	Bajt 3	Bajt 4	
Kasuj pamięć programu	ATiny15/22	PB0 0,1000,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,1100_00 x,xxxx,xxxx_xx	Czekaj po przesłaniu bajtu nr 3 (4 dla ATiny12) dopóki linia PB2 nie przejdzie w stan wysoki (odczekać czas t _{WPW} , ce dla AT90S232/2343, ATiny10/11/22)
	ATiny12/11/12	PB0 0,1000,1100_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,1100_00 x,xxxx,xxxx_xx	
	AT90S232/2343	PB0 0,0001,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0001,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0000,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0000,1100_00 x,xxxx,xxxx_xx	
	ATiny12/11/22	PB0 0,0001,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0001,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0000,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0000,1100_00 x,xxxx,xxxx_xx	
	AT90S232/2343	PB0 0,0001,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0001,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0000,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0000,1100_00 x,xxxx,xxxx_xx	
	ATiny15	PB0 0,0001,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0001,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0000,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0000,1100_00 x,xxxx,xxxx_xx	
Zapisz adres pamięci programu w kodzie starszy, modyfikuj bajt	ATiny12/11/22	PB0 0,0000,0010_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	Czekaj po bajcie nr 3 dopóki linia PB2 nie przejdzie w stan wysoki. Powtarzaj całość dla każdego nowego adresu.
	AT90S232/2343	PB0 0,0000,0010_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	ATiny22	PB0 0,0000,0010_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,1100_00 x,xxxx,xxxx_xx	
	AT90S232/2343	PB0 0,0000,0010_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,1100_00 x,xxxx,xxxx_xx	
	ATiny12/11/22	PB0 0,0000,0010_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,1100_00 x,xxxx,xxxx_xx	
	AT90S232/2343	PB0 0,0000,0010_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0111,1100_00 x,xxxx,xxxx_xx	
Czytaj starszy i modyfikuj bajt adresu pamięci programu	ATiny12/11/22	PB0 0,0000,0010_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	AT90S232/2343	PB0 0,0000,0010_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	ATiny12/11/22	PB0 0,0000,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	AT90S232/2343	PB0 0,0000,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	ATiny15	PB0 0,0000,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	ATiny12/15/22	PB0 0,0001,0001_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
Zapisz adres pamięci EEPROM	ATiny12/15/22	PB0 0,0001,0001_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	Powtarzaj bajt nr 3 dla każdego nowego adresu
	AT90S232/2343	PB0 0,0001,0001_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	ATiny12/15/22	PB0 0,0000,0011_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	AT90S232/2343	PB0 0,0000,0011_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	ATiny12/15/22	PB0 0,0000,0011_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	AT90S232/2343	PB0 0,0000,0011_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
Czytaj adres pamięci EEPROM	ATiny2/15/22	PB0 0,0000,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	Czekaj po bajcie nr 3 dopóki linia PB2 nie przejdzie w stan wysoki
	AT90S232/2343	PB0 0,0000,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	ATiny12/15/22	PB0 0,0000,0011_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	AT90S232/2343	PB0 0,0000,0011_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	ATiny2/15/22	PB0 0,0000,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	
	AT90S232/2343	PB0 0,0000,0000_00 PB1 0,0100,1100_00 PB2 x,xxxx,xxxx_xx	0,0000,0000_00 0,0100,0100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	0,0000,0000_00 0,0110,1100_00 x,xxxx,xxxx_xx	

LEGENDA:

a	starsza część adresu	b	młodsza część adresu
i	dane wejściowe	o	dane wyjściowe
1	1 bit zabezpieczenia	2	2 bit zabezpieczenia
R	RCEN	S	SPIEN
F	ESTRI	.	OKCCE
4	CKSEL1	x	bez znaczenia

Rozkaz	Dotyczy	Format rozkazu				Uwagi
		Bajt 1	Bajt 2	Bajt 3	Bajt 4	
Zapisz bity konfiguracji	AT90S2323	PB0 C_0100_0000_00	0_1151_111F_00	0_0000_0000_00	0_0000_0000_00	Odczytaj, czas trwania, po przesłaniu bajtu nr 3
	PB1 C_0100_1100_00	0_0010_1100_00	C_0110_0100_00	0_0110_1100_00		
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny22	PB0 C_0100_0000_00	0_1151_111F_00	0_0000_0000_00	0_0000_0000_00	
	AT90S2343	PB1 C_0100_1100_00	0_0010_1100_00	C_0110_0100_00	0_0110_1100_00	
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny10/11	PB0 C_0100_0000_00	0_0007_6943_00	0_0000_0000_00	0_0000_0000_00	
	PB1 C_0100_1100_00	0_0010_1100_00	C_0110_0100_00	0_0110_1100_00		
Zapisz bity konfiguracji	ATtiny12	PB0 C_0100_0000_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	Po przesłaniu bajtu nr 3 czeka dopóki linia PB2 nie przejdzie w stan wysoki
	PB1 C_0100_1100_00	0_0010_1100_00	C_0110_0100_00	0_0110_1100_00		
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny15	PB0 C_0100_0000_00	0_8765_1143_00	0_0000_0000_00	0_0000_0000_00	
	PB1 C_0100_1100_00	0_0010_1100_00	C_0110_0100_00	0_0110_1100_00		
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny22	PB0 C_0010_0000_00	0_1111_1111_00	0_0000_0000_00	0_0000_0000_00	
	PB1 C_0100_1100_00	0_0010_1100_00	C_0110_0100_00	0_0110_1100_00		
Czytaj bity konfiguracji bity zabezpieczenia pamięci	AT90S2323/2343	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	Czytaj bity konfiguracji bity zabezpieczenia pamięci
	ATtiny10/11/2/15	PB0 C_0010_0000_00	0_0000_031C_00	0_0000_0000_00	0_0000_0000_00	
	PB1 C_0100_1100_00	0_0010_1100_00	C_0110_0100_00	0_0110_1100_00		
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	AT90S2323	PB0 C_0000_0100_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	
	PB1 C_0100_1100_00	0_0110_1100_00	C_0110_0100_00	0_0110_1100_00		
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny22	PB0 C_0000_0100_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	
Czytaj bity konfiguracji	AT90S2343	PB1 C_0100_1100_00	0_0000_0000_00	0_0110_1100_00	0_0110_1100_00	Czytaj bity konfiguracji bity zabezpieczenia pamięci
	ATtiny10/11	PB0 C_0000_0100_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	
	PB1 C_0100_1100_00	0_0110_1100_00	C_0110_0100_00	0_0110_1100_00		
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny12	PB0 C_0000_0100_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	
	PB1 C_0100_1100_00	0_0110_1100_00	C_0110_0100_00	0_0110_1100_00		
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny15	PB0 C_0000_0100_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	
Czytaj bity konfiguracji	ATtiny10/11/2/15	PB0 C_0100_1100_00	0_0110_1100_00	C_0110_0100_00	0_0111_1100_00	Czytaj bity konfiguracji bity zabezpieczenia pamięci
	PB1 C_0100_1100_00	0_0110_1100_00	C_0110_1100_00	0_0110_1100_00		
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny12/15/22	PB0 C_0000_1000_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	
	AT90S2323/2343	PB1 C_0100_1100_00	0_0000_1100_00	C_0110_0100_00	0_0110_1100_00	
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny12/15	PB0 C_0000_1000_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	
	PB1 C_0100_1100_00	0_0000_1100_00	C_0110_0100_00	0_0111_1100_00		
Czytaj bity konfiguracji	ATtiny12/15	PB0 C_0000_1000_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	Czytaj bity konfiguracji bity zabezpieczenia pamięci
	PB1 C_0100_1100_00	0_0000_1100_00	C_0110_0100_00	0_0111_1100_00		
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny12/15	PB0 C_0000_1000_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	
	PB1 C_0100_1100_00	0_0000_1100_00	C_0110_0100_00	0_0111_1100_00		
	PB2 *_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx	*_xxxx_xxxx_xx		
	ATtiny12/15	PB0 C_0000_1000_00	0_0000_0000_00	0_0000_0000_00	0_0000_0000_00	
	PB1 C_0100_1100_00	0_0000_1100_00	C_0110_0100_00	0_0111_1100_00		

LEGENDA:

ATtiny15

5	RSTDSDL	6	SPIEN
7	BODEN	8	BODLEVEL

ATtiny10/11/12

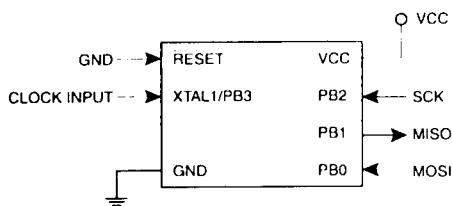
5	CKSEL2	9,6	RSTISBL
7	FSTRT	8	CKSEL3
A	SPIEN	B	BODEN
C	BODLEVEL		

Opis programowania szeregowego niskonapięciowego (Downloading)

Interfejs fizyczny

Interfejs programowania w trybie niskonapięciowym składa się z trzech linii. Stanowi on standardowy interfejs SPI i jest dostosowany do przeprogramowywania kontrolerów w układzie docelowym. Obsługa programowania i weryfikacji w trybie niskonapięciowym wymaga przesłania rekordu, o rozmiarze czterech bajtów. Programowanie po wystąpieniu niskiego poziomu napięcia na wyprowadzeniu RESET, rozpoczynamy od przesłania kombinacji, zezwalającej na programowanie. Ma to na celu zabezpieczyć przed przypadkowym przeprogramowaniem pamięci mikrokontrolera. Na schemacie 12 przedstawiony jest układ dołączenia kontrolera w trybie programowania niskonapięciowego.

Schemat 12. Układ programowania szeregowego niskonapięciowego



Podstawowym warunkiem wprowadzenia kontrolera w tryb programowania niskonapięciowego jest podanie stanu niskiego na wyprowadzenie RESET. Aby transmisja w trakcie programowania przebiegła poprawnie, muszą zostać spełnione następujące warunki:

- czas trwania stanu niskiego > 2 cykle XTAL1;
- czas trwania stanu wysokiego > 2 cykle XTAL1.

Poniżej zostały opisane wszystkie czynności, które należy wykonać, aby przeprowadzić poprawne programowanie.

Sekwencja załączenia zasilania programowanego kontrolera

Aby nie spowodować niewłaściwego zachowania kontrolera, opracowana została specjalna metoda przejścia w tryb programowania niskonapięciowego.

Sama metoda polega na:

- ustawieniu RESET=0 i SCK=0, jeżeli nie ma podłączonego kwarcu, to wymaga się podania sygnału taktującego na wejście XTAL1 oraz załączeniu zasilania Vcc. Jeżeli nie ma gwarancji, iż SCK w momencie załączenia zasilania było równe 0, to po ustawieniu SCK=0 należy wygenerować przynajmniej 2 cykle zegarowe na wyprowadzeniu XTAL1
- odczekaj co najmniej 20 ms, a następnie prześlij rozkaz przejścia w tryb programowania (patrz tabela 16)
- kontroler sygnalizuje wejście w tryb programowania, poprzez zwrócenie w rozkazie załączenia trybu programowania wartości na trzecim bajcie, wynoszącej \$53 (powtarza wartość bajtu nr 2 przesyłanego rozkazu). Jeżeli nie nastąpi opisana sytuacja w trakcie 32 prób przesyłania rozkazu przejścia w tryb programowania, to oznacza brak fizycznego połączenia z kontrolerem

Dodatkowe uwagi do trybu programowania niskonapięciowego

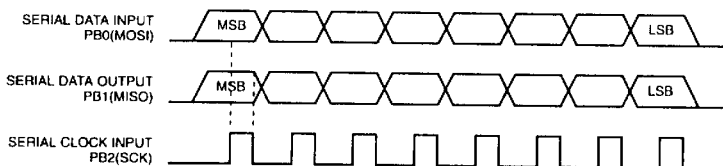
- jeżeli jest wykonywana komenda kasowania, to po przesłaniu całego rozkazu odczekaj czas t_{WD_ERASE} , a następnie należy wygenerować impuls do stanu wysokiego na wejściu RESET. Dalszy ciąg programowania rozpoczynamy od drugiego punktu sekwencji załączenia zasilania
- programowanie odbywa się kolejnymi bajtami, każdy w trakcie przesyłania jednego rozkazu
- „Data Polling”. Kiedy trwa programowanie bajtu pamięci programu lub EEPROM, odczyt danego adresu zwraca wartość \$FF. Po zakończeniu programowania bajtu kontroler zwraca poprawną wartość. Ta właściwość pozwala na testowanie zakończenia programowania. W przypadku zapisu bajtu o wartości \$FF, nie jesteśmy w stanie określić, czy programowanie bajtu zostało zakończone. Dla tego szczególnego przypadku musimy odczekać czas $t_{WD_PROG_FL}$, a w przypadku pamięci EEPROM czas $t_{WD_PROG_EE}$
- dowolna komórka pamięci może zostać zweryfikowana poprzez wykonanie rozkazu odczytu
- po wykonaniu programowania można podać stan wysoki na wejście RESET. Spowoduje to przejście w tryb wykonywania rozkazów

Sekwencja wyłączenia zasilania programowanego kontrolera

- ustaw XTAL1=0 (jeżeli nie podłączono rezonatora kwarcowego)
- ustaw RESET=1
- wyłącz zasilanie Vcc

Wykres 5 przedstawia przebiegi sygnału transmisji w trakcie programowania. Linie przerywane wskazują moment akwizycji i zmiany danych w trakcie programowania.

Wykres 5. Przebiegi transmisji dla trybu programowania niskonapięciowego



Wykres 6. Przebiegi czasowe transmisji dla trybu programowania niskonapięciowego

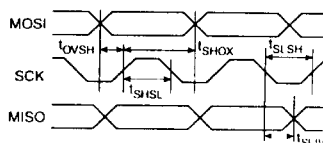


Tabela 15. Charakterystyka czasowa trybu programowania niskonapięciowego

Symbol	Parametr	Min	Typ	Max	J.m.
$1/t_{CLCL}$	Częstotliwość oscylatora	0,8		1,0	MHz
t_{CLCL}	Okres oscylatora	1000		1250	ns
t_{SHSL}	SCK czas trwania stanu wysokiego	$2t_{CLCL}$			ns
t_{SLSH}	SCK czas trwania stanu niskiego	$2t_{CLCL}$			ns
t_{OVSH}	Wyprzedzenie danych na wejściu MOSI w stosunku do SCK	t_{CLCL}			ns
t_{SHOX}	Przytrzymanie danych po zboczu opadającym SCK	$2t_{CLCL}$			ns
t_{SLIV}	Czas przytrzymania danych na wyprowadzeniu MISO względem SCK	10	16	32	ns
t_{WD_ERASE}	Minimalny czas wykonania rozkazu ERASE	18,0			ms
$t_{WD_PROG_FF}$	Czas zapisu komórki pamięci EEPROM	9,0			ms
$t_{WD_PROG_FI}$	Czas zapisu komórki pamięci programu	9,0			ms

Tabela 16. Formaty rozkazów dla trybu programowania niskonapięciowego

Rozkaz	Dotyczy	Format rozkazu				Wagi
		Bajt 1	Bajt 2	Bajt 3	Bajt 4	
Wprowadź tryb programowania	ATiny10/11/12/15/22 AT90S2323/2343	1C10 1100	0101 0011	xxxx xxxx	xxxx xxxx	Rozkaz przesyła się przy RESET=0
Kasowanie pamięci kontrolera	ATiny10/11/12/15/22 AT90S2323/2343	1C10 1100	100x xxxx	xxxx xxxx	xxxx xxxx	Rozkaz kasuje pamięć programu i EEPROM
Czytaj pamięć programu	ATiny10/11/12/15/22 AT90S2323/2343	0C10 H000	xxxx x0aa	bbbb bbbb	0000 0000	H=0 młodszy bajt; H=1 starszy bajt
Zapisz pamięć programu	ATiny10/11/12/15/22 AT90S2323/2343	0100 H000	xxxx x0aa	bbbb bbbb	iiii iii	H=0 młodszy bajt; H=1 starszy bajt
Czytaj pamięć EEPROM	ATiny10/11/12/15/22 AT90S2323/2343	1C10 0000	0000 0000	xbbb bbbb	0000 0000	
Zapisz pamięć EEPROM	ATiny10/11/12/15/22 AT90S2323/2343	1100 0000	0000 0000	xbbb bbbb	iiii iii	
Czytaj bity zabezpieczenia i konfiguracji	ATiny22 AT90S2343	0101 1000	xxxx xxxx	xxxx xxxx	12 Sx xxxx	zaprogramowany, dla bitu=0
Czytaj bity zabezpieczenia i konfiguracji	AT90S2323	0101 1000	xxxx xxxx	xxxx xxxx	12 Sx xxxx	zaprogramowany, dla bitu=0
Zapisz bit RCEN	ATiny22 AT90S2343	1C10 1100	1011 111R	xxxx xxxx	xxxx xxxx	zaprogramowany, dla bitu=0
Zapisz bit FSTR1	AT90S2323	1C10 1100	1011 111F	xxxx xxxx	xxxx xxxx	zaprogramowany, dla bitu=0
Zapisz bity zabezpieczenia	ATiny10/11/12/15/22 AT90S2323/2343	1C10 1100	1111 1211	xxxx xxxx	xxxx xxxx	zaprogramowany, dla bitu=0
Czytaj bity zabezpieczenia	ATiny10/11/12/15/22 AT90S2323/2343	C101 1000	xxxx xxxx	xxxx xxxx	xxxx x21x	zaprogramowany, dla bitu=0
Czytaj bajty sygnatur	ATiny10/11/12/15/22 AT90S2323/2343	0C11 0000	xxxx xxxx	0000 000b	0000 0000	bb - adres kolejnego bajtu sygnatury
Czytaj bajt kalibracji	ATiny10/11/12	0C11 1000	xxxx xxxx	0000 0000	0000 0000	zaprogramowany, dla bitu=0
Zapisz bity konfiguracji	ATiny10/11/12	1C10 1100	101x xxxx	xxxx xxxx	A987 6543	zaprogramowany, dla bitu=0
Czytaj bity konfiguracji	ATiny10/11/12	C101 0000	xxxx xxxx	xxxx xxxx	A987 6543	zaprogramowany, dla bitu=0
Zapisz bity konfiguracji	ATiny15	1C10 1100	101x xxxx	xxxx xxxx	8765 1143	zaprogramowany, dla bitu=0
Czytaj bity konfiguracji	ATiny15	C101 0000	xxxx xxxx	xxxx xxxx	8765 1143	zaprogramowany, dla bitu=0

LEGENDA:

a starszy bajt adresu
H H=0 młodszy bajt

b młodszy bajt adresu
H H=1 starszy bajt

o	dane wyjściowe	i	dane wejściowe
1	1 bit zabezpieczenia	2	2 bit zabezpieczenia
3	bit CKSEL0	4	bit CKSEL1
5	bit CKSEL2	6	bit CKSEL3
7	bit RSTDISBL	8	bit SPIEN
9	bit BODEN	A	bit BODLEVEL
F	bit FSTRT	R	bit RCEN
S	bit SPIEN	x	bit bez znaczenia

W „Linux Plus”, w numerze 12 zostaną umieszczone na płyci pakiety asemblera AVRA oraz języka C-GCC dla AVR-ów.

