Systemy wbudowane

Mikrokontroler ATmega32U4

Podstawowe informacje

- 8-bitowy mikrokontroler
- architektura typu RISC
  - 135 instrukcji procesora
  - 32 rejestry ogólnego przeznaczenia
  - w pełni statyczne operacje
  - 16 MHz taktowanie
  - 16 MIPS
  - wbudowany układ mnożenia (2 taktowy)

Pamięć nieulotna

- 32KB systemowej pamięci Flash (10.000 pełnych cykli zapisów)
- 2,5 KB pamięci SRAM
- 1KB pamięci EEPROM (100.000 pełnych cykli zapisów)
- programowalne zamknięcie oprogramowania

JTAG (IEEE 1149.1)

- wewnątrz układowe wspomaganie odpluskwiania

- możliwość programowania pamięci nieulotnej oraz bitów blokady bezpośrednio przez interfejs JTAG

Kontroler USB 2.0

- w pełni kompatybilny ze standardem
- obsługiwane szybkości 12/1.5 Mpbs
- sześć programowalnych punktów końcowych
- niezależne 832 bajty do alokacji pamięci dla punktu końcowego
- przerwania zawieszenia/wznowienia
- układ taktujący 48 MHz PLL (phase locked loop)

Peryferia

- wbudowany układ PLL dla USB oraz czasomierza (32-96 MHz)
- 8-bitowy czasomierz/licznik z niezależnym trybem przedskalera (prescaler) i porównania (compare)
- dwa 16-bitowe czasomierze/liczniki z niezależnym trybem przedskalera, porównania oraz przechwycenia
- 10-bitowy czarmierz/licznik z trybem PLL (64 MHz) oraz porównania
- cztery 8-bitowe kanały PWM
- cztery kanały PWM z możliwością programowania z rozdzielczością 2-16 bit
- sześć kanałów PWM dla operacji wysokiej szybkości (Hight-Speed) z rozdzielczością 2-11 bit

Peryferia

- wyjściowy modulator porównania
- 12 wejściowy układ 10 bitowy ADC (z kanałami różniocowymi z programowanym zyskiem)
- programowalny szeregowy USART ze sprzętową kontrolą przepływu
- szeregowy interfejs SPI relacji Mistrz/Niewolnik
- szeregowy interfejs 2-wire z układem bajtowym (Byte oriented)
- programowalny czasomierz strażniczy z niezależnym oscylatorem
- komparator analogowy
- wbudowany czujnik temperatury
- reakcje na przerwania oraz impulsy budzące przy zianie stanu wejść (Pin Change)

Wyposażenie dodatkowe

- impuls zerowania (Reset) oraz programowalne wykrywanie obniżonej wartości napięcia (Brown-out)

- wewnętrzny 8MHz skalibrowany oscylator

- wewnętrzny preskaler zegara oraz dynamiczne przełączanie zegara (Int RC / Ext Osc)

- wewnętrzne oraz zewnętrze źródła przerwań

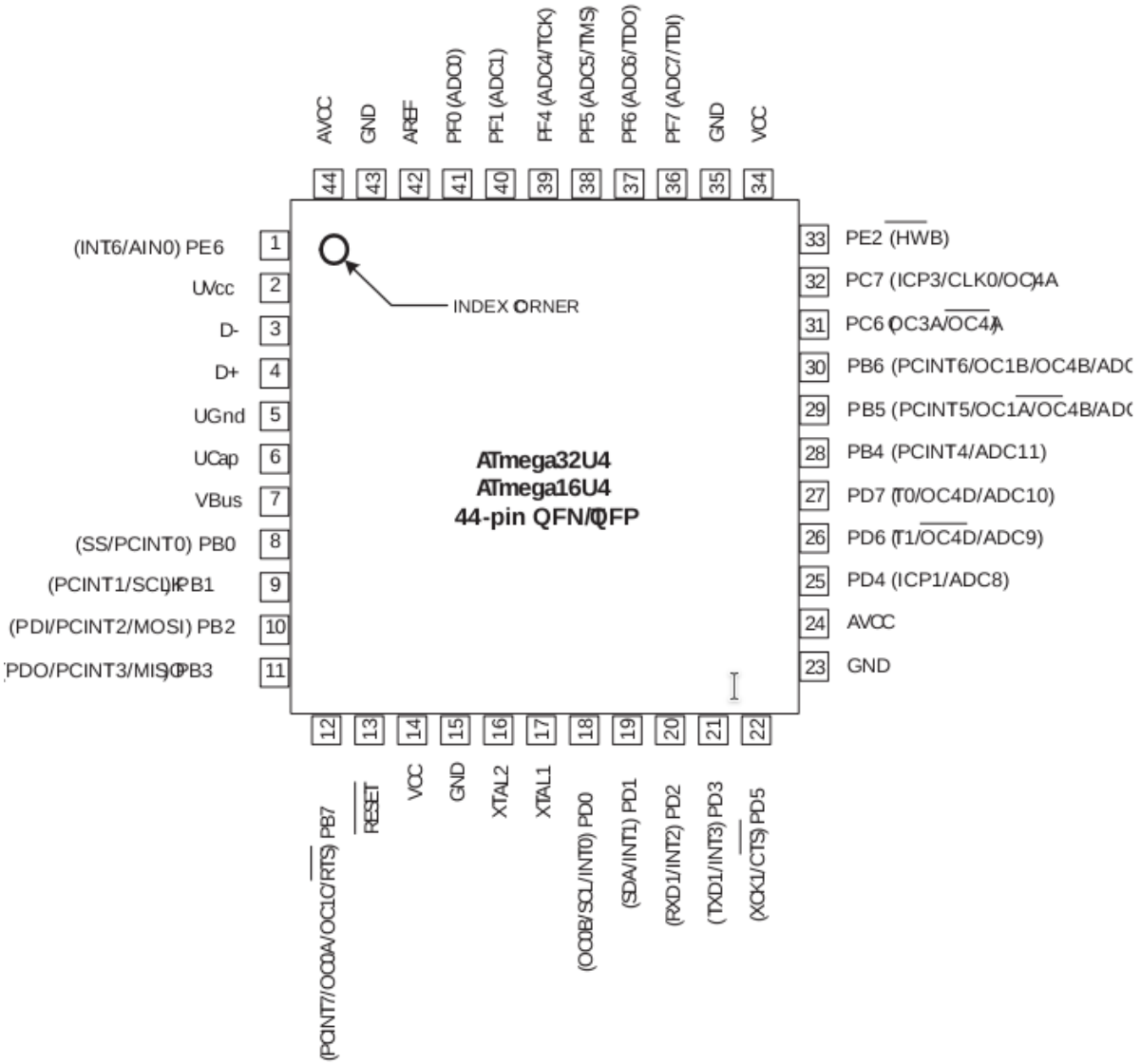
- sześć trybów uśpienia: bezczynność (idle), redukcja szumów ADC, oszczędzanie energii, wyłączenie, oczekiwanie, rozszerzone oczekiwanie

Informacje dodatkowe

- We/Wy łączone do wyjść CMOS oraz wejść LVTTL
- 26 programowalnych linii We/Wy
- 44-wyprowadzeniowa obudowa TQFN 10x10 mm
- napięcie pracy 2,7-5,5 V
- temperatura pracy -40 - +85 stopni Celsjusza
- maksymalna częstotliwość przy 2,7 V – 8 MHz
- maksymalna częstotliwość przy 4,5 V – 16 MHz

Schemat wyprowadzeń z obudowy procesora

Figure 1-1. Pinout



Opis wyprowadzeń

- VCC – zasilanie cyfrowe

- GND – masa

- PB7 … PB0 (Port B) – łącznik 8 bitowy, dwukierunkowy. Wyjście charakteryzuje się symetrycznym buforowaniem oraz wysoką kompatybilnością stanową (stanu przejściowego z wysoki na niski/niski na wysoki). Piny portu są trzystanowe w przypadku sygnału reset – stają się aktywne niezależnie od zegara. Port ten posiada szybszą konwergencję niż pozostałe.

- PC7,PC6 (Port c) – 8 bitowe wyprowadzenie we/wy połączone do rezystorów typu pull-up. Tylko dwa bity tego protu posiadają wyprowadzenie.

- PD7 … PD0 (Port D) – 8-bitowe wyprowadzenie wejścia/wyjścia. Działanie podobne do portu B.

- PE6,PE2 (Port E) – podobny do portu D. Różnica polega na wyprowadzeniach – z 8 bitów dostępne mamy tylko 2.

Opis wyprowadzeń

- PF7 … PF4, PF1, PF0 (Port F) – port wejściowy dla konwertera A/D. Działanie podobne do pozostałych portów. Wyprowadzenia 2 oraz 3 nie są dostępne na obudowie. Dodatkowo pełni on rolę interfejsu JTAG. W tym przypadku piny PF7,PF5 oraz PF4 pozostają aktywne nawet w przypadku nadejścia sygnału reset.

- D-/D+ - linie danych USB

- UGND – masa USB

- UVCC – zasilanie USB

- UCAP – złącze wewnętrznego regulatora napięcia wyjściowego USB

Opis wyprowadzeń

- VBUS – monitor wejścia USB VBUS

- RESET – wejście ponownego rozruchu. Niski stan na tym wejściu przez czas określony jako minimum powoduje restart mikrokontrolera.

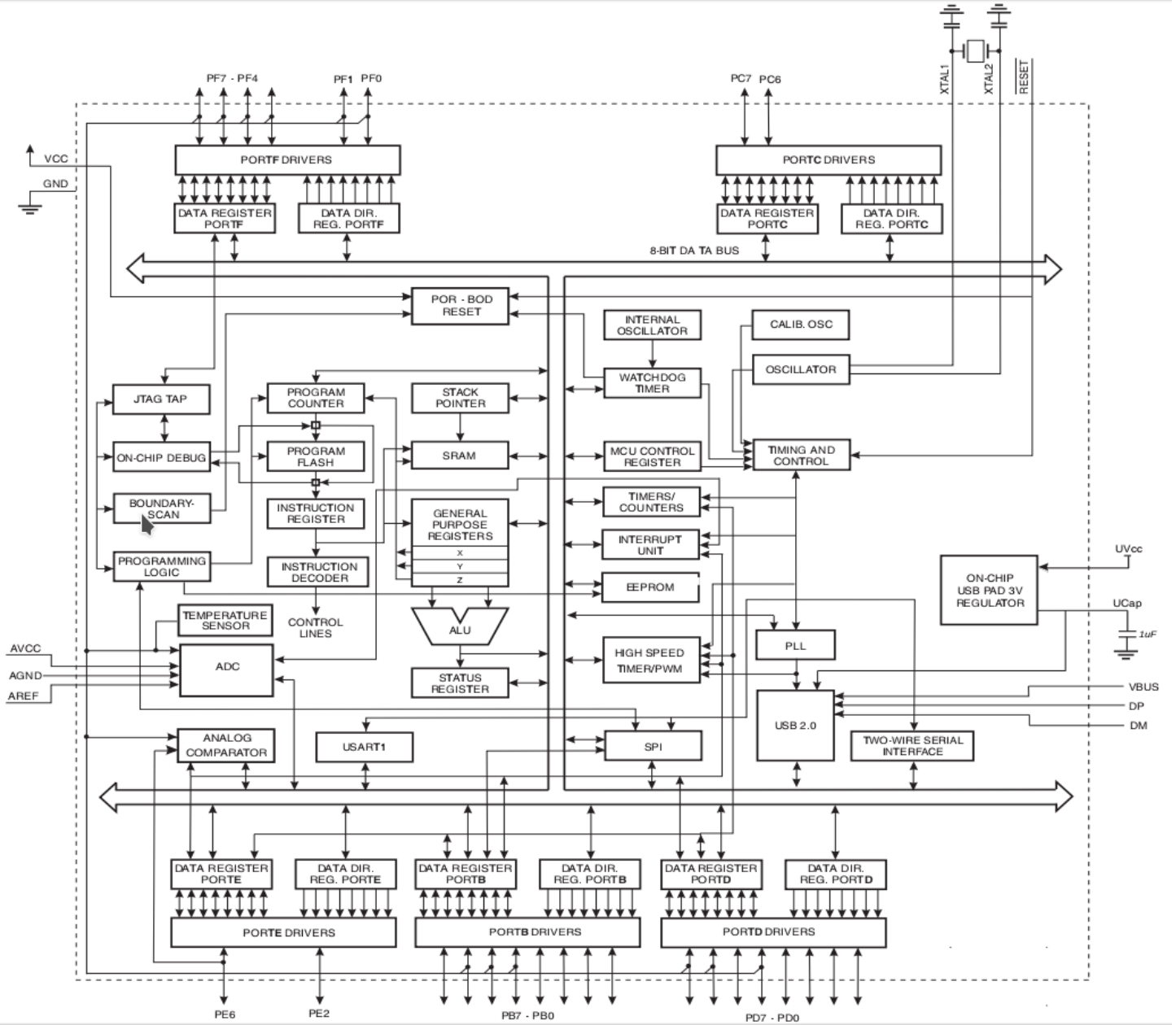
- XTAL1 – wejścia dla oscylatora odwróconej amplitudy oraz do obwodu wewnętrznego zegara mikrokontrolera

- XTAL2 – wyjście dla oscylatora odwróconej amplitudy

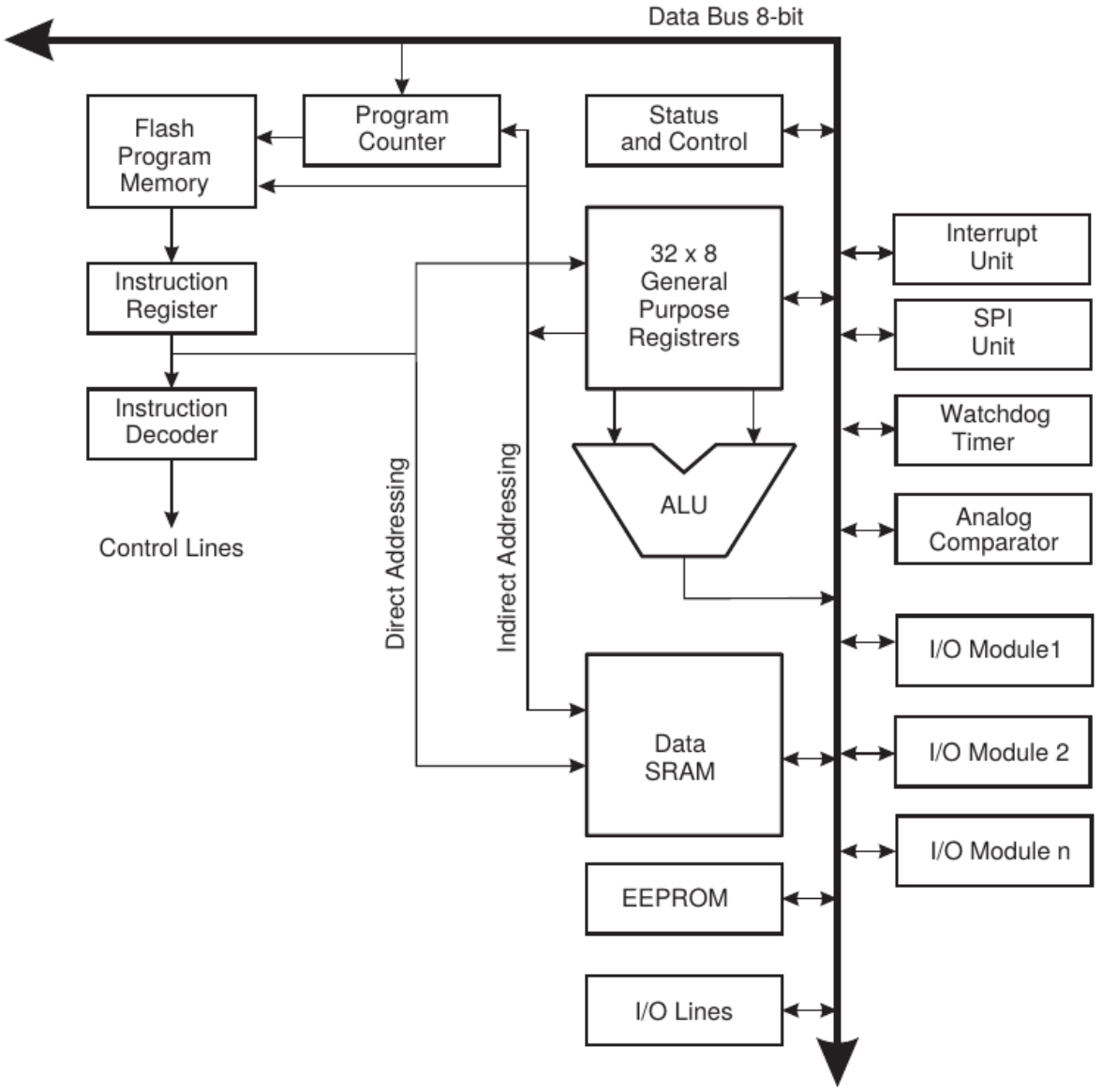
- AVCC – wejście zasilania dla kanałów konwertera A/D. W przypadku braku konwertera powinny być zewnętrznie podłączone do Vcc.

- AREF – wejście odnośnikowe dla konwertera A/D.

Schemat blokowy mikrokontrolera



Schemat architektury AVR



Architektura procesora

- procesor wykonany według architektury Harvard

- instrukcje programu wykonywane są w jednopoziomowym potoku

- w czasie wykonywania jednej instrukcji następna jest pobierana z pamięci

- dzięki temu zminimalizowano do minimum tzw. puste operacje (kolejne operacje mogą być wykonywane co cykl zegarowy procesora)

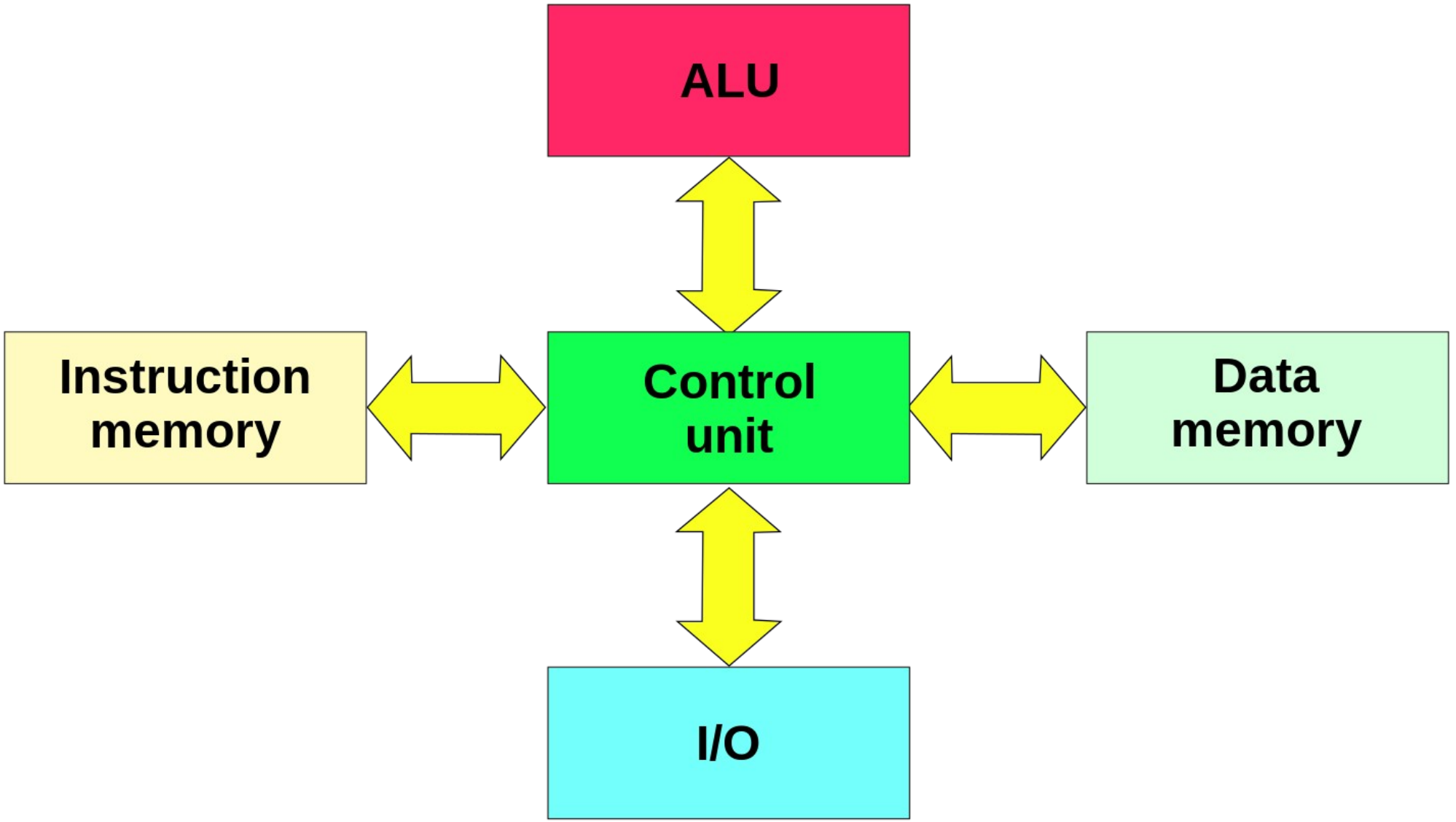
Architektura von Neumann a Harvard

- czym charakteryzuje się architektura Harvard?

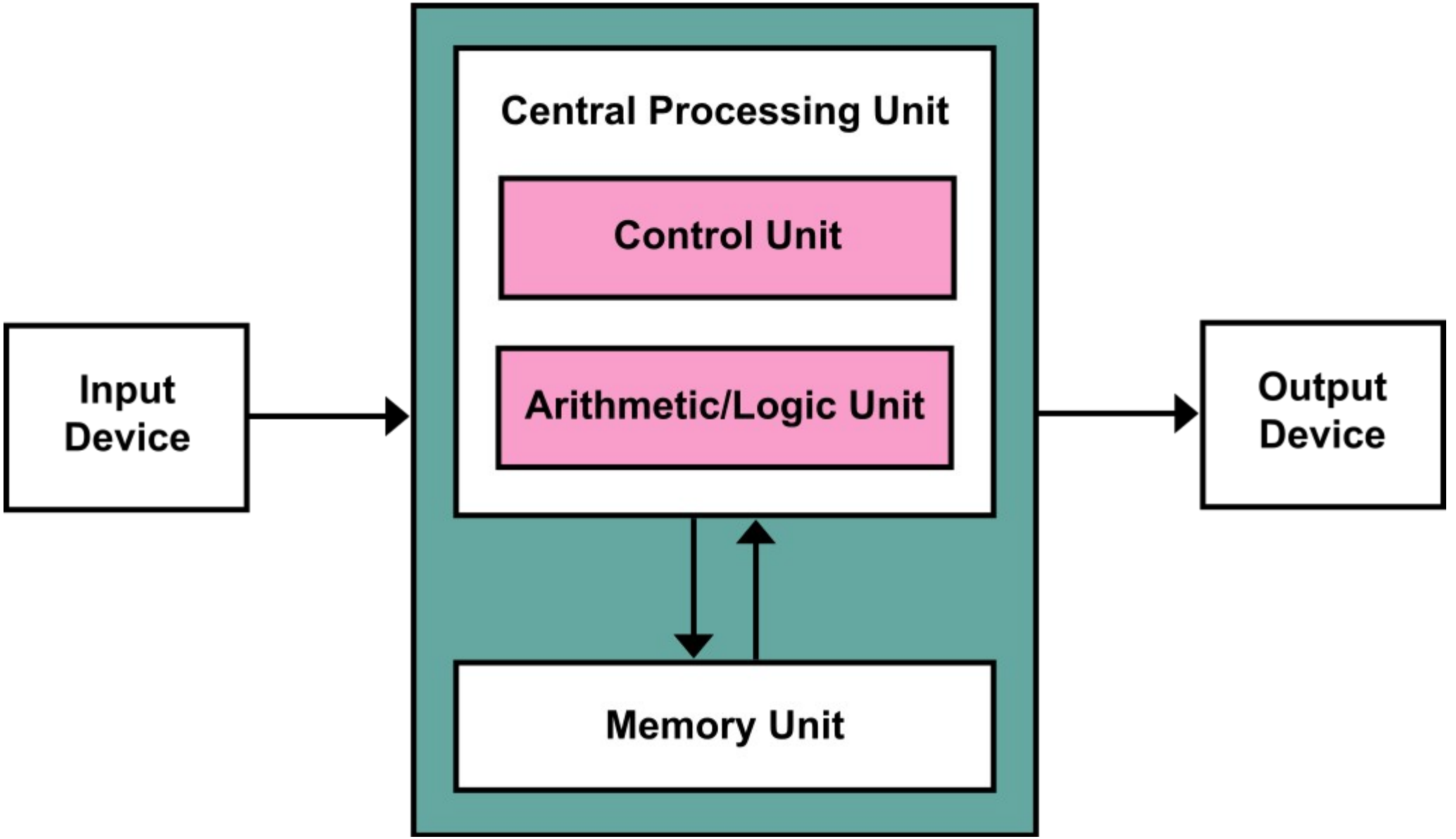
- czym charakteryzuje się architektura von Neumann?

- która z architektur jest lepsza ewentualnie jakie są najlepsze zastosowania każdej z nich

Schemat architektury Harvard



Schemat architektury von Neumann



Rejestry ogólnego przeznaczenia

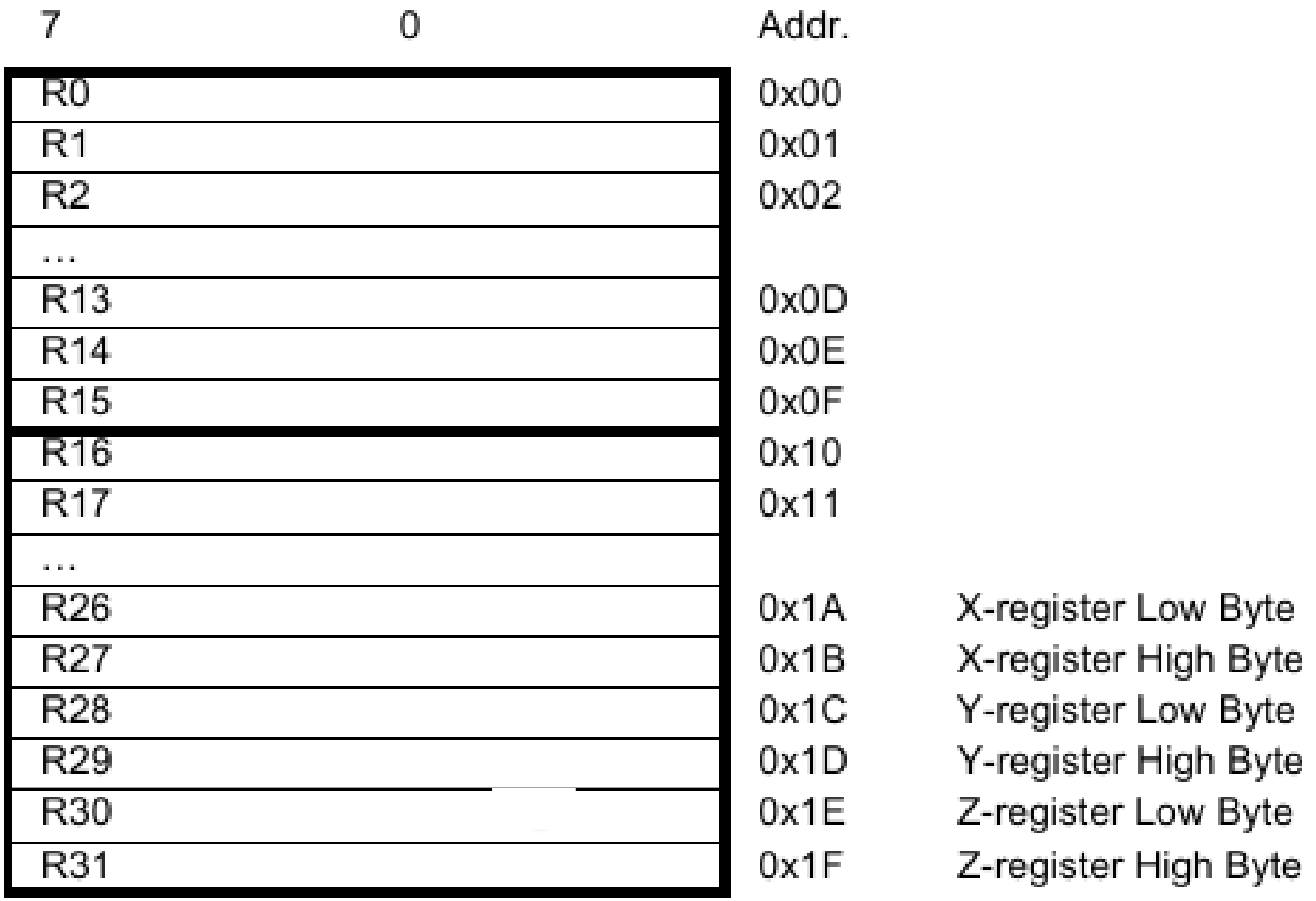
- 32 rejestry ogólnego przeznaczenia o dostępnie pojedynczego czasu dostępu.

- każdy rejestr jest 8-bitowy

- sześć rejestrów może być używanych jako trzy 16-bitowe, w tym jeden z tych rejestrów może być użyty jako wskaźnik adresu w pamięci Flash

- typowe operacje arytmetyczno-logiczne dokonywane przez procesor odbywają się na dwóch rejestrach, rejestrze ze stałą lub na pojedynczym rejestrze.

Schemat poglądowy rejestrów ogólnego przeznaczenia



Schematy operacji wejścia/wyjścia na rejestrach

- pojedynczy wyjściowy operand 8 bit i pojedynczy wejściowy wynik 8 bit

- dwa wyjściowe operandy 8 bit i pojedynczy wejściowy wynik 8 bit

- dwa wyjściowe operandy 8 bit i pojedynczy wejściowy wynik 16 bit

- pojedynczy wyjściowy operand 16 bit i pojedynczy wejściowy wynik 16 bit

Zasada działania rejestrów

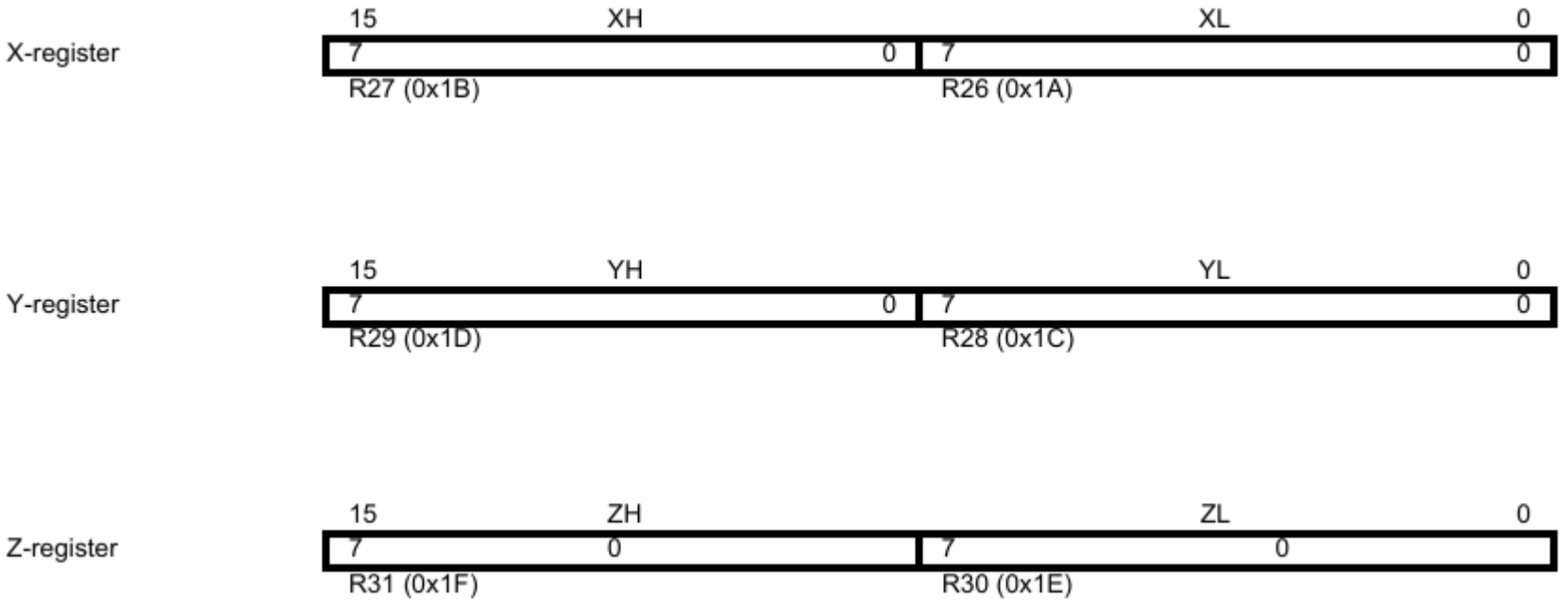
- większość instrukcji operacyjnych posiada bezpośredni dostęp do wszystkich rejestrów

- większość z tych instrukcji zajmuje kontrolerowi jeden cykl

- każdy z rejestrów jest bezpośrednio mapowany do pierwszych 32 lokacji przestrzeni danych użytkownika w pamięci

Rejestry X, Y, Z

- rejestry te stanowią 16-bitowe wskaźniki adresów do pośredniego adresowania przestrzeni danych



Ogólna zasada wykonywania programu przez mikrokontroler

- działanie programu zapewniają skoki bezwarunkowe i warunkowe oraz instrukcje wywołania czy przerwania

- większość instrukcji posiada 16-bitowy format, a pod adresem pamięci można zapisać 16 oraz 32 bitowe instrukcje

- każdorazowo po wykonanej operacji aktualizowany jest Rejestr Statusu (Stanu) o wyniku operacji

- pamięć Flash podzielona jest na dwie części – dla rozruchu oraz dla aplikacji (programu)

- obie części można chronić poprzez bit zamknięcia

- instrukcje SPM, które modyfikują (zapisują) sekcję aplikacji muszą rezydować w części rozruchu

Ogólna zasada wykonywania programu przez mikrokontroler

- w przypadku wywołania przerwania i/lub podprogramu, na stos przerzucany jest adres PC (Program Counter)

- stos ulokowany jest w pamięci SRAM

- Wskaźnik Stosu (SP) musi być zainicjowany przez program zanim zostanie użyty

- SP jest ogólnodostępny w przestrzeni wejścia/wyjścia

- przerwania są dostępne tylko w przypadku, gdy odpowiednia flaga (bit) w Rejestrze Status jest ustawiona

- każde przerwanie posiada swój wektor w tabeli Wektorów Przerwań

- przerwania z niższym adresem w tabeli mają wyższy priorytet

Jednostka arytmetyczno-logiczna

- jednostka może wykonywać operacje na wszystkich 32 rejestrach

- wykonywane operacje podzielone są na 3 kategorie – arytmetyczne, logiczne oraz funkcje bitowe

Rejestr stanu (Status Register)

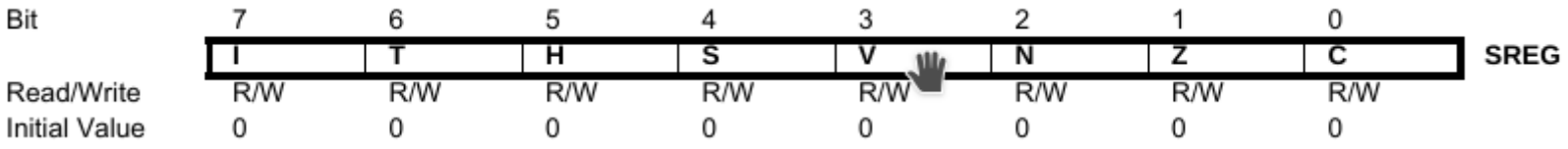
- zawiera informacje o wyniku większości instrukcji arytmetycznych, które mogą zostać wykorzystane w operacjach warunkowych przy wykonywaniu/zmianie instrukcji programu

- rejestr aktualizowany jest po operacjach wykonanych przez ALU

- takie rozwiązanie zdejmuje z programisty obowiązek tworzenia własnych instrukcji porównujących

- WAŻNE – rejestr ten (a w zasadzie jego stan) nie jest przechowywany w przypadku wywołania przerwania; programista sam musi o to zadbać

Rejestr stanu



Opis bitów rejestru stanu

- Bit 7 (flaga globalna przerwań) - flaga decyduje, czy dochodzące do kontrolera przerwania mogą zostać obsłużone. Flaga ta zerowana jest sprzętowo w przypadku wystąpienia przerwania i ponownie ustawiana w chwili nadejścia instrukcji RETI. Flaga może być czyszczona przez programistę poprzez polecenie CLI oraz ustawiana przez polecenie SEI

- Bit 6 (bit przechowania kopii) – bit można ustawić (skopiować) z Rejestru Pliku poprzez instrukcję BST (Bit Store) oraz skopiować do bitu Rejestru Pliku poprzez instrukcję BLD (Bit LoaD).

- Bit 5 (flaga połowy przeniesienia) – ustawiony sygnalizuje wystąpienie Przeniesienia Połowy operacji arytmetycznej (użyteczne w operacjach BCD).

- Bit 4 (bit znaku) – wartość tego bitu zawsze jest wynikiem operacji wyłączeni lub (exclusive or) pomiędzy flagą ujemną oraz flagi przepełnienia ($S = N \oplus V$)

- Bit 3 (flaga przepełnienia porównania dwóch wartości) – flaga przepełnienia z porównania dwóch wartości arytmetycznych

- Bit 2 (flaga ujemna) – informuje o ujemny wyniku operacji arytmetycznej lub logicznej

- Bit 1 (flaga zera) – informuje o zerowym wyniku operacji arytmetycznej lub logicznej

- Bit 0 (flaga przeniesienia) – flaga informuje o wystąpieniu przeniesienia przy operacji arytmetycznej lub logicznej.

Rejestr Stosu (Stack Register)

- stos używany jest do przechowywania danych tymczasowych, np. danych lokalnych oraz adresów powrotu z przerwań bądź wywołań

- zawsze wskazuje na szczyt stosu; dodanie do stosu (PUSH) powoduje zmniejszenie wskaźnika

stosu

- przestrzeń stosu musi zostać zaalokowana PRZED włączeniem przerwań czy odwołań

- wskazuje na dane w pamięci SRAM od jej wyższej lokalizacji do niższej, jednak musi być alokowana powyżej adres 0x0100 (najlepiej na końcu pamięci)

- zmniejszenie wartości wskaźnika o jeden następuje po komendzie PUSH, natomiast przy odłożeniu adresu przez przerwanie – o trzy

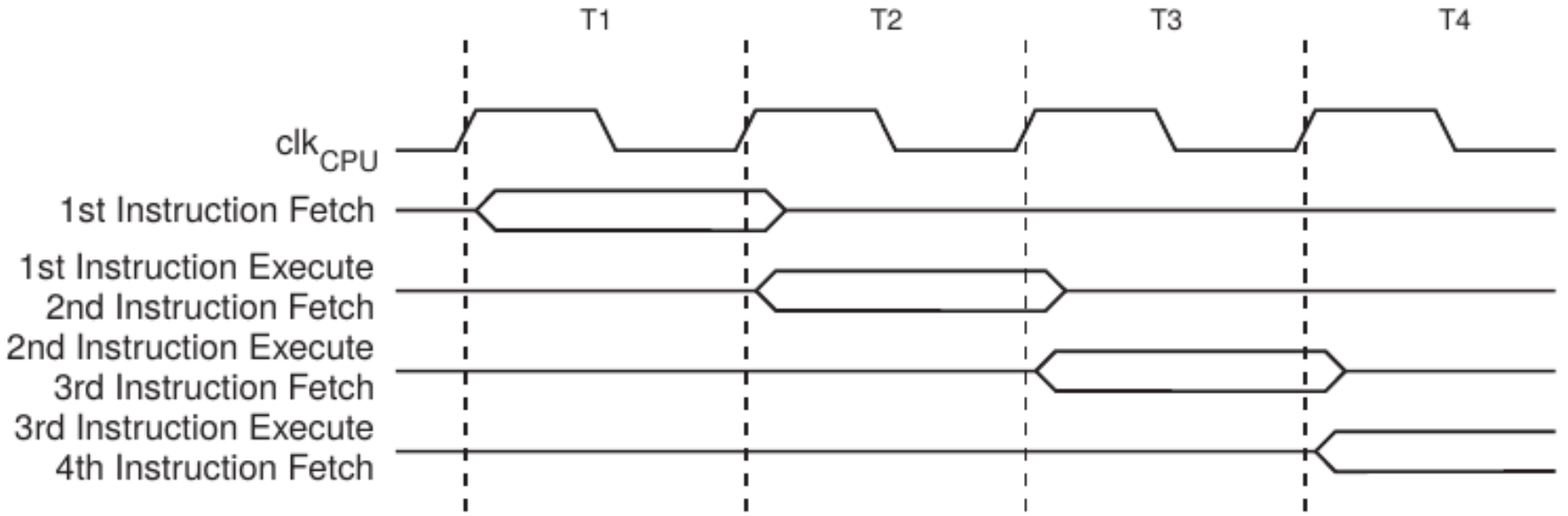
- zwiększenie wartości następuje przy ściągnięciu wartości ze stosu (komenda POP, RET lub RETI)

- stos składa się z dwóch 8 bitowych rejestrów wejścia wyjścia, dzielonych na dwie częście – SPL oraz SPH

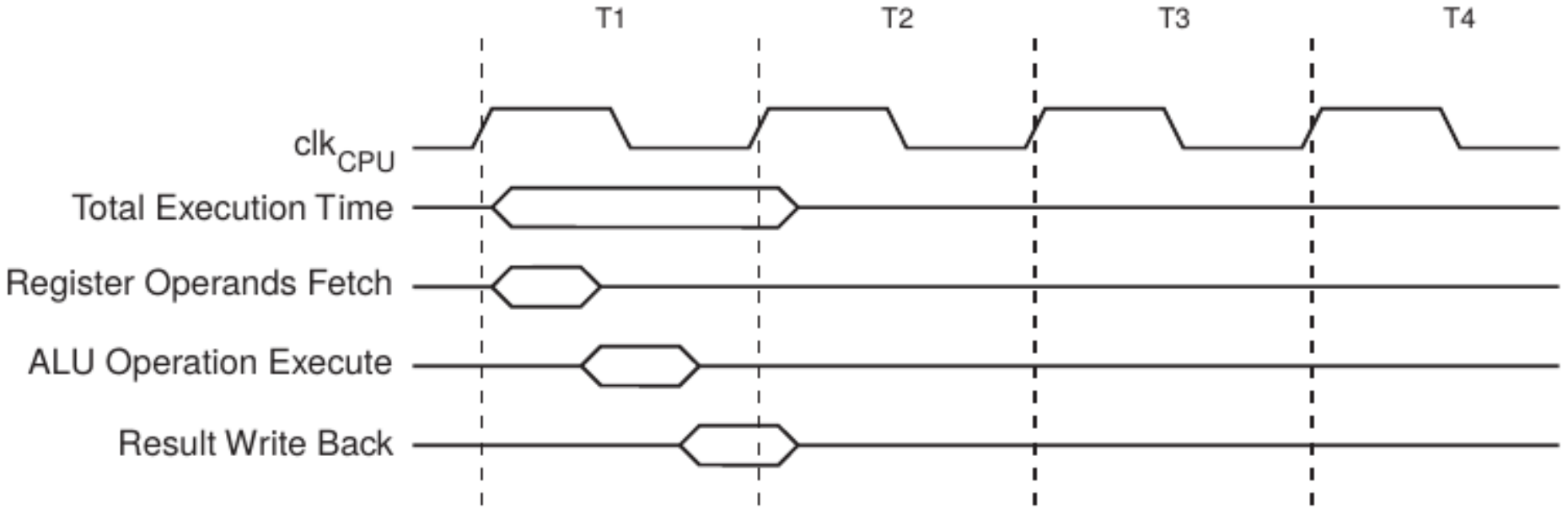
Rejestr stosu

Bit	15	14	13	12	11	10	9	8	
	SP15	**SP14**	**SP13**	**SP12**	**SP11**	**SP10**	**SP9**	**SP8**	**SPH**
	SP7	**SP6**	**SP5**	**SP4**	**SP3**	**SP2**	**SP1**	**SP0**	**SPL**
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	1	0	0	0	0	0	
	1	1	1	1	1	1	1	1	

Czas wykonywania instrukcji



Czas wykonywania instrukcji



Obsługa przerwań

- procesor obsługuje kilka źródeł przerwań

- każde z przerwań posiada swój wektor programu w pamięci; każde przerwanie posiada własne bity ustawienia, które muszą zostać zmienione na stan 1 wraz z bitem przerwań globalnych by włączyć dane przerwanie

- przerwania mogą zostać automatycznie zablokowane gdy bity Zamknięcia startu – BLB02 oraz BLB12 są zaprogramowane

- najniższe adresy zarezerwowane są dla najważniejszych przerwań (najniższy adres posiada w wektorze przerwań RESET)

Obsługa przerwań

- istnieją dwa typy przerwań

- pierwsze z nich to wywołanie przerwania przez kod programu

- dla tego typu przerwań PC jest wektoryzowane do odpowiedniego wektora przerwań

- przerwania programowe, w przypadku obsługi jednego z nich, są zapamiętywane i wykonywane kolejno wedle priorytetu

- drugim typem przerwań jest reakcja na zachodzące warunki (wewnętrzne bądź zewnętrzne)

- w tym wypadku jeżeli stan wywołania przerwania zaniknie to przerwanie nie zostanie wykonane

Czas wywołania przerwań

- odpowiedź kontrolera na przerwanie trwa pięć cykli zegarowych
- podczas tego czasu zawartość PC jest przenoszona na stos
- wektor dokonuje skoku do procedury przerwania (3 cykle)
- jeżeli przerwanie wystąpi podczas wykonywania instrukcji zajmującej więcej niż jeden cykl, czas obsługi się wydłuży
- jeżeli kontroler jest w stanie uśpienia, czas obsługi przerwania zwiększa się o kolejne 5 cykli
- wyjście z obsługi przerwania również zajmuje 5 cykli: 3 bajty PC ściągane są ze stosu, wskaźnik stosu zwiększa się o 3, a bit przerwania (7) w SR zostaje na powrót ustawiony

Materiały

http://www.keil.com/forum/7881/fully-static-operation/

http://www.atmel.com/Images/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf

http://mikrokontrolery.blogspot.com/2011/03/prescaler-postscaler-co-to.html

http://electronics.stackexchange.com/questions/37561/what-is-a-brownout-condition

https://en.wikipedia.org/wiki/Harvard_architecture

https://en.wikipedia.org/wiki/Von_Neumann_architecture

http://www.keil.com/support/docs/3445.htm

http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.faqs/ka3839.html

http://www.atmel.com/webdoc/avrassembler/avrassembler.wb_SPM.html

https://www.atmel.com/Images/doc1644.pdf

http://sandbox.mc.edu/~bennet/cs110/tc/orules.html

http://www.atmel.com/webdoc/avrassembler/avrassembler.wb_ELPM.html