Obsługa zdarzeń - przerwania, zapytywanie wykład 12

Adam Szmigielski
aszmigie@pjwstk.edu.pl

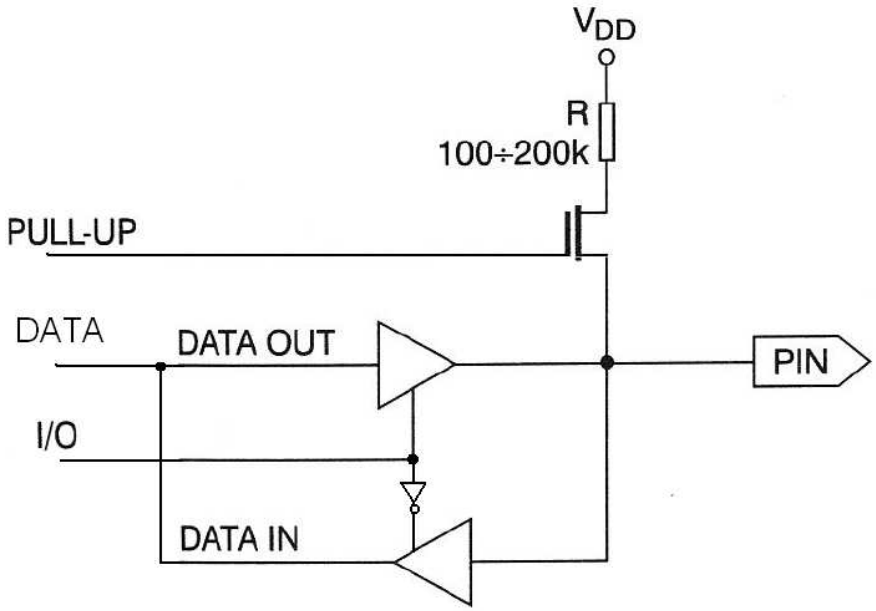
Metody obsługi zdarzeń

- *Przerwanie* (ang. Interrupt) - zmiana sterowania, niezależnie od aktualnie wykonywanego programu, spowodowana pojawieniem się sygnału przewania. Pojawienie się przerwania powoduje wstrzymanie aktualnie wykonywanego programu i wykonanie przez kontroler procedury obsługi przerwania.
- *Zapytywanie* (ang. Polling) - aktywne, okresowe, próbkowanie (sprawdzanie) statusu urządzeń zewnętrznych przez kontroler.

Zapytywanie **(ang. Polling)**

- Technika *polling* jest najczęściej używana w kontekście obsługi urządzeń wejścia/ wyjścia,
- W *polling-u* komputer centralny cyklicznie sprawdza stan urządzenia zewnętrznego w oczekiwaniu na gotowość tego urządzenia - czeka na gotowość,
- *Polling* znajduje zastosowanie w sytuacjach, gdy komputer łączy się z zewnętrznymi urządzeniami w celu zebrania (odświeżenia) danych, przy czym współpraca ta odbywa się w trybie *off-line*,
- *Polling* może być wykorzystany do wymiany informacji z urządzeniami zewnętrznymi, w sytuacji gdy z jakiś względów urządzenia te nie mogą rozpocząć komunikacji,
- W systemach obsługujących jedno zadanie *polling* może również mieć zastosowanie. Większość czasu procesora byłaby wówczas tracona na sprawdzanie gotowości urządzenia,
- W systemach, które wymagają wykonania **wielu zadań** *polling* **jest mało efektywny** w stosunku do przerwań.

Port wejścia-wyjścia - wybór trybu pracy



- Zmiana funkcji z wyjścia na wejście:
  - zablokowanie lub odblokowanie bufora (sygnał I/O),
  - możliwość uaktywnienia obwodu podciągającego (sygnał PULL-UP),
- Możliwe stany wyjścia:
  - stan niski,
  - stan wysoki,
  - stan wysokiej impedancji (domyślny po resecie).

Konfiguracja programowa portu - BASCOM-AVR

- Należy ustawić funkcje portów:
  - **Config** PORT = **Output** - dla wyjścia,
  - **Config** PIN = **Input** - dla wejścia,
- Przy braku zewnętrznych rezystorów podciągających (stan wysokiej impedancji) należy ustawić stan portu na wysoki:
  - **Set** PORT
- Przykład:

```
$crystal = 18432000
$regfile = "m32def.dat"

Config PIND.2 = Input
Config PORTD.5 = Output
Set PORTD.2
Set PIND.2

Do
PORTD.5 = PIND.2
Loop

End
```

Można ustawiać i odwoływać się do pojedynczych pinów portu.

Rodzaje przerwań

1. *Sprzętowe:*
   - *Zewnętrzne* sygnał przerwania pochodzi z zewnętrznego źródła. Przerwania te służą do komunikacji z urządzeniami zewnętrznymi.
   - *Wewnętrzne* **-** pochodzące od timera
   - *Wewnętrzne wyjątki* **-** (ang. exceptions) – zgłaszane przez procesor dla sygnalizowania sytuacji wyjątkowych (np. dzielenie przez zero)
2. *Programowe:* z kodu programu wywoływana jest procedura obsługi przerwania (do komunikacji z systemem operacyjnym).

Wektory przerwań

- *Wektor przerwań* jest adresem początku obsługi przerwania,
- *Wektor przerwań*, w momencie wystąpienia przerwania, wpisywany jest do *licznika rozkazów* - rejestr PC, a zawartość rejestru PC jest kładziona na *stos*,
- Adresy procedur obsługi przerwań zapisane są w *tablicy wektorów przerwań*,
- Przechowuje ona adresy poszczególnych procedur obsługi przerwań,

Tablica wektorów przerwań dla μC Atmel ATMega32

Vector No.	Program Address	Source	Interrupt Definition
1	$000	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	$002	INT0	External Interrupt Request 0
3	$004	INT1	External Interrupt Request 1
4	$006	INT2	External Interrupt Request 2
5	$008	TIMER2 COMP	Timer/Counter2 Compare Match
6	$00A	TIMER2 OVF	Timer/Counter2 Overflow
7	$00C	TIMER1 CAPT	Timer/Counter1 Capture Event
8	$00E	TIMER1 COMPA	Timer/Counter1 Compare Match A
9	$010	TIMER1 COMPB	Timer/Counter1 Compare Match B
10	$012	TIMER1 OVF	Timer/Counter1 Overflow
11	$014	TIMER0 COMP	Timer/Counter0 Compare Match
12	$016	TIMER0 OVF	Timer/Counter0 Overflow
13	$018	SPI, STC	Serial Transfer Complete
14	$01A	USART, RXC	USART, Rx Complete
15	$01C	USART, UDRE	USART Data Register Empty
16	$01E	USART, TXC	USART, Tx Complete
17	$020	ADC	ADC Conversion Complete
18	$022	EE_RDY	EEPROM Ready
19	$024	ANA_COMP	Analog Comparator
20	$026	TWI	Two-wire Serial Interface
21	$028	SPM_RDY	Store Program Memory Ready

Dokładniejsze informacje w dokumentacji.

Przerwania programowe

- Z kodu programu wywoływana jest procedura obsługi przerwania,
- Najczęściej wykorzystywane do komunikacji z systemem operacyjnym, który w procedurze obsługi przerwania (np. w DOS 21h)

Przerwania maskowalne i niemaskowalne

- *Przerwania maskowalne* które można blokować i odblokować programowo,
- *Przerwania niemaskowalne* **-** przerwania, których nie można zablokowć programowo. Są to przerwania, których wystąpienie każdorazowo powoduje bezwarunkowy skok do funkcji obsługi tego przerwania, np. reset

Obsługa przerwania

- Procedura obsługi przerwania - ciąg rozkazów realizujących pożądaną reakcję na przerwanie,
- *Program główny* - sekwencja działań (rozkazów) mikroprocesora realizowanych gdy nie ma przerwań,
- Obsługa przerwania nie może wprowadzać żadnych zmian w programie głównym.

Procedura obsługi przerwania

1. Rozpoznanie przyczyny przerwania (realizacja może być sprzętowa),
2. Skasowanie przyczyny przerwania (realizacja może być sprzętowa),
3. Zablokowanie przerwania,
4. Składowanie na stosie rejestrów roboczych,
5. Właściwa obsługa przerwania,
6. Odtworzenie rejestrów roboczych ze stosu,
7. Odblokowanie przerwania,
8. Powrót do zawieszonego programu.

Stos

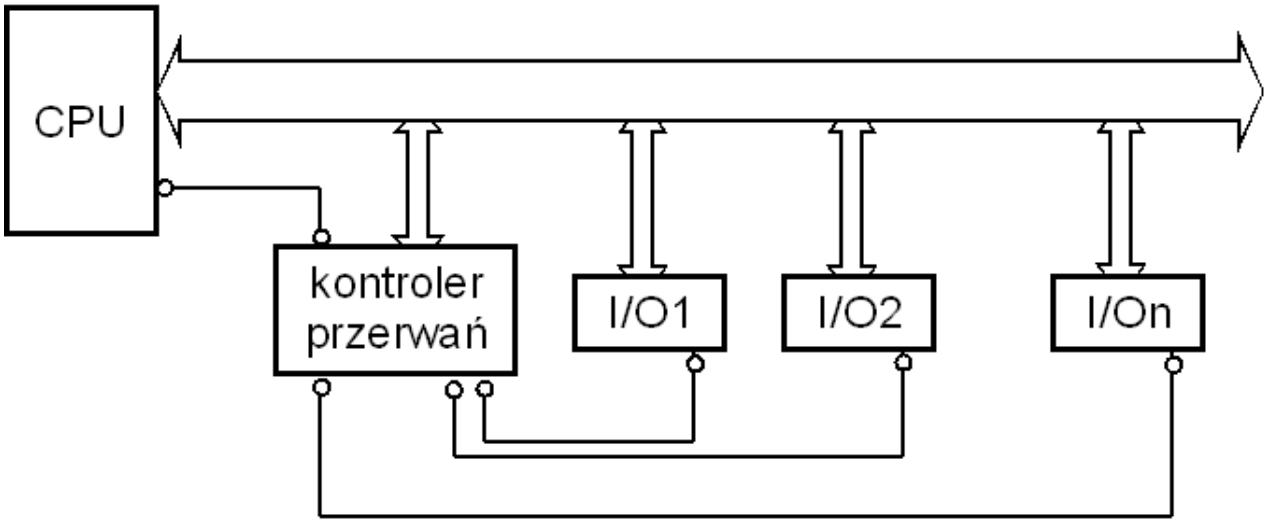
- W momencie wywołania przerwania adres odkładany jest na stos,
- Wskaźnik stosu powinien być ustawiony na miejsce gdzie znajduje się stos.

Priorytet przerwań

- *Priorytet przerwań* **-** zróżnicowanie co do ważności (pilności) zadań realizowanych przez system mikroprocesorowy,
- W szczególności zadaniami tymi mogą być procedury obsługi przerwań różnicując ich pilność dokonuje się określenia priorytetów poszczególnych przerwań,
- W przypadku AVR system obsługi przerwań jest płaski (brak hierarchii). Wszystkie przerwania są jednakowo ważne.

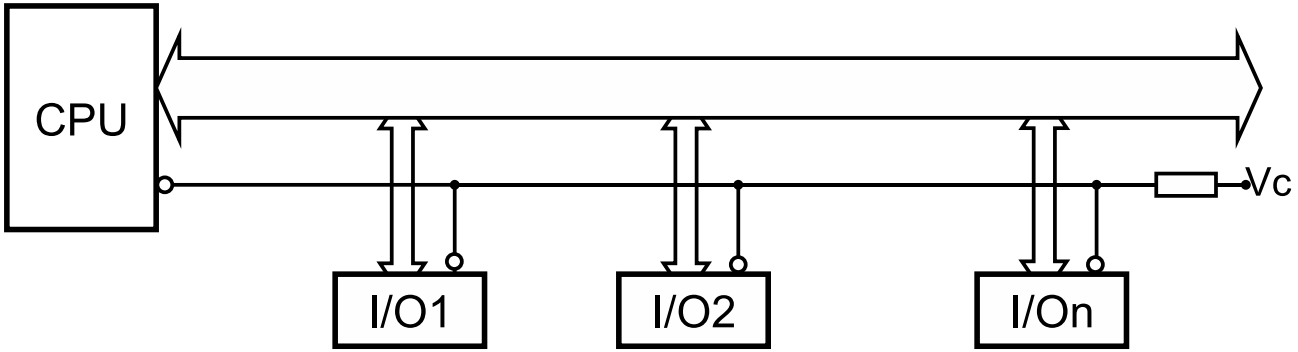
Sprzętowa realizacja hierarchii przerwań

- *sprzętowo* - o wyborze przerwania decyduje *kontroler przerwań*.



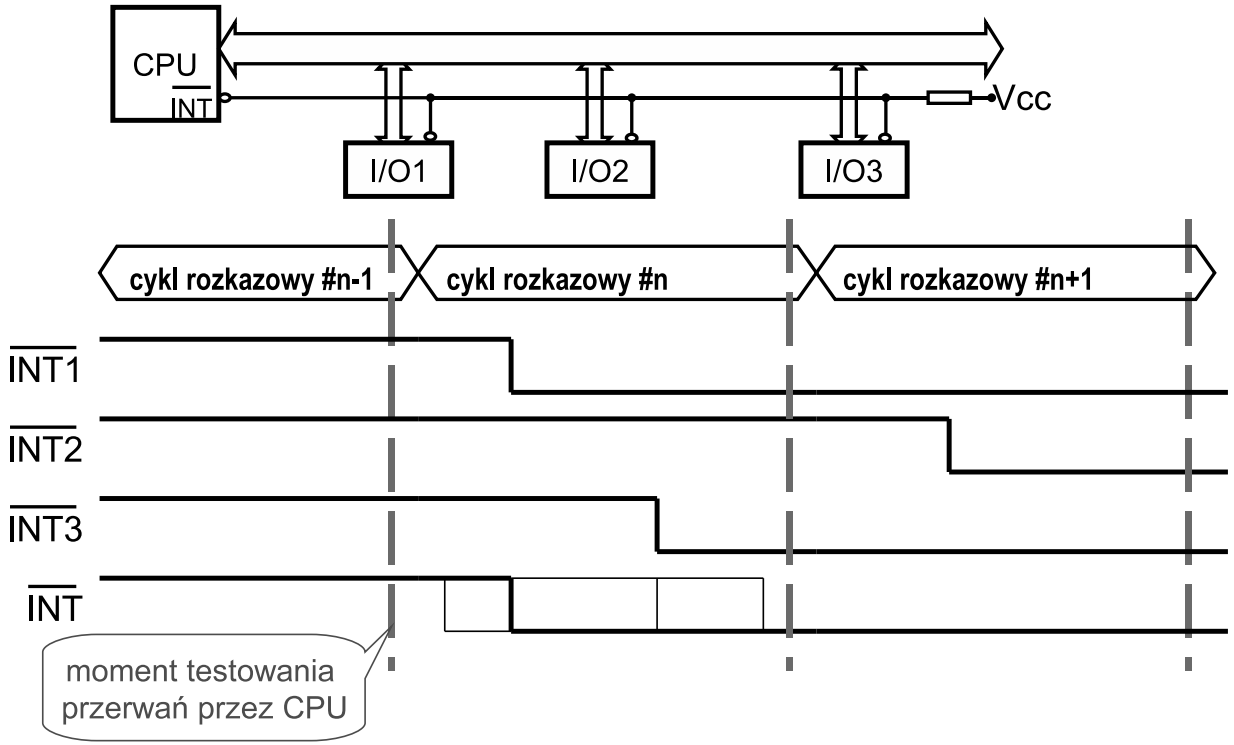
Programowa realizacja hierarchii przerwań

- *programowo* - poprzez wspólną procedurę obsługi przerwań. Jest on arbitrem systemu przerwań (rozpoznaje źródła aktualnych przerwań i decyduje o kolejności ich obsługi)



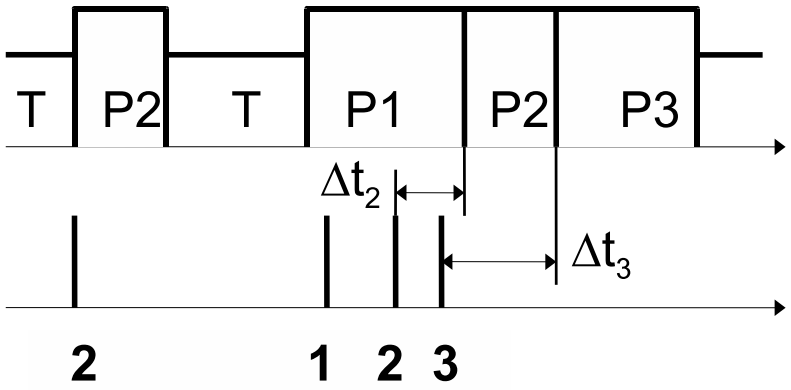
Asynchroniczność przerwań

- Przerwania z różnych źródeł pojawiają się w dowolnych, niezależnych od siebie, chwilach czasu,
- z punktu widzenie procesora przerwania 1 i 3 wystąpiły jednocześnie.



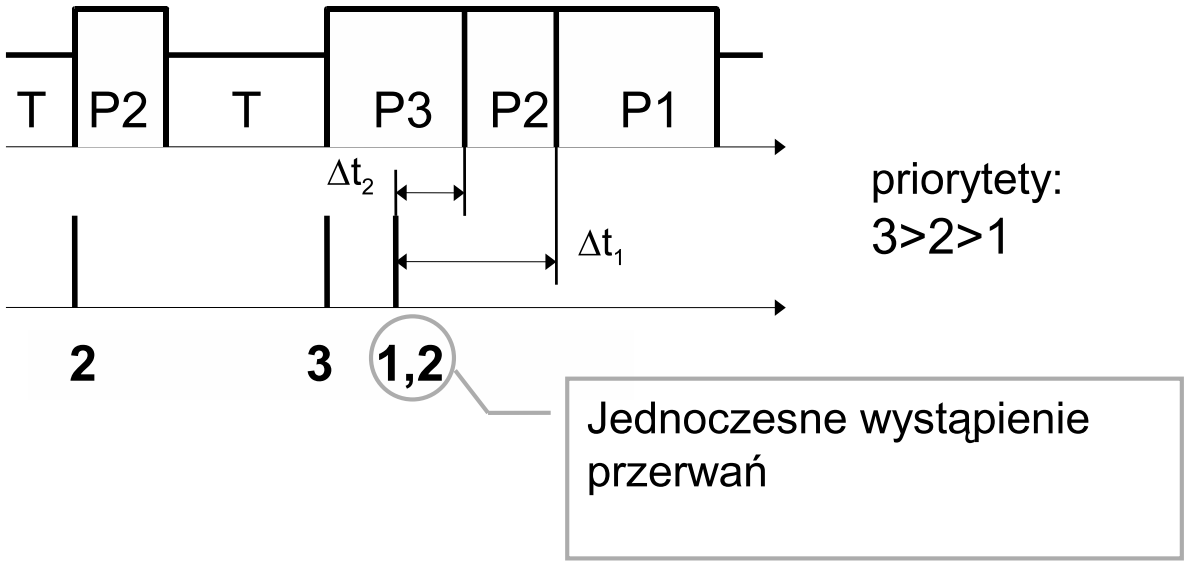
System przerwań bez priorytetów

- Opóźnienia $\{\Delta t_2, \Delta t_3\}$ w reakcji na obsługę przerwań,
- Możliwość zgubienia przerwania podczas tych opóźnień,
- Maksymalny czas oczekiwania na obsługę przerwania może być równy sumie czasów obsługi pozostałych przerwań w systemie,



System przerwań z priorytetami

- Istnieje hierarchia ważności przerwań,
- Opóźnienia $\{\Delta t_1, \Delta t_2\}$ w reakcji na obsługę przerwań,
- Przerwania o niższych priorytetach mogą dłużej czekać na obsługę (w skrajnych przypadkach mogą być nie obsłużone),



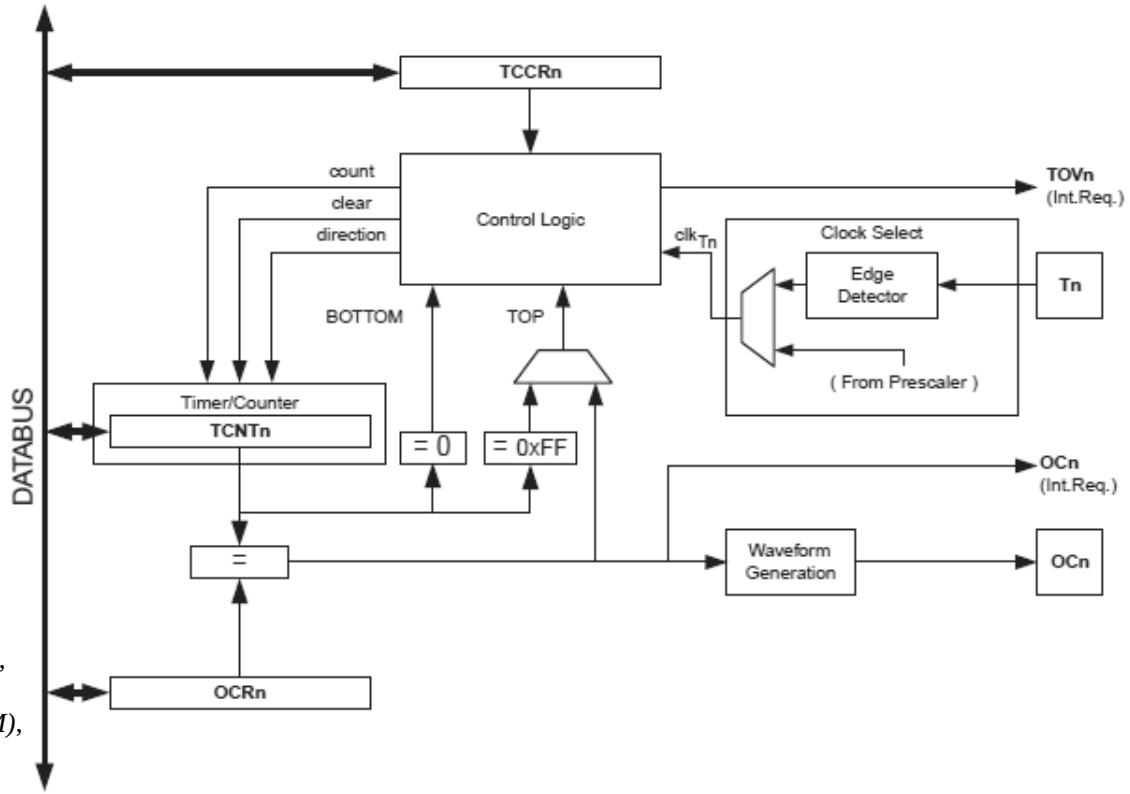
Rodzaje przerwań

- Przerwania zegarowe - odmierzanie czasu,
- Przerwania od urządzeń zewnętrznych - nieregularne,
- Przerwania od układów kontrolujących pracę systemu - o najwyższym priorytecie. Sygnalizują stan pracy jak
  - zanik zasilania,
  - błąd/wyjątek procesora,
  - inne

Liczniki i timery

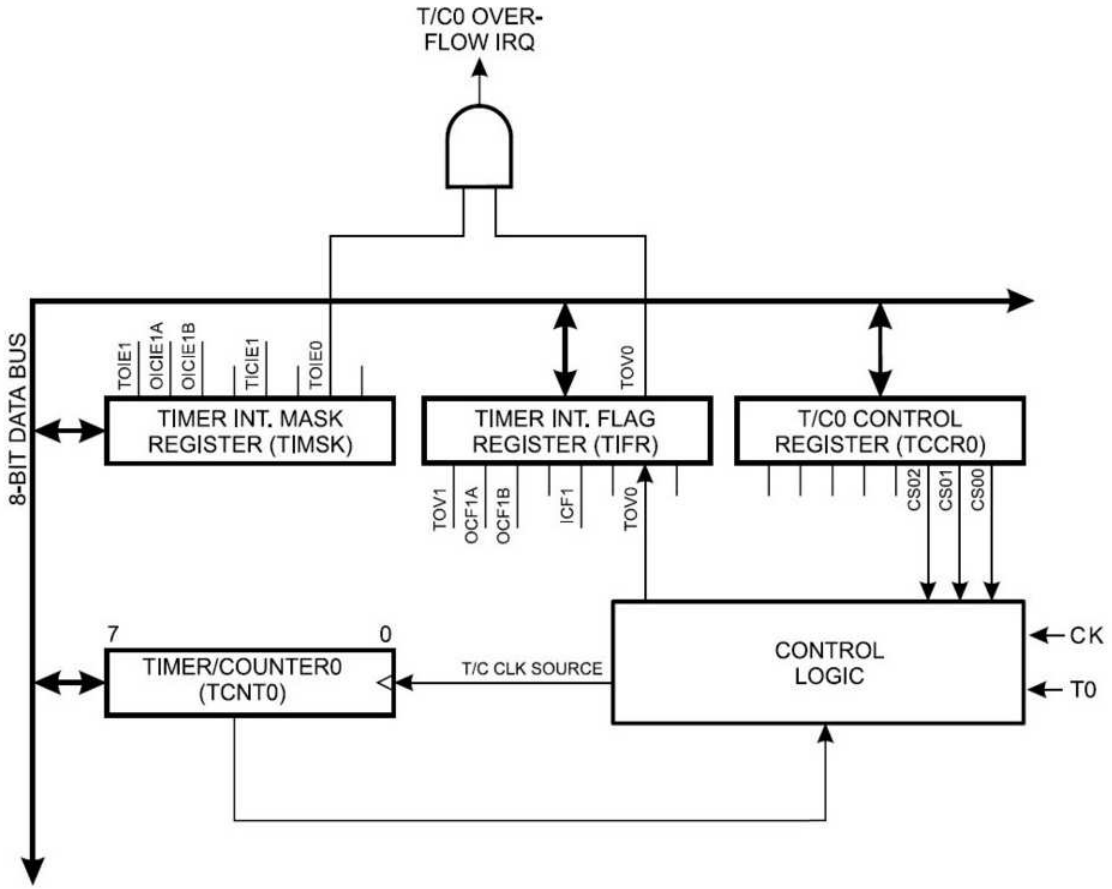
- Timery są to liczniki służące do odmierzania okresów czasu,
- Częstotliwość pracy licznika jest określana poprzez podział częstotliwości zegara,
- Timery (liczniki) mogą mieć różną długość - zazwyczaj 8 albo 16 bitów,
- Przerwanie od timera generowane jest w momencie przepełnienia licznika.

Timer0 w μC Atmel ATMega32



- Licznik pojedynczy,
- Automatyczne zerowanie (Auto Reload),
- Generator *Pulse Width Modulator (PWM)*,
- generator częstotliwości,
- Licznik zdarzeń zewnętrznych,
- 10-bitowy prescaler,
- Przepełnienie (TOV0 and OCF0).

Timer0 w μC Atmel ATMega32 - schemat blokowy



Uruchomienie timera0 i timera1

1. Ustawienie trybów pracy timera. Normalny tryb pracy jest domyślny. Pozostałe tryby omówione będą na następnych wykładach.
2. Ustawienie *prescalera* określenie częstotliwości pracy zegra licznika w oparciu o zegar systemowy. Dostępne dzielniki to $N = \{1, 8, 64, 256, 1024\}$ - tyle razy można zmiejszyć częstotliwość zegara systemowego,
3. Ustawienie etykiety wektora przerwań danego przerwania,
4. Uruchomienie wszystkich przerwań oraz przerwań timera (*rejestr Timer/Counter Interrupt Mask Register*),
5. Wystartowanie timera
6. Obsługa przerwań.

Struktura programu obsługi przerwań (Bascom-AVR)

```
$crystal = 18432000
$regfile = "m32def.dat"
$baud = 19200


Config Timer1 = Timer , Prescale = 8
Enable Timer1
On Timer1 _timer1

Enable Int0
On Int0 _int0

Enable Interrupts

Start Timer1

Do
'Petla glowna
Loop

End

_int0:
'obsługa przerwania int0
print "Tu program obslugi int0"
Return

_timer1:
'obsłyga przerwania od timera1, po przepełnieniu licznika
Print "Tu program obslugi timet1"
Timer1 = 32000
Return
```

Zadania na ćwiczenia

1. Wykorzystując przerwanie *int0* zrealizuj system, który zlicza wciśnięcia przycisku i wypisuje wynik na łącze szeregowe,
2. Posługując się timerem pracującym w normalnym trybie zrealizuj generator o częstotliwości $f = \ldots$ i współczynniku wypełnienia $\omega = \ldots$ podanych przez prowadzącego. Po naciśnięciu przycisku generowana częstotliwość powinna zwiększać się o 10%.