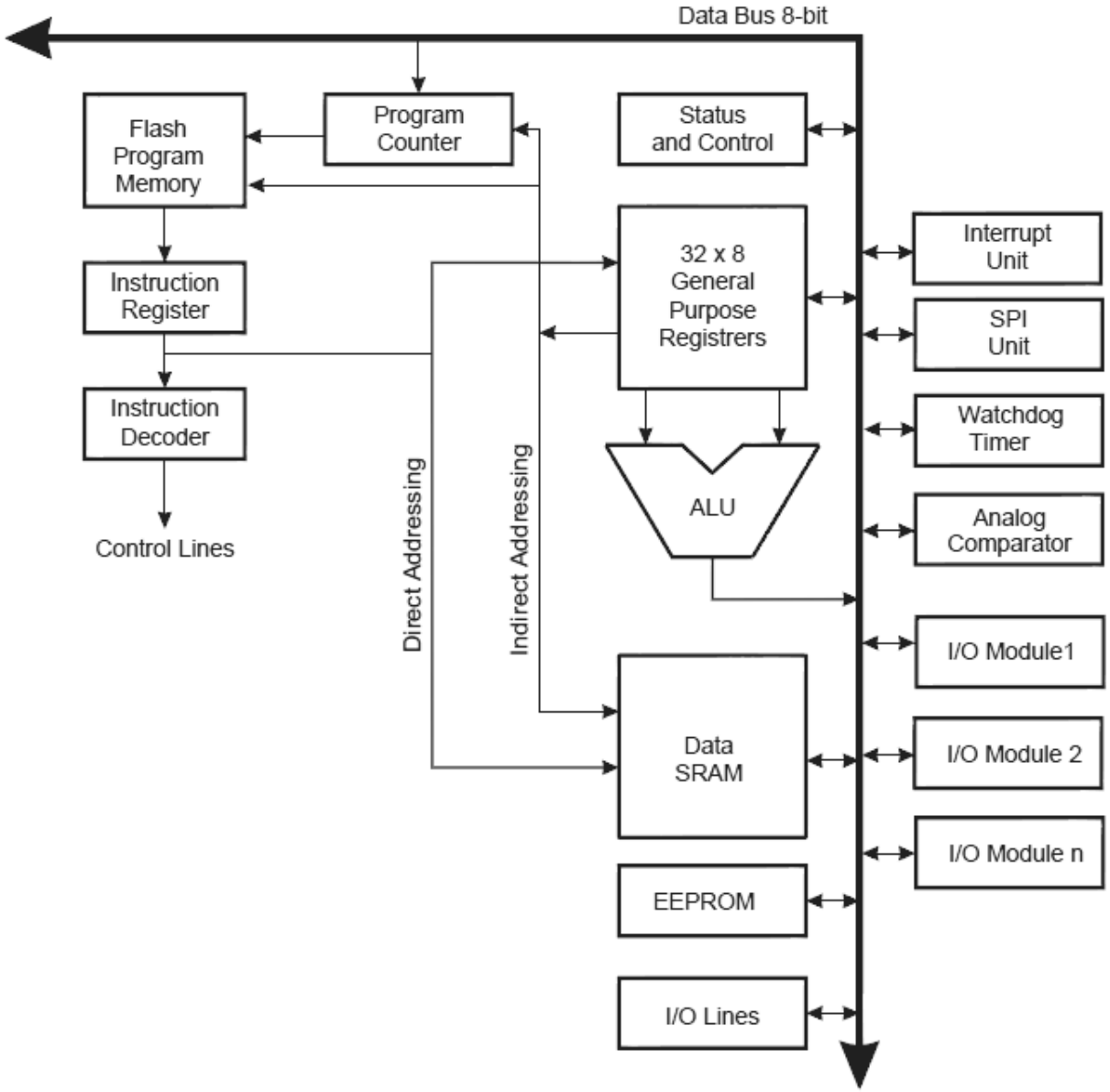
Mikrokontroler AVR ATmega32 - wykład 9

Adam Szmigielski
aszmigie@pjwstk.edu.pl

Cechy μC ATmega32

1. Architektura RISC - 131 instrukcji procesora (wykonywanych w jednym cyklu zegara), 32×8-bitowych rejestrów ogólnego przeznaczenia,
2. Nieulotna pamięć programu i danych - $32kB$ pamięci programu, ISP, z opcją Bootloadera, 1024 bajty pamięci danych EEPROM, $2kB$ wewnętrznej pamięci SRAM,
3. Peryferia - 8-bitowe i 16-bitowy timery/liczniki z opcją preskalera, cztery kanały PWM, $8 \times 10 - bitowych$ przetworników ADC, interface Two-wire, programowalny Serial USART, interface Master/Slave SPI, programowalny Watchdog, komparator analogowy,
4. Inne cechy - wewnętrzny generator RC, wewnętrzne i zewnętrzne źródła przerwań, układy oszczędności energii, zasilanie $4.5 - 5.5V$, częstotliwość pracy zegara do 16MHz.

Schemat blokowy ARV ATmega32



Wyprowadzenia μC AVR ATmega32

- **VCC** - Zasilanie układu,
- **GND** - Masa,
- **Port A** ($PA_7 \ldots PA_0$) - Wejścia przetworników AC,
- **Port B** ($PB_7 \ldots PB_0$) - Dwukierunkowe porty I/O z wewnętrznymi rezystorami podciągającymi,
- **Port C** ($PC_7 \ldots PC_0$) - Dwukierunkowe porty I/O z wewnętrznymi rezystorami podciągającymi,
- **Port D** ($PD_7 \ldots PD_0$) - Dwukierunkowe porty I/O z wewnętrznymi rezystorami podciągającymi,
- $\overline{RESET}$ - wejście przerwania RESET,
- **XTAL1, XTAL2** - wejścia zewnętrznego generatora (zegara),
- **AREF** - wejście zewnętrznego napięcia odniesienia przetwornika AC,

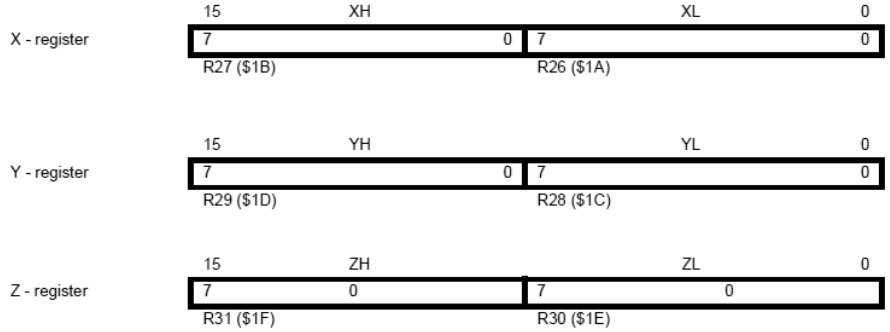
Dostępne rejestry μC AVR ATmega32

- **Status Register -** rejestr flag monitorujący stan operacji ALU (znaczenia bitów w dokumentacji),
- **Rejestry ogólnego przeznaczenia**

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

- **Wskaźnik stosu** (ang. Stack Pointer) - Dwa 8-bitowe rejestry SPH, SPL wskazujący obecne miejsce na stosie (szczyt stosu). Musi być powyżej adresu $\$60F$ w pamięci RAM, jest dekrementowany.

- **Rejestry X, Y i Z** - rejestry ogólnego przeznaczenia, dodatkowo służące do adresowania pośredniego w przestrzeni danych:

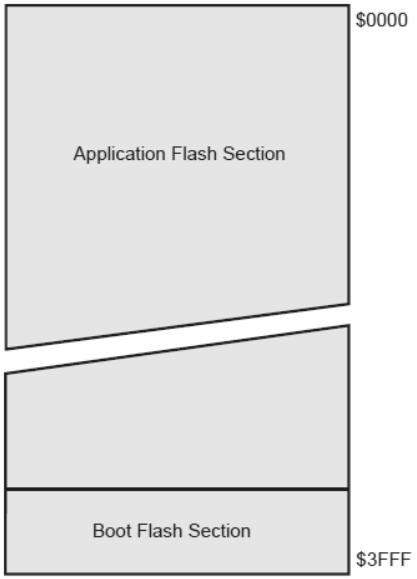


- **Rejestry EEARH, EEARL, EEDR i EECR** - rejestry umożliwiające dostęp do pamięci danych EEPROM. Znaczenia rejestrów: EEARH i EEARL - adres, EEDR - wpisywane lub odczytywane dane, EECR - rejestr kontrolny.

Dokładny opis wszystkich rejestrów w dokumentacji.

Pamięć programu μC AVR ATmega32

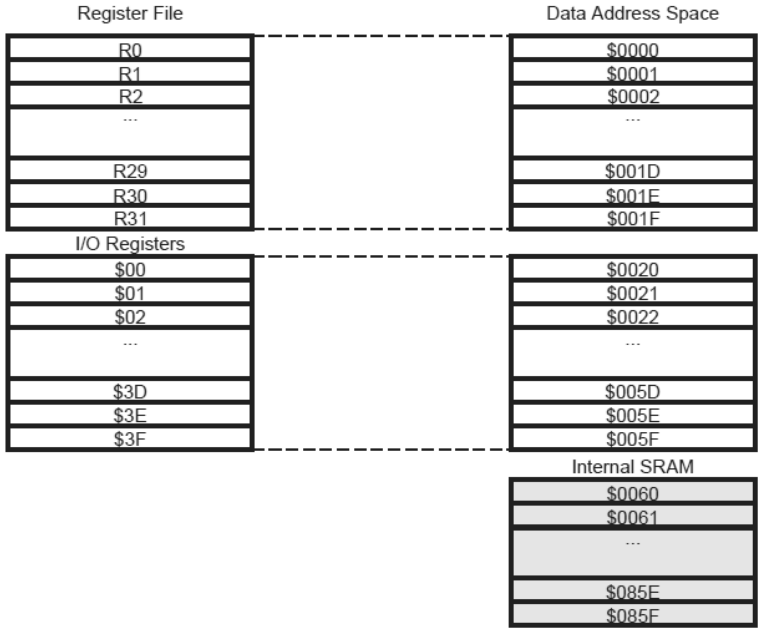
- $32kB$ pamięci Flesh do przechowywania programu o organizacji $16k \times 16$. Instrukcje AVR są 16 lub 32 bitowe. Licznik programu (PC) jest 14 bitowy, umożliwia zaadresowanie $2^{14} = 16kB$ komórek pamięci.
- Pamięć programu jest podzielona na dwa obszary - *sekcja aplikacji* i *sekcja bootloadera*.



- Pamięć FLESH przewidziana jest na 10.000 cykli zapisu/kasowania.

Pamięć danych SRAM μC AVR ATmega32

- Najmłodsze 2144 komórek pamięci zawiera *rejestry ogólnego przeznaczenia* (32), *rejestry wejścia/wyjścia* (64 komórek) oraz wewnętrzną pamięć SRAM (2048 komórek),



- Istnieje pięć różnych trybów adresowania: Bezpośrednie, Pośrednie, Pośrednie z przesunięciem, Pośrednie z pre i post inkrementacją. Adresowania pośrednie odbywją się z wykorzystaniem rejestrów X,Y i Z,
- Do każdej komórka pamięci SRAM można się odwołać w dowolnym trybie adresowania.

Nieulotna pamięć danych EEPROM μC AVR ATmega32

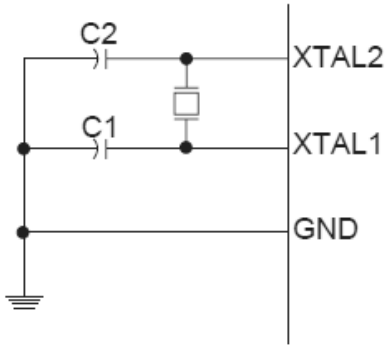
- μC AVR ATmega32 zawiera 1024 bajty nieulotnej pamięci danych, zorganizowanej w osobnej przestrzeni adresowej.
- W przestrzeni tej mogą być zapisane lub odczytane pojedyncze bajty,
- Pamięć EEPROM data przewidziana jest na 100.000 cykli zapisu/kasowania,
- Dostęp do tej pamięci możliwy jest za pomocą rejestrów **EEARH, EEARL, EEDR i EECR**. Rejestry te pełnią następujące funkcje: EEARH i EEARL - adres, EEDR - wpisywane lub odczytywane dane, EECR - rejestr kontrolny,
- Pamięć EEPROM data jest chroniona przed zniszczeniem, które może powodować zbyt niskie napięcie zasilania V_{CC}.

Pamięć urządzeń wejścia/wyjścia μC AVR ATmega32

- Do wszystkich peryferii μC AVR ATmega32 odwołuje się za pomocą rejestrów I/O umieszczonych w tej samej przestrzeni adresowej co *Rejestry ogólnego przeznaczenia* i pamięć SRAM,
- Używając mnemoników *IN* lub *OUT* należy używać adresów $\$00 - \$3F$, w przypadku, gdy odwołujemy się do nich używając mnemoników *LD* i *ST* należy dodać $\$20_{HEX}$ tj. 32_{10} do ich adresu (gdyż poprzedzają je 32 rejestry ogólnego przeznaczenia).
- Niektóre rejestry I/O, w zależności od pełnionej funkcji w urządzeniach peryferynych, są chronione przed zapisem. Do niektórych rejestrów można się odwołać dopiero po ustawieniu odpowiedniej flagi (patrz dokumentacja).

Źródła sygnału zegara w μC AVR ATmega32

- **Wewnętrzny generator** - nie wymaga żadnych elementów zewnętrznych. Uzyskane nominalne częstotliwości wynoszą 1, 2, 4, lub 8 MHz,
- **Zewnętrzny rezonator kwarcowy**

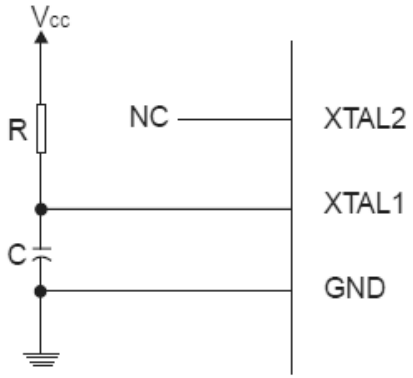


Wejścia XTAL1 i XTAL2 są wejściami wzmacniacza wewnętrznego układu. Maksymalna częstotliwość pracy wynosi $16MHz$,

- **Zewnętrzny generator niskich częstotliwości** - Kwarc powinien być dołączony jak w poprzednim przypadku. Dołączane są wewnętrzne pojemności $36pF$. Uzyskana częstotliwość wynosi $32,768kHz$.

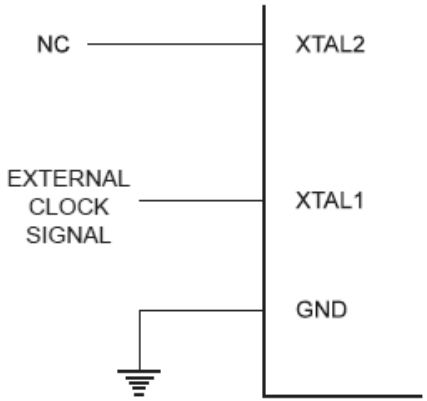
Źródła sygnału zegara w μC AVR ATmega32 cd.

- **Zewnętrzny generator RC**



Zewnętrzne elementy R i C są wykorzystane do budowy generatora,

- **Zegar zewnętrzny**



Umożliwia podanie zewnętrznego sygnału zegarowego.

Reset w μC AVR ATmega32

- *Power-on Reset* - μC resetuje się wskutek obniżenia napięcia zasilającego poniżej pewnego progu V_{POT},
- *External Reset* - μC resetuje się wskutek podania na pin RESETU niskiego poziomu przez odpowiednio długi czas,
- *Watchdog Reset* - μC resetuje się wskutek przekroczenia okresu oczekiwania przez Watchdoga (Watchdog musi być włączony),
- *Brown-out Reset* - μC resetuje się, gdy napięcie zasilania V_{CC} spadnie poniżej poziomu V_{BOT} (detektor Brown-out musi być włączony),
- *JTAG AVR Reset* - μC resetuje się, gdy jest logiczna 1 w Reset Register (możliwość wykrycia tylko w ramach systemu JTAG).

System przerwań μC AVR ATmega32

Vector No.	Program Address(2)	Source	Interrupt Definition
1	$000(1)	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

Dokładny opis na następnych wykładach.

Alternatywne funkcje portów μC AVR ATmega32

Port Pin	Alternate Function
PA7	ADC7 (ADC input channel 7)
PA6	ADC6 (ADC input channel 6)
PA5	ADC5 (ADC input channel 5)
PA4	ADC4 (ADC input channel 4)
PA3	ADC3 (ADC input channel 3)
PA2	ADC2 (ADC input channel 2)
PA1	ADC1 (ADC input channel 1)
PA0	ADC0 (ADC input channel 0)

Port Pin	Alternate Functions
PB7	SCK (SPI Bus Serial Clock)
PB6	MISO (SPI Bus Master Input/Slave Output)
PB5	MOSI (SPI Bus Master Output/Slave Input)
PB4	$\overline{SS}$ (SPI Slave Select Input)
PB3	AIN1 (Analog Comparator Negative Input) OC0 (Timer/Counter0 Output Compare Match Output)
PB2	AIN0 (Analog Comparator Positive Input) INT2 (External Interrupt 2 Input)
PB1	T1 (Timer/Counter1 External Counter Input)
PB0	T0 (Timer/Counter0 External Counter Input) XCK (USART External Clock Input/Output)

Port Pin	Alternate Function
PC7	TOSC2 (Timer Oscillator Pin 2)
PC6	TOSC1 (Timer Oscillator Pin 1)
PC5	TDI (JTAG Test Data In)
PC4	TDO (JTAG Test Data Out)
PC3	TMS (JTAG Test Mode Select)
PC2	TCK (JTAG Test Clock)
PC1	SDA (Two-wire Serial Bus Data Input/Output Line)
PC0	SCL (Two-wire Serial Bus Clock Line)

Port Pin	Alternate Function
PD7	OC2 (Timer/Counter2 Output Compare Match Output)
PD6	ICP (Timer/Counter1 Input Capture Pin)
PD5	OC1A (Timer/Counter1 Output Compare A Match Output)
PD4	OC1B (Timer/Counter1 Output Compare B Match Output)
PD3	INT1 (External Interrupt 1 Input)
PD2	INT0 (External Interrupt 0 Input)
PD1	TXD (USART Output Pin)
PD0	RXD (USART Input Pin)

Dokładny opis na następnych wykładach.

Przerwania zewnętrzne

- Przerwania zewnętrzne są wyzwalane na pinach INT0, INT1, i INT2,
- Jeśli system przerwań jest włączony wejścia INT0, INT1, i INT2 wykryją przerwanie, nawet jeśli piny są ustawione jako wyjścia - umożliwia to wykorzystanie ich jako przerwania programowe,
- Przerwania mogą być wyzwalane zboczem narastającym, opadającym lub poziomem (INT2 tylko zboczem).

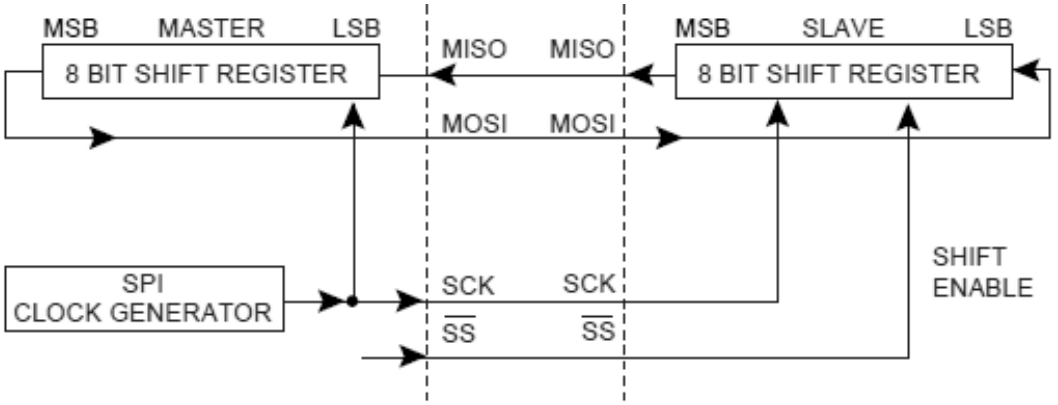
Dokładny opis na następnych wykładach.

Liczniki i timery - Timer0 i Timer1

- Możliwość wyzerowania liczników,
- Generator częstotliwości,
- Licznik zdarzeń zewnętrznych,
- Możliwość pracy w trybie PWM,
- Preskaler zegara $10 - bitowy$,
- Przepełnienie licznika jest źródłem przerwania.

Dokładny opis na następnych wykładach.

Interface szeregowy SPI - ang. Serial Peripheral Interface

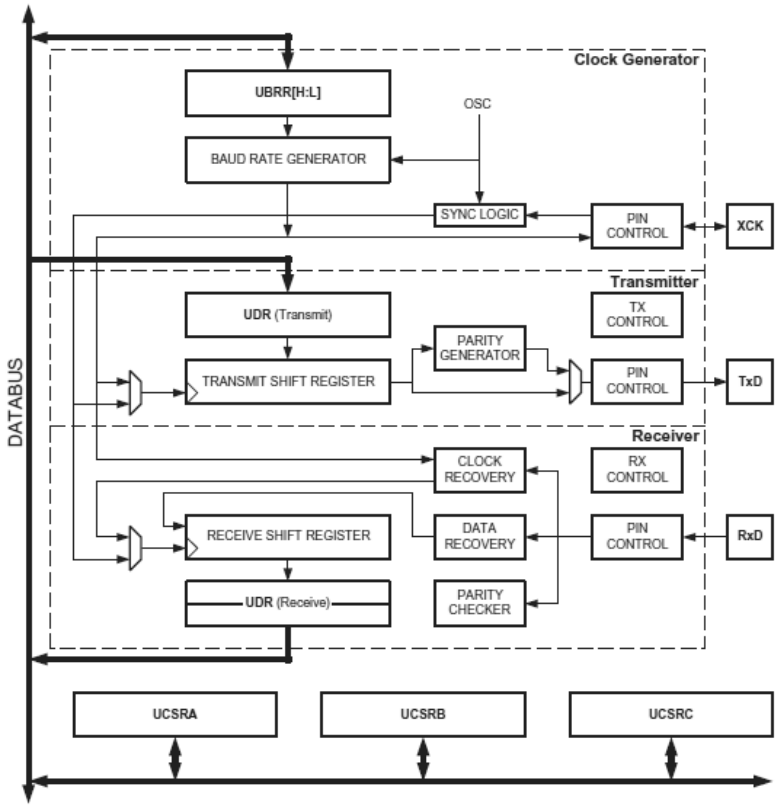


Pin	Direction, Master SPI	Direction, Slave SPI
MOSI	User Defined	Input
MISO	Input	User Defined
SCK	User Defined	Input
$\overline{SS}$	User Defined	Input

- Full-duplex, synchroniczny transfer danych, 7 programowalnych prędkości transmisji,
- Możliwość pracy w trybie Master lub Slave,
- Koniec transmisji identyfikowany flagą przerwań.

Dokładny opis na następnych wykładach.

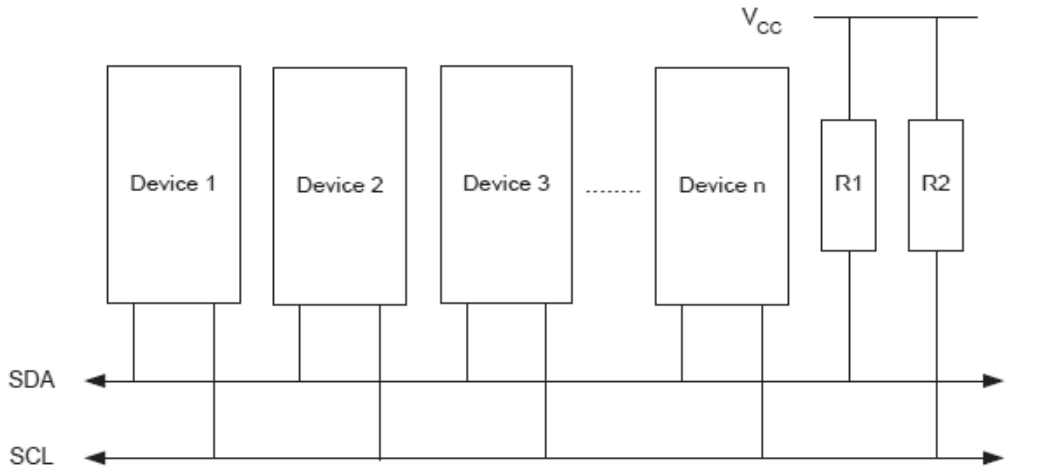
Interfejs szeregowy USART



- Full Duplex Operation, transmisja synchroniczna i asynchroniczna,
- Możliwość transmisji danych 5, 6, 7, 8 lub 9 bitowych z 1 lub 2 bitami stopu, bit parzystości wykrywany sprzętowo, wykrywanie błędów transmisji,
- Przerwania: TX Complete, TX Data Register Empty, RX Complete.

Dokładny opis na następnych wykładach.

Interface szeregowy - Two-wire



Term	Description
Master	The device that initiates and terminates a transmission. The master also generates the SCL clock.
Slave	The device addressed by a master.
Transmitter	The device placing data on the bus.
Receiver	The device reading data from the bus.

- Obsługa trybów Master i Slave, arbitraż Multi-master,
- 7-bitowy adres (128 adresów Slave), w pełni programowalny adres Slave-a w ramach General Call Support
- Prędkość transmisji do $400kHz$

Dokładny opis na następnych wykładach.

Przetworniki analogowo-cyfrowy ADC

- Rozdzielczość 10-bitowa,
- Nieliniowość na poziomie $\frac{1}{2}$ LSB,
- Dokładność bezwzględna na poziomie ± 2 LSB,
- Czas konwersji $65 \div 260 \mu s$,
- 8 multipleksowanych kanałów,
- 7 kanałów różnicowych,
- 2 wejść różnicowych ze wzmocnieniem $10\times$ and $200\times$,
- Zakres mierzonych napięć $0 \div V_{CC}$,
- Wybór napięcia odniesienia: wewnętrzne $2,56V$ lub zewnętrzne,
- W tryb konwersji *Single* przetwornik pracuje w sposób ciągły,
- Zgłaszane przerwanie po zakończeniu konwersji.

Interface JTAG

JTAG Debuging System umożliwia:

- Kontrolę wszystkich jednostek peryferyjnych,
- Kontrolę pamięci RAM, EEPROM data i pamięć FLESH,
- Kontrolę rejestrów wewnętrznych oraz rejestr PC,
- Analizę uszkodzeń,
- Programowanie pamięci Flash, EEPROM,
- Odblokowanie flag,
- inne.

Programowanie μC AVR ATmega32

Programowanie μC AVR ATmega32 może odbywać się:

- Z użyciem programatorów,
- poprzez bootloader.

Istnieje kilka środowisk programistycznych, w skład których wchodzą kompilator, symulator, programator, terminal i inne elementy. Najbardziej popularne to:

- AVR Studio,
- Bascom AVR.

Bascom-AVR

To środowisko będzie używane na ćwiczeniach.

- Licencja: *Shareware, Wersja DEMO do 4k kodu*
- System operacyjny: Windows XP, 98, NT
- Środowisko do pisania i debugging-u układów AVR
- Do pobrania na stronie producenta:
  http://www.bipom.com

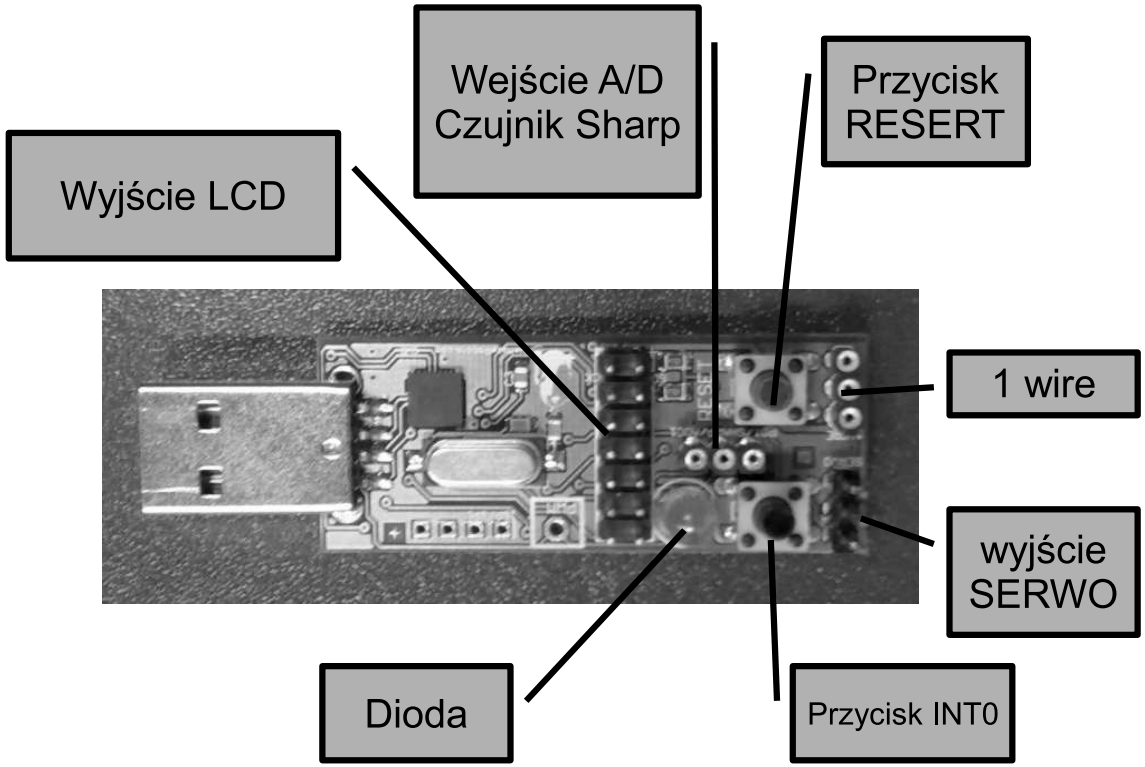
- Wymaga instalacji

BASCOM-AVR, język Basic - typy zmiennych

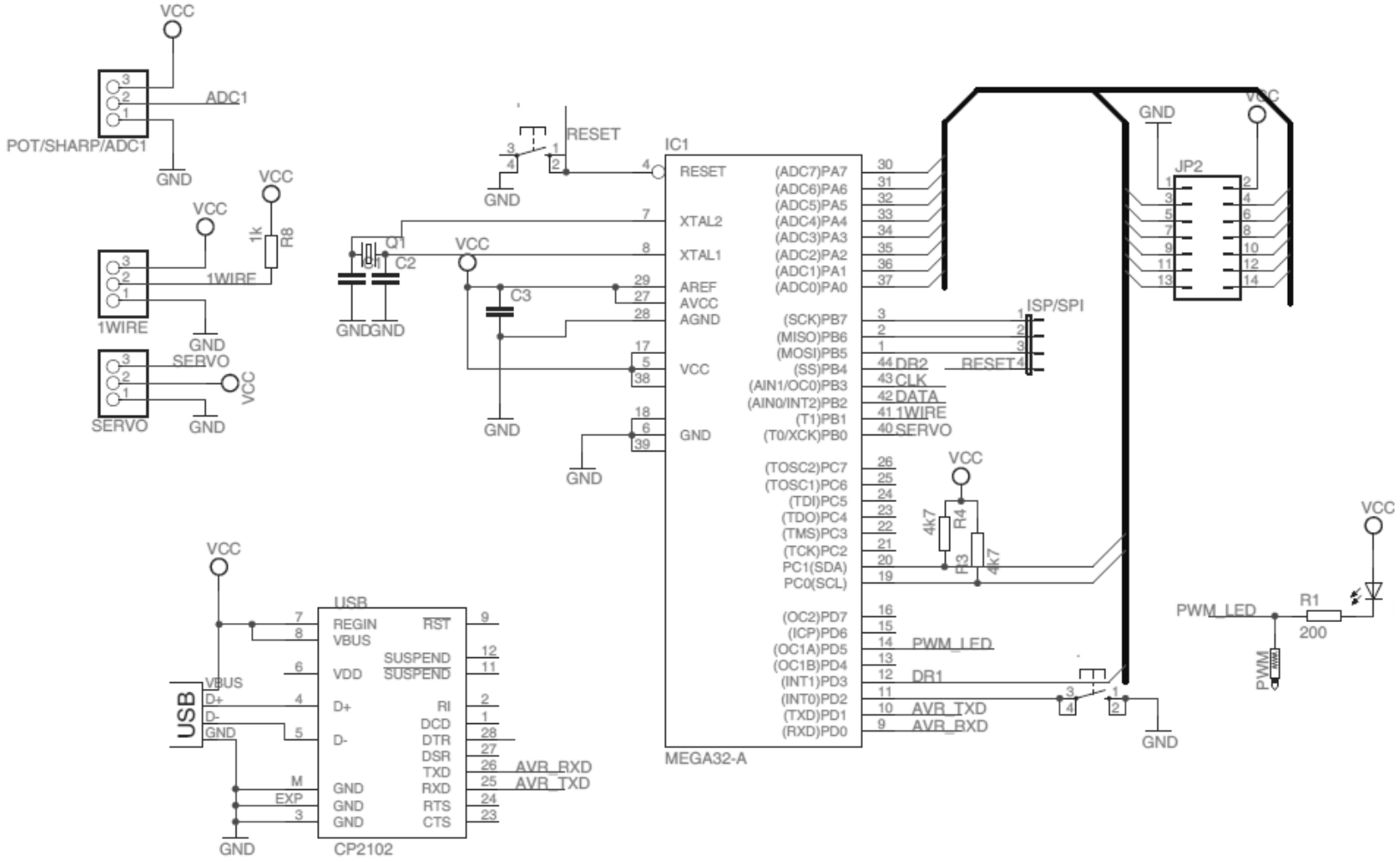
- Bit ($\frac{1}{8}$ byte),
- Byte (1 byte) - liczba 8-bitowa bez znaku ($0 \div 255$),
- Integer (dwa bajty) - liczby całkowitoliczbowe ze znakiem ($-32,768 \div +32,767$),
- Word (dwa bajty) - liczba całkowitoliczbowa bez znaku ($0 \div 65535$),
- Long (cztery bajty) - liczby całkowitoliczbowe ze znakiem ($-2147483648 \div 2147483647$),
- Single (32 bity) liczba zmiennoprzecinkowa ze znakiem ($1.5x10^{-45} \div 3.4x10^{38}$),
- Double (64 bity) liczba zmiennoprzecinkowa ze znakiem ($5.0x10^{-324} \div 1.7x10^{308}$),
- String (do 254 bajtów) -przechowywane jako bajty i zakończone 0-bajtem tj. rozmiar stringa jest o jeden większy.

Układy AVR nie posiadają koprocesora. Operacje na liczbach zmiennoprzecinkowych (*Single, Double*) są emulowane.

Kontroler ARV ATmega32 używany na ćwiczeniach



Kontroler ARV ATmega32 - Schemat ideowy



Zadania na ćwiczenia

1. Zrealizuj kalkulator wykonujący operacje dodawania "+" i odejmowania "-" na dwóch operandach w notacji infiksowej. Do wczytywania użyj funkcji *input()*. Każdą daną (w tym liczby i znaki) należy zatwierdzić enterem.
2. Zrealizuj kalkulator wykonujący operacje dodawania "+", odejmowania "-", mnożenia "*" i dzielenia "\" na dwóch operandach w notacji infiksowej. Do wczytywania użyj funkcji *waitkey()*. Operandy powinny być typu *byte*, wynik typu *integer*. W przypadku przekroczenia zakresu operandów lub nieprawidłowych danych należy wyświetlić stosowny komunikat.
3. Zrealizuj kalkulator wykonujący operacje dodawania "+", odejmowania "-", mnożenia "*" i dzielenia "\" na dwóch operandach w notacji infiksowej. Wykorzystując funkcję *input()* wczytaj całe wyrażenie, następnie dokonaj analizy poprawności danych. Załóż, że dane są reprezentowane poprzez bajt a wynik może być innym typem danych.