



WYDZIAŁ FIZYKI  
i INFORMATYKI STOSOWANEJ  
Uniwersytet Łódzki



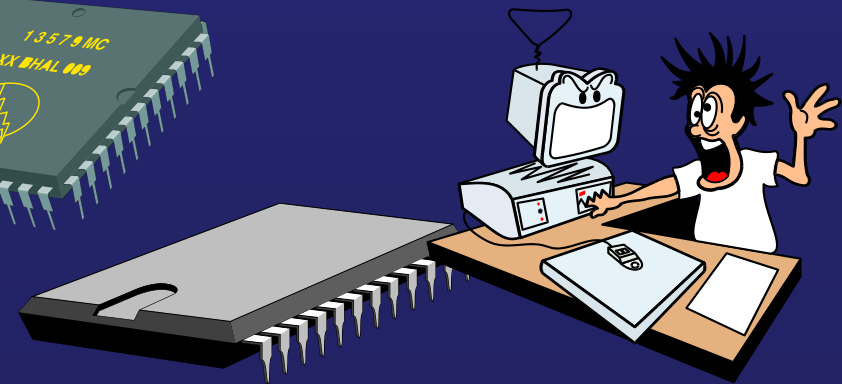
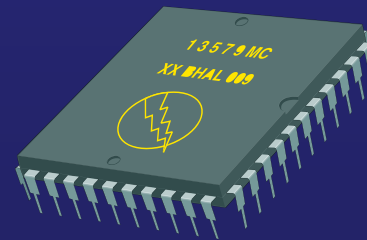
## *Systemy wbudowane*

*Witold Kozłowski*



*Zakład Fizyki i Technologii Struktur Nanometrowych  
90-236 Łódź, Pomorska 149/153*

<https://std2.phys.uni.lodz.pl/mikroprocesory/>



# *Systemy wbudowane*

Kierunek: Informatyka  
PRACOWNIA DYDAKTYCZNA

## **Uwaga !!!**

**Proszę o wyłączenie  
telefonów komórkowych**

**na wykładzie i laboratorium**

# *Systemy wbudowane*

Kierunek: Informatyka  
PRACOWNIA DYDAKTYCZNA

## Wykład 1.

# Wprowadzenie do mikroprocesorów

# Systemy wbudowane

Kierunek: Informatyka  
PRACOWNIA DYDAKTYCZNA

**System wbudowany** (ang. *Embedded system*) – system komputerowy specjalnego przeznaczenia, który staje się integralną częścią obsługiwanego przez niego sprzętu.

System wbudowany spełnia określone wymagania, zdefiniowane do zadań, które ma wykonywać. Nie można nim więc nazywać typowego wielofunkcyjnego komputera osobistego. Każdy system wbudowany oparty jest na mikroprocesorze (lub mikrokontrolerze), zaprogramowanym do wykonywania ograniczonej ilości zadań, lub nawet do jednego.

Zależnie od złożoności wykonywanych zadań, może zawierać oprogramowanie dedykowane wyłącznie temu urządzeniu ([firmware](#)), lub może to być system operacyjny wraz ze specjalizowanym oprogramowaniem. Może o tym decydować też stopień niezawodności jakie ma oferować dany system wbudowany. Ogólną zasadą jest, im mniej złożone i specjalizowane jest oprogramowanie, tym bardziej system jest niezawodny, oraz może szybciej reagować na zdarzenia krytyczne.

Niezawodność systemu może być zwiększona, przez rozdzielenie zadań na mniejsze podsystemy, a także przez **redundancję**. Polegać to może np. na zastosowaniu do jednego zadania dwóch identycznych urządzeń, które mogą przejąć zadania drugiego, w przypadku awarii jednego z nich.

## Ciekawostka:

Za pierwszy komputer wbudowany uznaje się ten, który sterował amerykańskim statkiem kosmicznym Apollo. Pierwszy komputer wbudowany produkowany masowo sterował raketą LGM-30 Minuteman.



# *Systemy wbudowane*

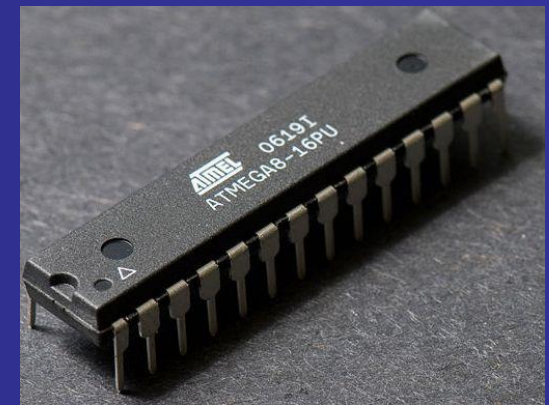
## Zastosowania

Tego typu systemy znajdują zastosowanie we wszystkich dziedzinach, gdyż w obecnych czasach dąży się aby wszystkie urządzenia były inteligentne i zdolne do pracy autonomicznej oraz wykonywały coraz bardziej złożone zadania.





# Mikrokontrolera ATmega8



Budowa mikrokontroler AVR opiera się na architekturze harwardzkiej:

Mikroprocesory AVR należą do grupy układów o architekturze RISC (Reduced Instruction Set Computer)

- rozdzielono przestrzeń adresowej pamięci programu i przestrzeni adresowej pamięci danych
- uzyskano to dzięki zastosowaniu oddzielnych magistral adresowych
- dzięki temu możliwe było zastosowanie słowa o różnej szerokości dla pamięci programu i pamięci danych
- chroni to przed przypadkowym odczytaniem danej i interpretowanie jej jako instrukcji
  
- Większość rozkazów RISC jest realizowana w jednym taktcie zegara co zapewnia szybsze wykonanie programu
- programy pisane dla procesorów RISC charakteryzują się większą spójnością, a co za tym idzie mniejszym kodem wynikowym

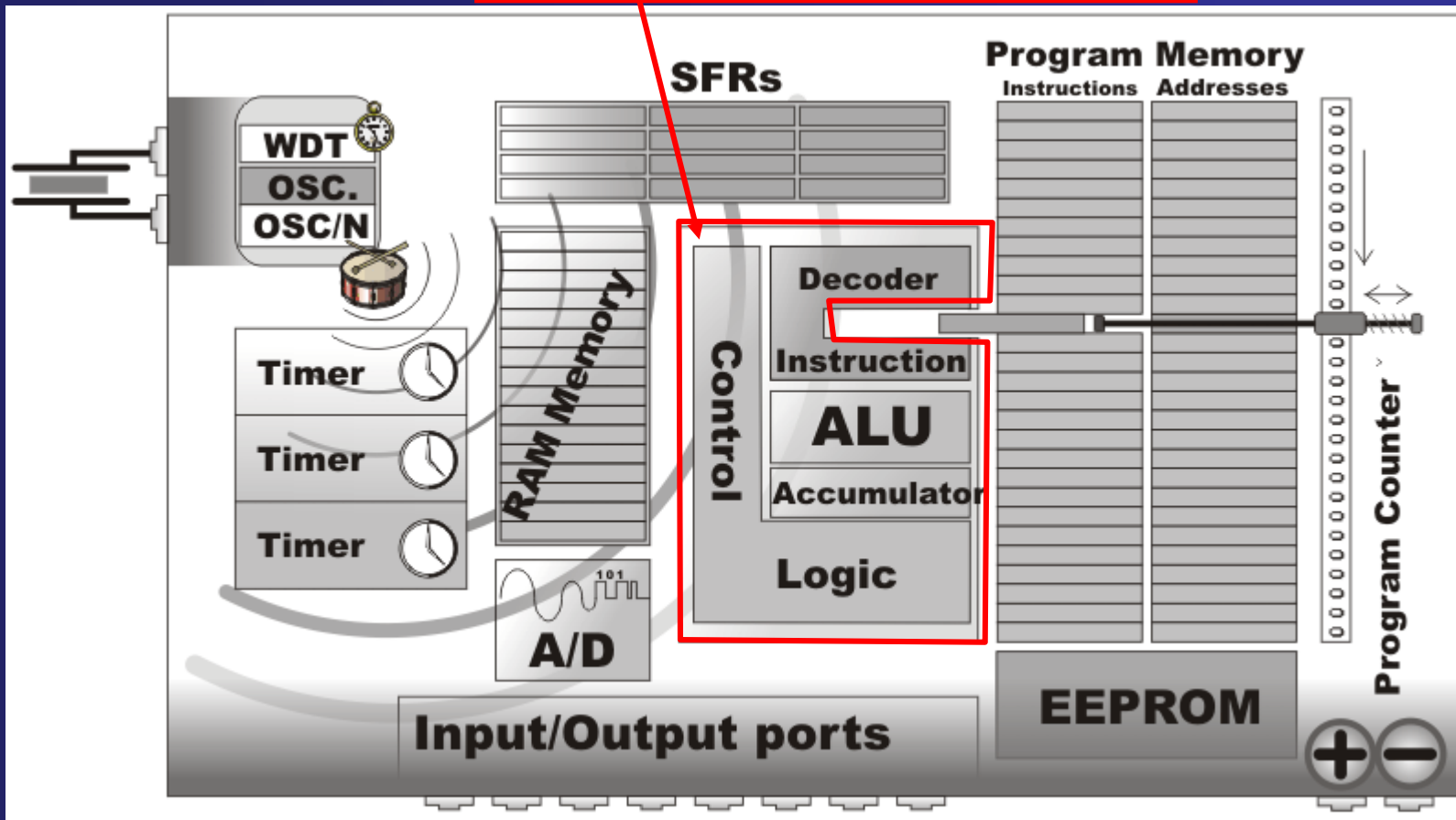
(Architektura 8051 jest określona nazwą CISC (Complex Instruction Set Computer) wykonanie jednego rozkazu CISC wymaga zazwyczaj wykonania wielu operacji, co zwykle trwa kilka taktów zegara)

Cechą wyróżniającą mikrokontrolery AVR jest również

- Zaimplementowanie wielu rejestrów wewnętrznych, z których każdy może pełnić rolę akumulatora podczas wykonywania operacji arytmetycznych i logicznych
- Minimalizuje to liczbę przesłań między rejestrowych, co korzystnie wpływa na szybkość wykonywania programu

# Schemat blokowy mikrokontrolera ATmega8

Sercem mikrokontrolera jest centralna jednostka sterująca (CPU - Central Processing Unit)



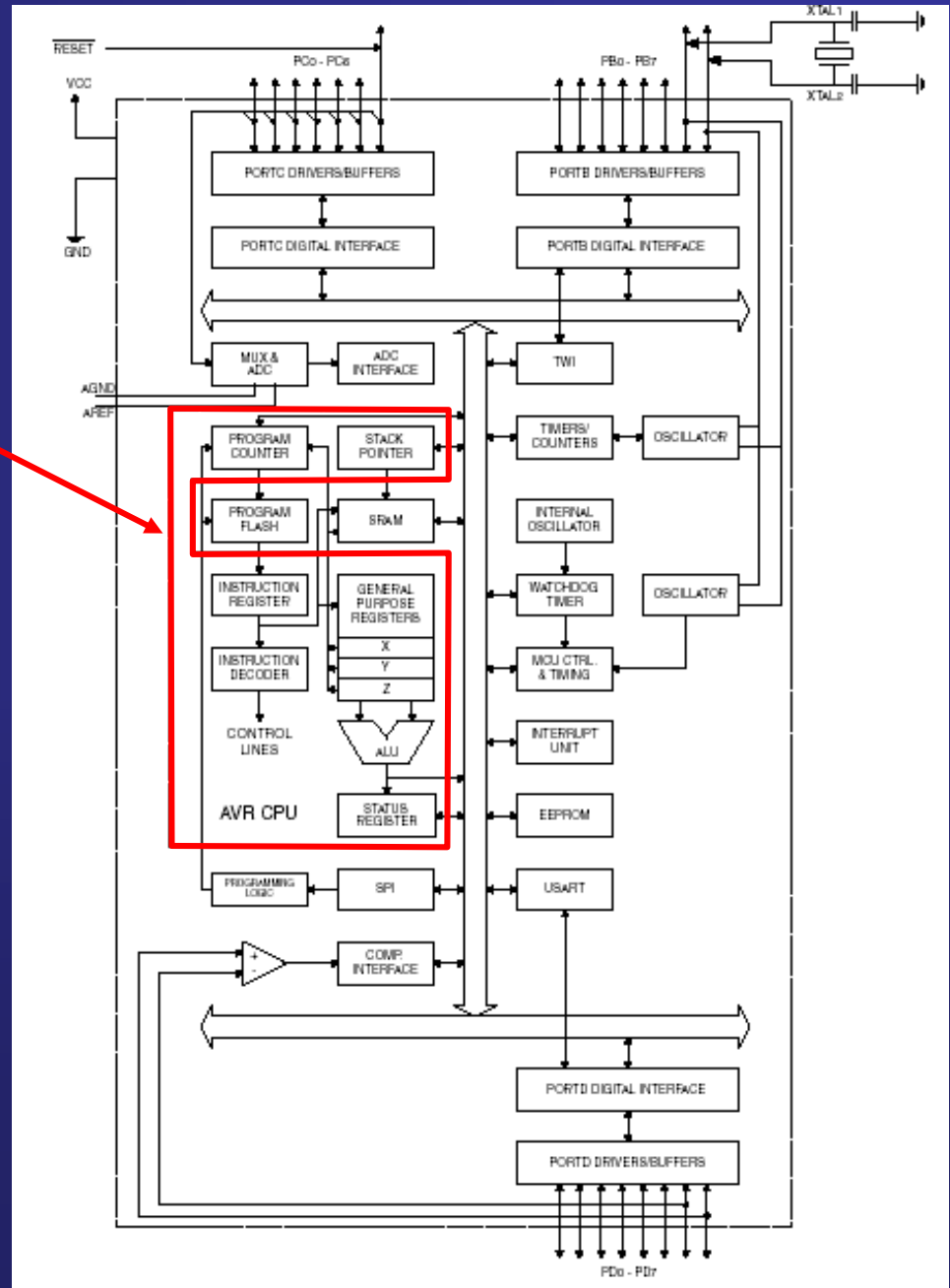
Jej zadaniem jest sterowanie procesem pobierania rozkazów, ich rozpoznawania i wykonywania.



# Schemat blokowy mikrokontrolera ATmega8

Sercem mikrokontrolera jest centralna jednostka sterująca  
(CPU - Central Processing Unit)

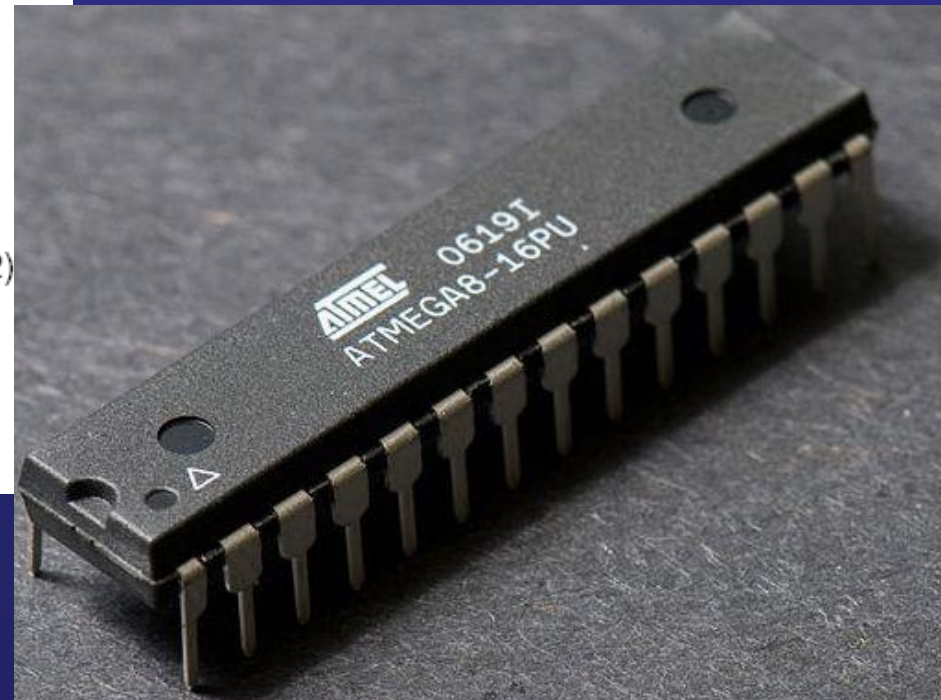
Jej zadaniem jest sterowanie procesem pobierania rozkazów, ich rozpoznawania i wykonywania.



# Rozmieszczenie wyprowadzeń Mikrokontrolera - ATmega8 w obudowie DIP 28

## PDIP

$\overline{\text{RESET}}$ PC6	□ 1	28	□ PC5 (ADC5/SCL)
(RXD) PD0	□ 2	27	□ PC4 (ADC4/SDA)
(TXD) PD1	□ 3	26	□ PC3 (ADC3)
(INT0) PD2	□ 4	25	□ PC2 (ADC2)
(INT1) PD3	□ 5	24	□ PC1 (ADC1)
(XCK/T0) PD4	□ 6	23	□ PC0 (ADC0)
VCC	□ 7	22	□ GND
GND	□ 8	21	□ AREF
(XTAL1/TOSC1) PB6	□ 9	20	□ AVCC
(XTAL2/TOSC2) PB7	□ 10	19	□ PB5 (SCK)
(T1) PD5	□ 11	18	□ PB4 (MISO)
(AIN0) PD6	□ 12	17	□ PB3 (MOSI/OC2)
(AIN1) PD7	□ 13	16	□ PB2 ( $\overline{\text{SS}}$ /OC1B)
(ICP1) PB0	□ 14	15	□ PB1 (OC1A)



# Mikrokontrolera ATmega8

Mikrokontroler ATmega8 ma następujące parametry oraz cechy funkcjonalne:

- Mały pobór mocy
- Zaawansowana architektura RISC (Reduced Instruction Computer) charakteryzująca się:
  - 130 instrukcjami, z których większość jest wykonywana w jednym cyklu maszynowym
  - 32 rejestrami 8-bitowymi ogólnego przeznaczenia
  - Dużą wydajnością 16 MIPS (milion operacji na sekundę) przy częstotliwości zegara 16 MHz
- Pamięć:
  - **8 kB nietlotnej pamięci Flash** o trwałości 10 000 zapisów/kasowań,
  - Ma opcjonalne miejsce na Bootloader
  - **512 B pamięci EEPROM** o trwałości 100 000 zapisów/kasowań (może przechowywać dane do 10 lat)
  - **1 kB pamięci SRAM,**
  - Ma programowalne zabezpieczenia pamięci programu przed odczytem
- Układy peryferyjne:
  - dwa 8 – bitowe czasomierze/liczniki z oddzielnymi preskalerami,
  - jeden 16 – bitowy czasomierz/licznik z oddzielnym preskalerem, możliwość pracy z w trybie Capture oraz Compare
  - zegar czasu rzeczywistego z oddzielnym oscylatorem
  - trzy kanały PWM (OC1, OC1B, OC2)
  - 6-kanałowy przetwornik A/C – cztery kanały o rozdzielczości 10 bitów i dwa o rozdzielczości 8 bitów
  - programowalny USART do transmisji przez RS232
  - szeregowy interfejs Master/Slave SPI
  - programowalny Watchdog z oddzielnym oscylatorem
  - wbudowany komparator analogowy

# Mikrokontrolera ATmega8

Mikrokontroler ATmega8 ma następujące parametry oraz cechy funkcjonalne:

- **Specjalne wyposażenie mikrokontrolera:**
  - zerowanie po włączeniu mikrokontrolera
  - wewnętrzny kalibrowany oscylator RC
  - wewnętrzne oraz zewnętrzne źródła sygnałów przerwań
  - pięć trybów uśpienia mikrokontrolera
  - 23 linie I/O dowolnego wykorzystania
- **Zakresy napięć zasilania:**
  - 2,7 V...5,5 V (ATmega 8L)
  - 4,5 V...5,5 V (ATmega8)
- **Zakres częstotliwości sygnału taktującego mikroprocesor:**
  - 0...8 MHz (ATmega 8L)
  - 0...16 MHz (ATmega8)
- **Pobierany prąd przy częstotliwości sygnału taktowania 4 MHz i przy napięciu zasilania 3V (ATmega8L):**
  - w stanie aktywnym: 3,6 mA
  - w trybie *Idle*: 1,0 mA
  - w trybie *Power-down*: 0,5 uA



*Pamięć stała miała pojemność 74 kB. Co istotne, pojęcie bajt **nie pojawia się w dokumentacji tego sprzętu** – AGC posługiwał się słowami maszynowymi mającymi długość 16 bitów. Pamięć kasowalna, czyli odpowiednik dzisiejszego RAM-u, miała pojemność 2048 16-bitowych słów, czyli 4 kB, a jednostka obliczeniowa Apollo Guidance Computer pracowała z częstotliwością 2,048 MHz.*



*Program Apollo  
Człowiek na Księżycu*

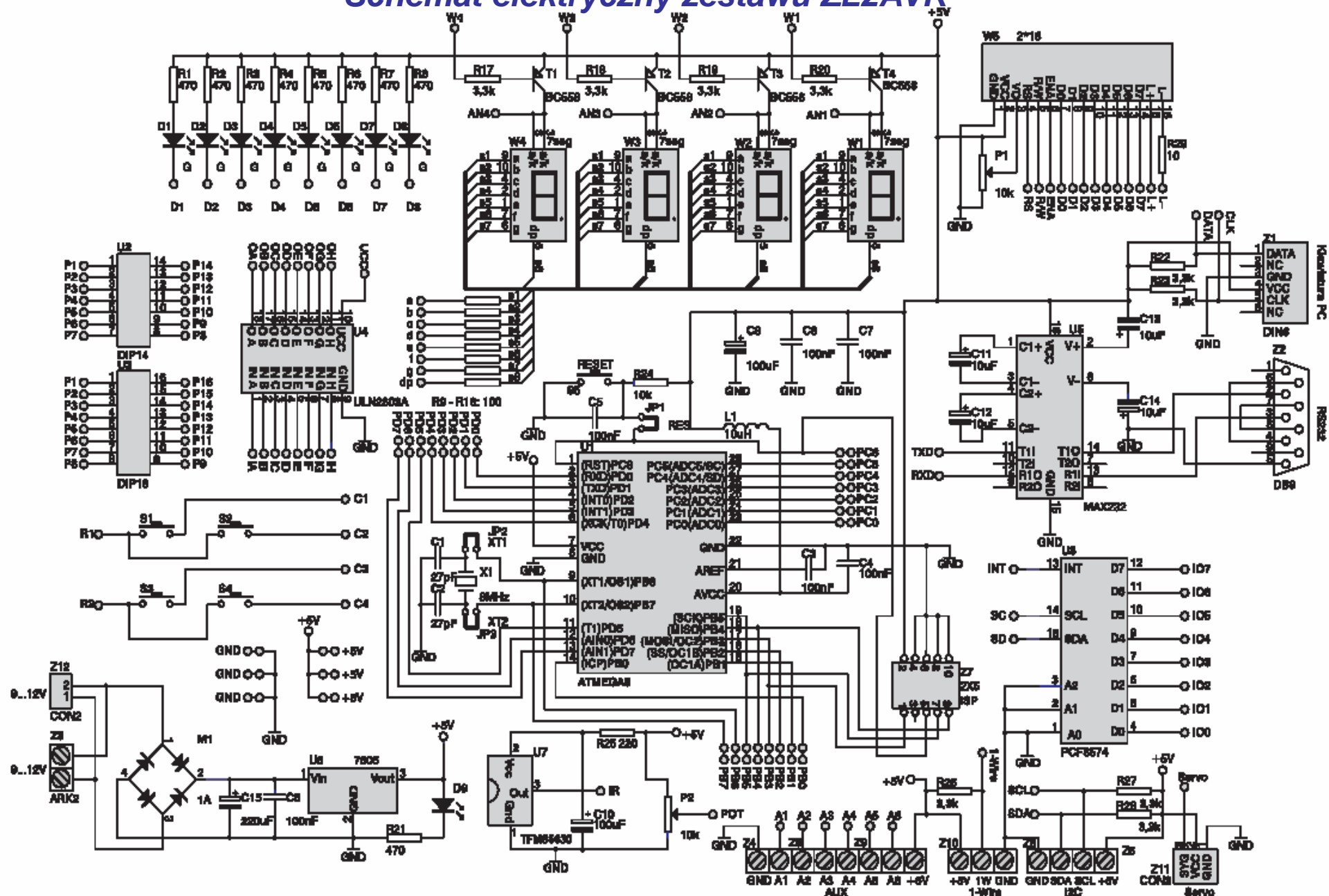


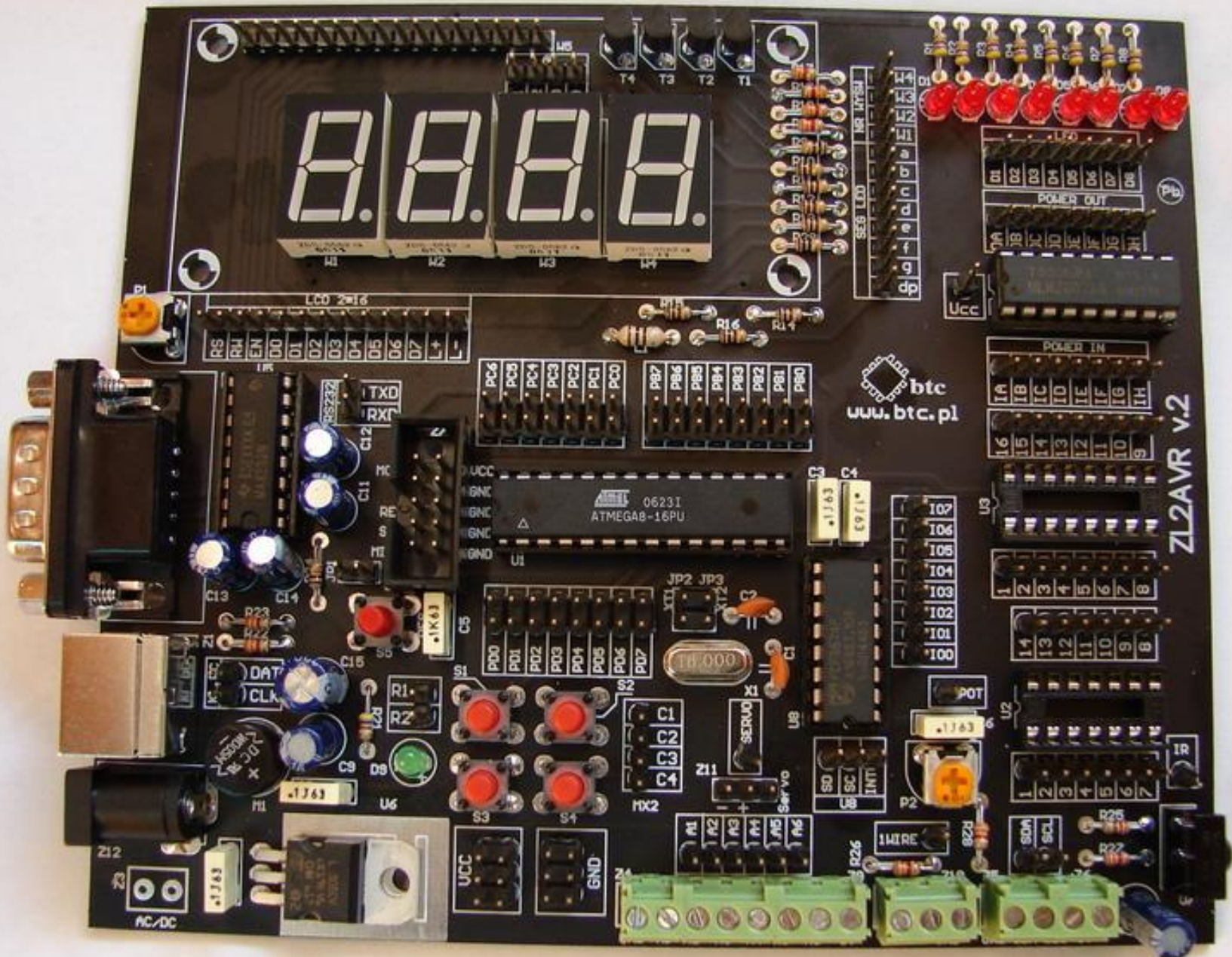


# **ZL2AVR**

***Zestaw uruchomieniowy dla  
mikrokontrolerów AVR ATmega8***

# Schemat elektryczny zestawu ZL2AVR





0000

btc  
www.btc.pl

ZL2AVR v.2

ATMEGA8-16PU

AC/DC

POWER IN

POWER OUT

LCD 2x16

RS RH EN DL D0 D1 D2 D3 D4 D5 D6 D7 L+ L-

PC6 PC5 PC4 PC3 PC2 PC1 PC0

PB7 PB6 PB5 PB4 PB3 PB2 PB1 PB0

IA IB IC ID IE IF IG IH

107 106 105 104 103 102 101 100

14 13 12 11 10 9 8

15 14 13 12 11 10 9 8

ML A2 A3 A4 A5 A6

S0A S0B

S1A S1B

S2A S2B

S3A S3B

S4A S4B

S5A S5B

S6A S6B

S7A S7B

S8A S8B

S9A S9B

S10A S10B

S11A S11B

S12A S12B

S13A S13B

S14A S14B

S15A S15B

S16A S16B

S17A S17B

S18A S18B

S19A S19B

S20A S20B

S21A S21B

S22A S22B

S23A S23B

S24A S24B

S25A S25B

S26A S26B

S27A S27B

S28A S28B

S29A S29B

S30A S30B

S31A S31B

S32A S32B

S33A S33B

S34A S34B

S35A S35B

S36A S36B

S37A S37B

S38A S38B

S39A S39B

S40A S40B

S41A S41B

S42A S42B

S43A S43B

S44A S44B

S45A S45B

S46A S46B

S47A S47B

S48A S48B

S49A S49B

S50A S50B

S51A S51B

S52A S52B

S53A S53B

S54A S54B

S55A S55B

S56A S56B

S57A S57B

S58A S58B

S59A S59B

S60A S60B

S61A S61B

S62A S62B

S63A S63B

S64A S64B

S65A S65B

S66A S66B

S67A S67B

S68A S68B

S69A S69B

S70A S70B

S71A S71B

S72A S72B

S73A S73B

S74A S74B

S75A S75B

S76A S76B

S77A S77B

S78A S78B

S79A S79B

S80A S80B

S81A S81B

S82A S82B

S83A S83B

S84A S84B

S85A S85B

S86A S86B

S87A S87B

S88A S88B

S89A S89B

S90A S90B

S91A S91B

S92A S92B

S93A S93B

S94A S94B

S95A S95B

S96A S96B

S97A S97B

S98A S98B

S99A S99B

S100A S100B

S101A S101B

S102A S102B

S103A S103B

S104A S104B

S105A S105B

S106A S106B

S107A S107B

S108A S108B

S109A S109B

S110A S110B

S111A S111B

S112A S112B

S113A S113B

S114A S114B

S115A S115B

S116A S116B

S117A S117B

S118A S118B

S119A S119B

S120A S120B

S121A S121B

S122A S122B

S123A S123B

S124A S124B

S125A S125B

S126A S126B

S127A S127B

S128A S128B

S129A S129B

S130A S130B

S131A S131B

S132A S132B

S133A S133B

S134A S134B

S135A S135B

S136A S136B

S137A S137B

S138A S138B

S139A S139B

S140A S140B

S141A S141B

S142A S142B

S143A S143B

S144A S144B

S145A S145B

S146A S146B

S147A S147B

S148A S148B

S149A S149B

S150A S150B

S151A S151B

S152A S152B

S153A S153B

S154A S154B

S155A S155B

S156A S156B

S157A S157B

S158A S158B

S159A S159B

S160A S160B

S161A S161B

S162A S162B

S163A S163B

S164A S164B

S165A S165B

S166A S166B

S167A S167B

S168A S168B

S169A S169B

S170A S170B

S171A S171B

S172A S172B

S173A S173B

S174A S174B

S175A S175B

S176A S176B

S177A S177B

S178A S178B

S179A S179B

S180A S180B

S181A S181B

S182A S182B

S183A S183B

S184A S184B

S185A S185B

S186A S186B

S187A S187B

S188A S188B

S189A S189B

S190A S190B

S191A S191B

S192A S192B

S193A S193B

S194A S194B

S195A S195B

S196A S196B

S197A S197B

S198A S198B

S199A S199B

S200A S200B

S201A S201B

S202A S202B

S203A S203B

S204A S204B

S205A S205B

S206A S206B

S207A S207B

S208A S208B

S209A S209B

S210A S210B

S211A S211B

S212A S212B

S213A S213B

S214A S214B

S215A S215B

S216A S216B

S217A S217B

S218A S218B

S219A S219B

S220A S220B

S221A S221B

S222A S222B

S223A S223B

S224A S224B

S225A S225B

S226A S226B

S227A S227B

S228A S228B

S229A S229B

S230A S230B

S231A S231B

S232A S232B

S233A S233B

S234A S234B

S235A S235B

S236A S236B

S237A S237B

S238A S238B

S239A S239B

S240A S240B

S241A S241B

S242A S242B

S243A S243B

S244A S244B

S245A S245B

S246A S246B

S247A S247B

S248A S248B

S249A S249B

S250A S250B

S251A S251B

S252A S252B

S253A S253B

S254A S254B

S255A S255B

S256A S256B

S257A S257B

S258A S258B

S259A S259B

S260A S260B

S261A S261B

S262A S262B

# **ZL2AVR - Zestaw uruchomieniowy dla mikrokontrolerów AVR ATmega8**

1. Mikrokontroler ATmega8 (U1), który może być taktowany wewnętrznym sygnałem zegarowym lub z zastosowaniem zewnętrznego rezonatora kwarcowego X1, dołączanego za pomocą zworek JP2, JP3.
2. S5, R24, C5, JP1 – zapewniające zerowanie mikrokontrolera. Zerowanie można przeprowadzić przyciskiem S5. Zworka JP1 umożliwia dołączenie zewnętrznego obwodu zerującego do mikrokontrolera.
3. Złącze Z7, które jest złączem programatora ISP.
4. Alfnumeryczny wyświetlacz LCD o organizacji 2x16 znaków (potencjometr P1 służy do ustawienia kontrastu wyświetlacza).
5. Cztery wyświetlacze 7-segmentowe LED ze wspólną anodą. Na płytce układu uruchomieniowego można wykorzystać jeden wyświetlacz LED sterowany statycznie lub wszystkie cztery przy sterowaniu multipleksowym.
6. Interfejs szeregowy RS232, który zrealizowano z wykorzystaniem konwertera poziomów U5.
7. Port wyjściowy dużej mocy zbudowany z wykorzystaniem układu ULN2803A (U4).
8. Blok ośmiu diod LED (D1...D8).
9. Blok przycisków (S1...S4).
10. Odbiornik transmisji danych w podczerwieni U7.



# **ZL2AVR - Zestaw uruchomieniowy dla mikrokontrolerów AVR ATmega8**

11. Konwerter I<sup>2</sup>C na 8-bitowy port I/O, który zrealizowano na układzie PCF8574. Adresy konwertera ustalono na stałe: adresem zapisu jest 64(dec), a odczytu 65(dec).
12. Złącza Z5 i Z6, które umożliwiają dołączanie zewnętrznych urządzeń sterowanych magistralą I<sup>2</sup>C.
13. Złącze Z10 (1-Wire) umożliwia dołączanie elementów sterowanych za pomocą interfejsu 1-Wire.
14. Złącze Z11 (Servo) umożliwia np. dołączenie serwomechanizmu modelarskiego.
15. Złącze Z1 umożliwia dołączenie klawiatury PS2.
16. Na złącza Z4, Z8 i Z9 (AUX) zostały wyprowadzone napięcia zasilające (masa i +5 V) oraz linie uniwersalne oznaczone A1...A6, które umożliwiają dołączanie elementów zewnętrznych, jak przyciski, przekaźniki itp.
17. Potencjometr P2 umożliwia zmienianie napięcia (w zakresie od 0 do 5 V) podawanego na wejście przetwornika A/C zawartego w mikrokontrolerze ATmega8.

# Programowanie mikrokontrolerów AVR w języku

# BASCOM





Pasek menu

Rozwijanie listy zdefiniowanych procedur

Rozwijanie listy etykiet w pisanim programie

Pasek narzędziowy

```
Dim A As Byte , B1 As Byte , C As Integer
A = 1
Print "print variable a " ; A
Print
Print "Text to print."
'new line
'constant to print

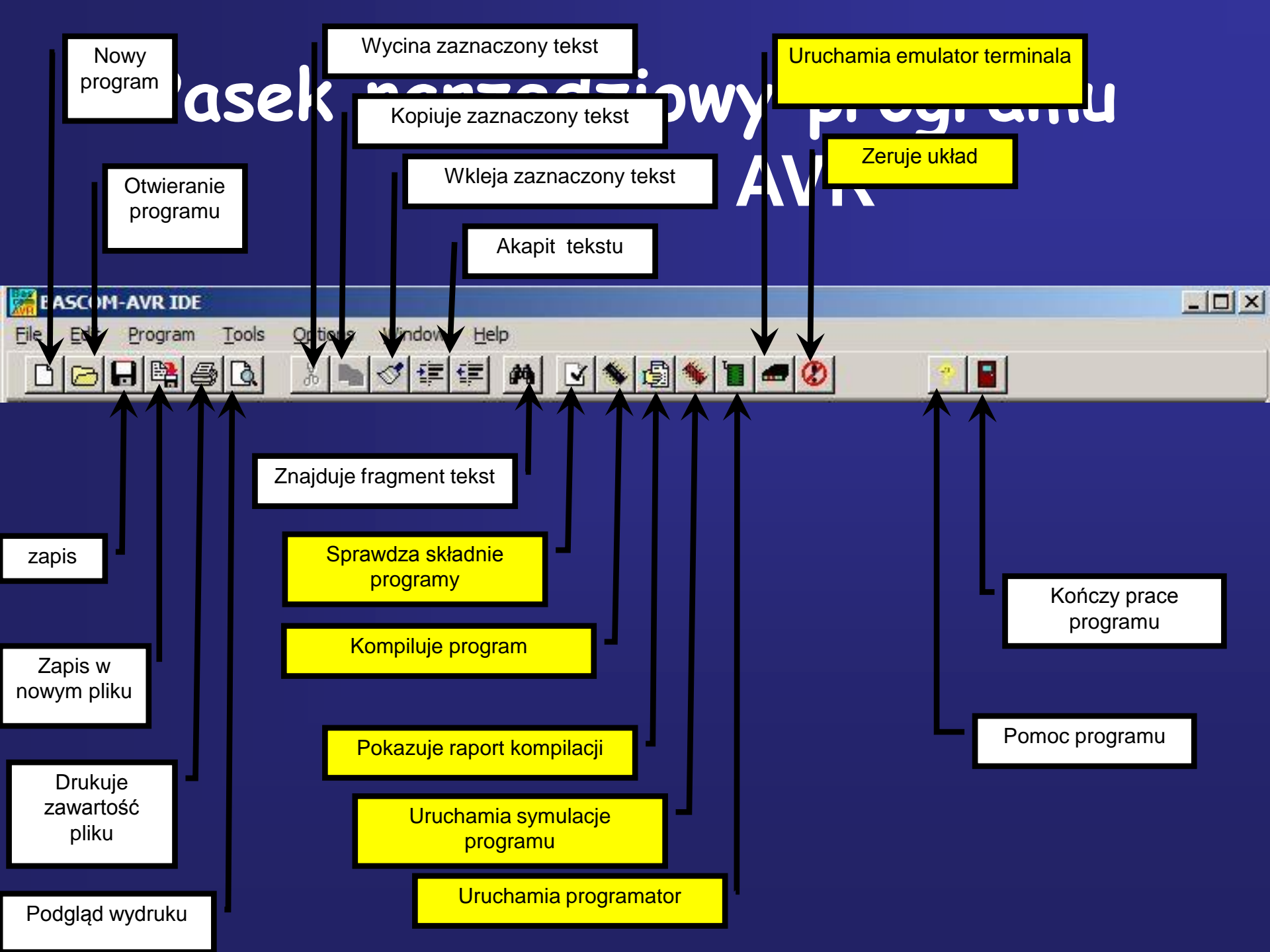
B1 = 10
Print Hex(b1)
C = &HA000
Print Hex(c)
Print C
'print in hexa notation
'assign value to c%
'print in hex notation
'print in decimal notation

C = -32000
Print C
Print Hex(c)
Rem Note That Integers Range From -32767 To 32768
End
```

Obszar roboczy okna edytora programu

Pasek statusu

I: 1 Insert



Nowy program

Otwieranie programu

Wycina zaznaczony tekst

Kopiuje zaznaczony tekst

Wkleja zaznaczony tekst

Akapit tekstu

Uruchamia emulator terminala

Zeruje układ



Znajduje fragment tekstu

Sprawdza składnię programu

Kompiluje program

Pokazuje raport kompilacji

Uruchamia symulację programu

Uruchamia programator

zapis

Zapis w nowym pliku

Drukuje zawartość pliku

Podgląd wydruku

Kończy prace programu

Pomoc programu



# Konfiguracja programu Bascom AVR

**BASCOM-AVR Options**

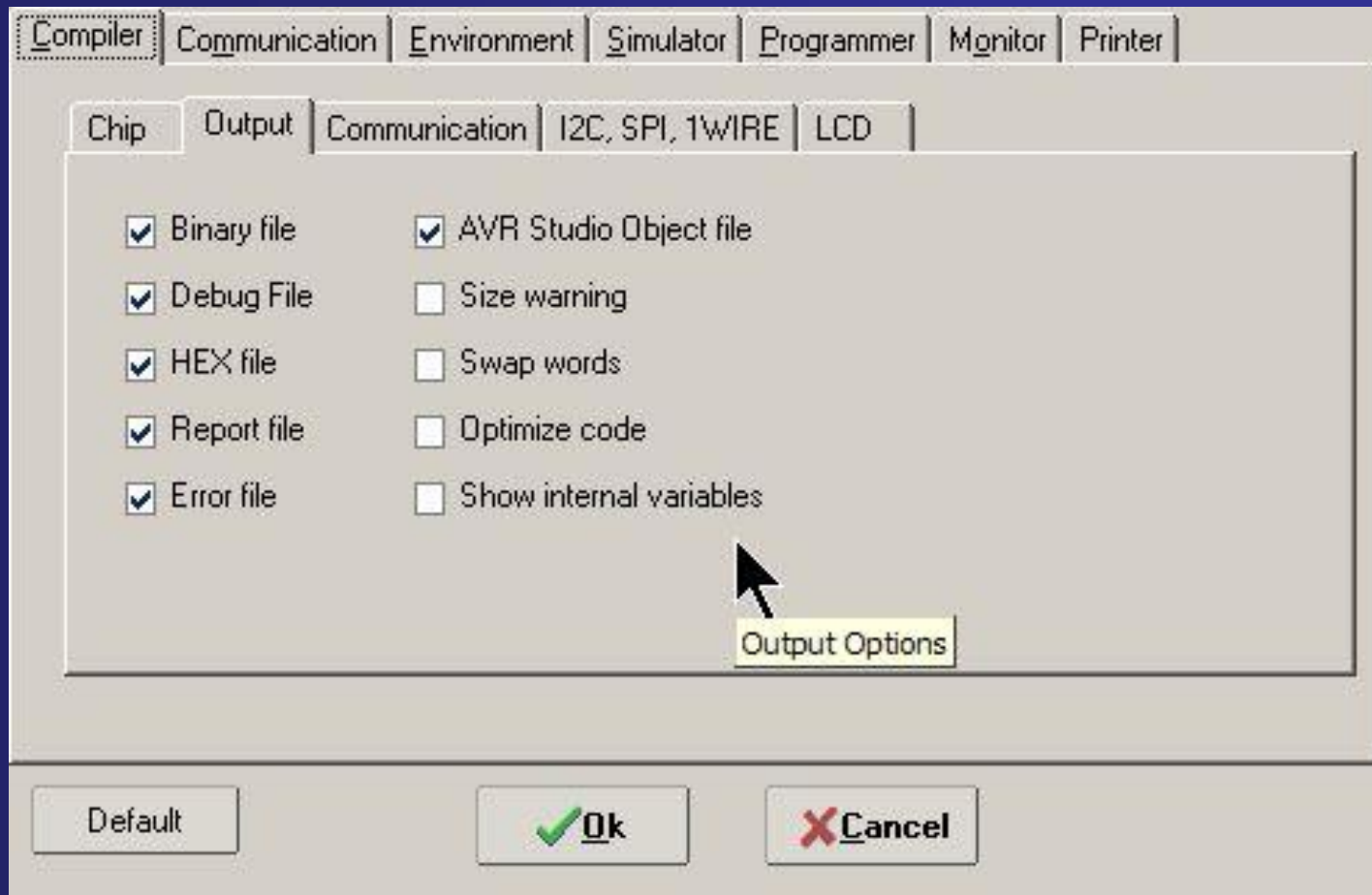
Compiler | Communication | Environment | Simulator | Programmer | Monitor | Printer

Chip | Output | Communication | I2C, SPI, 1WIRE | LCD

Chip	m8def.dat	FlashROM	8 KB
XRAM	None	SRAM	1024
HW Stack	32	EEPROM	512
Soft Stack	8	<input type="checkbox"/> XRAM waitstate	
Framesize	16	<input type="checkbox"/> External Access Enable	

Default       **Ok**       **Cancel**

# Konfiguracja programu Bascom AVR



# Konfiguracja programu Bascom AVR

The image shows a screenshot of the Bascom AVR configuration dialog box. The dialog has a menu bar at the top with the following options: **Compiler**, **Communication**, **Environment**, **Simulator**, **Programmer**, **Monitor**, and **Printer**. The **Communication** tab is selected. Below the menu bar, there are sub-tabs: **Chip**, **Output**, **Communication**, **I2C, SPI, 1WIRE**, and **LCD**. The **Communication** sub-tab is active. The main area contains three settings:

- Baudrate 0**: A dropdown menu showing the value **9600**.
- Frequency**: A dropdown menu showing the value **8000000** Hz.
- Error**: A text input field showing the value **0.16%**.

At the bottom of the dialog, there are three buttons: **Default**, **Ok** (with a green checkmark icon), and **Cancel** (with a red X icon).

# Konfiguracja programu Bascom AVR

Compiler | Communication | Environment | Simulator | Programmer | Monitor | Printer

Chip | Output | Communication | I2C, SPI, 1WIRE | LCD

I2C				
SCL port		PORTA.0	▼	
SDA port		PORTA.0	▼	
1 Wire				
1wire		PORTB.0	▼	
SPI				
Clock		PORTB.5	▼	
MOSI		PORTB.6	▼	
MISO		PORTB.7	▼	
SS		PORTB.5	▼	
		<input type="checkbox"/>	Use Hardware SPI	

Default  Ok  Cancel



# Konfiguracja programu Bascom AVR

Compiler | Communication | Environment | Simulator | Programmer | Monitor | Printer

Chip | Output | Communication | I2C, SPI, 1WIRE | LCD

LCD type: 16 \* 1a

BUS mode:  4-bit  8-bit

Data mode:  pin  bus

LCD-address: C000

RS-address: 8000

Make upper 3 bits 1 in LCD designer

Enable: PORTB.3

RS: PORTB.2

DB7: PORTB.7

DB6: PORTB.6

DB5: PORTB.5



DB4: PORTB.4

Default  Ok  Cancel

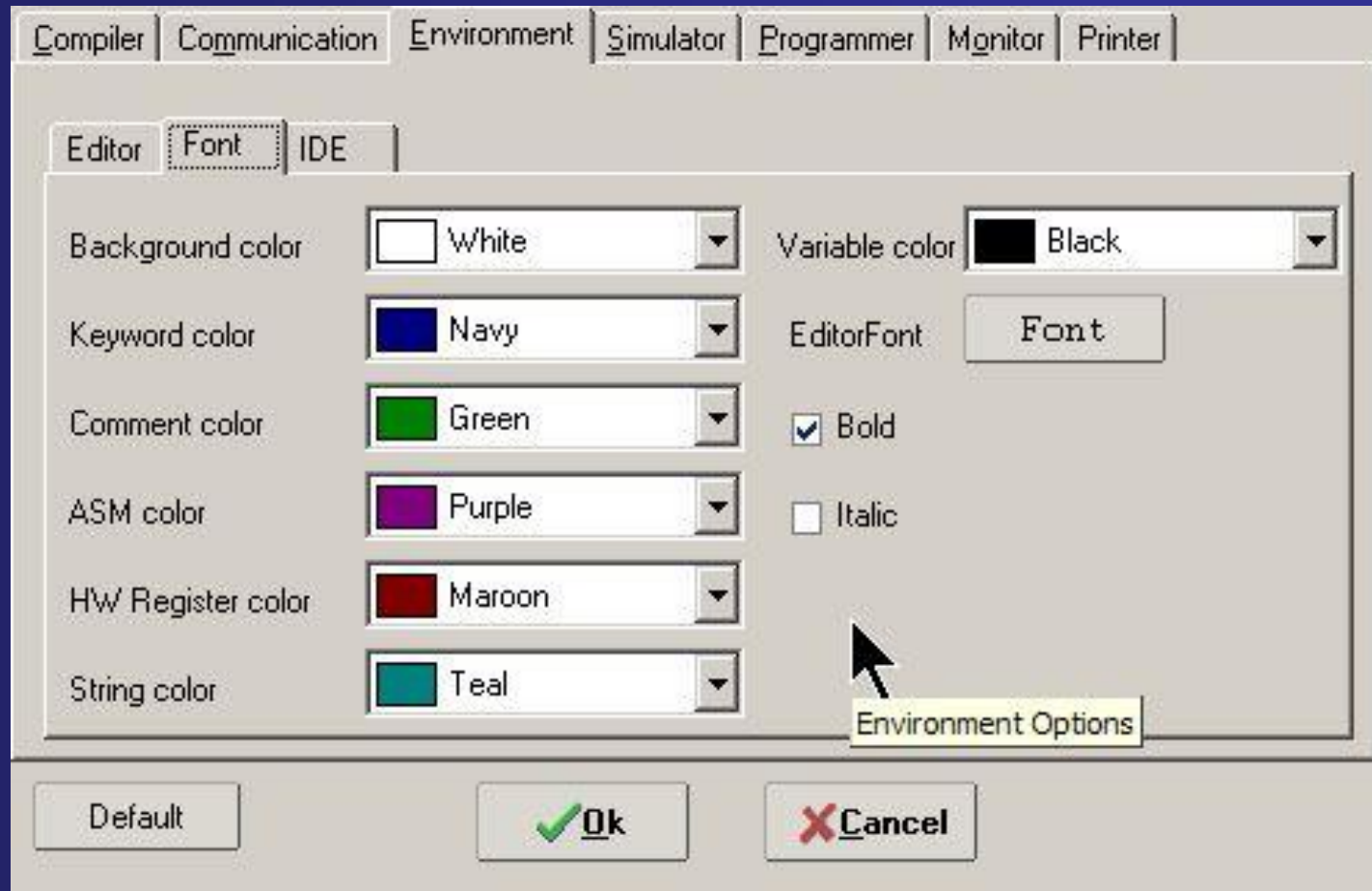
# Konfiguracja programu Bascom AVR

Compiler | Communication | Environment | Simulator | Programmer | Monitor | Printer

COM port	COM2	Handshake	None
Baudrate	19200	Emulation	TTY
Parity	None	<input type="checkbox"/> RTS	
Databits	8	Font	Font
Stopbits	1	BackColor	Navy

Default     **Ok**     **Cancel**

# Konfiguracja programu Bascom AVR



# Konfiguracja programu Bascom AVR

Compiler | Communication | Environment | **Simulator** | Programmer | Monitor | Printer

Use integrated Simulator  
 Run Simulator after compilation

Program  ..

Parameter

Default

# Konfiguracja programu Bascom AVR

Compiler | Communication | Environment | Simulator | Programmer | Monitor | Printer

Programmer:

Play sound:  ..

Erase warning    Auto Flash    AutoVerify    Upload Code and Data

Parallel | Serial | Other | Universal

LPT-address:  +

Port delay:

Default    **Ok**    **Cancel**

# Konfiguracja programu Bascom AVR

Parallel | Serial | Other | Universal

LPT-address 378 +

Port delay 10

Parallel | Serial | Other | Universal

COM-port 1

STK500 EXE

Parallel | Serial | Other | Universal

Program

Parameter  Use HEX file

Parallel | Serial | Other | Universal

Programmer stk200



# Konfiguracja programu Bascom AVR

Compiler | Communication | Environment | Simulator | Programmer | **Monitor** | Printer

Hex Mon

Upload speed

Monitor prefix  Prefix delay

Monitor suffix

Monitor delay

Default  **Ok**  **Cancel**

# Konfiguracja programu Bascom AVR

Compiler | Communication | Environment | Simulator | Programmer | Monitor | Printer

Font      Font

Printer      Setup



Black and White

Left Margin      25.4\_

Top Margin      25.4\_

Right Margin      25.4\_

Bottom Margin      25.4\_

Default       Ok       Cancel

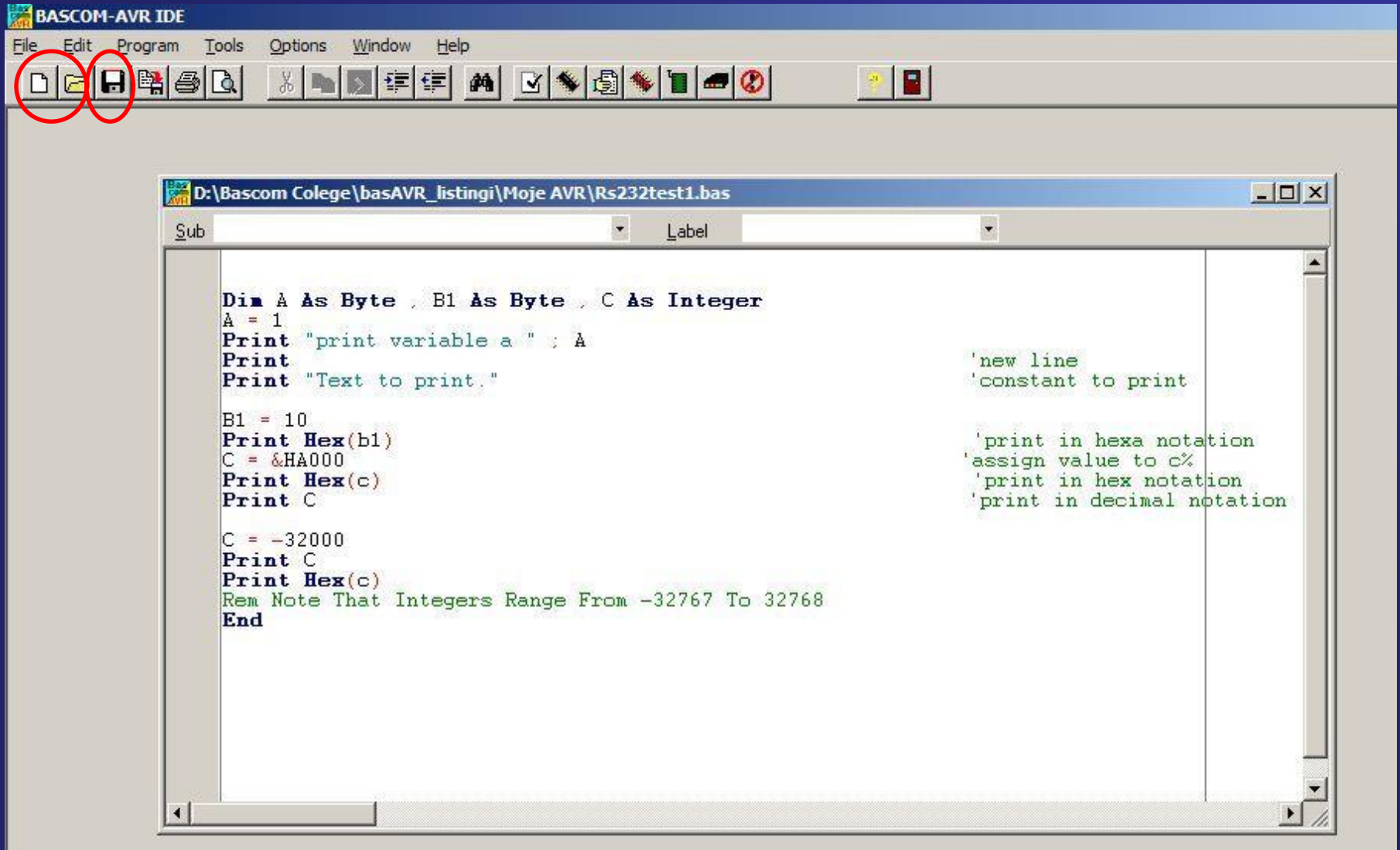
# Pierwszy program Bascom AVR

Proces tworzenia programu w języku Bascom można podzielić na kilka etapów:

1. Utworzenie nowego pliku programu lub otwarcie istniejącego
2. Napisanie programu lub wprowadzenie zmian do już istniejącego
3. Zapisanie programu
4. Przeprowadzenie kompilacji
5. Poprawienie błędów składniowych i ponowne przeprowadzenie kompilacji
6. Przeprowadzenie symulacji programowej lub sprzętowej
7. Poprawienie błędów logicznych w działaniu programu i powtórzenie kompilacji oraz symulacji
8. Zaprogramowanie mikrokontrolera i sprawdzenie jego działania w urządzeniu docelowym

# Pierwszy program

1. Utworzenie nowego pliku programu lub otwarcie istniejącego
2. Zapisanie programu na dysku !!!!



The screenshot shows the BASCOM-AVR IDE interface. The menu bar includes File, Edit, Program, Tools, Options, Window, and Help. The toolbar contains various icons, with the 'New File' and 'Open File' icons circled in red. The main window displays the following code:

```
Sub
Label

Dim A As Byte , B1 As Byte , C As Integer
A = 1
Print "print variable a " ; A
Print
Print "Text to print."
Print "new line"
Print "constant to print"

B1 = 10
Print Hex(b1)
Print "print in hexa notation"
C = &HA000
Print Hex(c)
Print "assign value to c%"
Print C
Print "print in hex notation"
Print "print in decimal notation"

C = -32000
Print C
Print Hex(c)
Rem Note That Integers Range From -32767 To 32768
End
```

Sprawdzenie składni  
napisanego programu

# Błędy

Przed kompilacją należy sprawdzić czy program nie zawiera błędów składniowych  
*Syntax check* klawisz Ctrl+F7

```
Sub
Label
If B1 < 1 Then
B1 = 10
Else
B1 = 11
Printt Hex(b1)
C = @HA000
Print Hex(c)
Print C

C = -32000
Print C
Print Hex(c)
Rem Note That Integers Range From -32767 To 32768
End
```

Error: 1 Line: 12 Unknown statement [PRINTT HEX(B1)] , in File : D:\BASCOM COLEGE\BASAVR\_LISTINGI\MOJE AVR\RS232TEST1.BAS  
Error: 7 Line: 0 IF THEN expected [ 21] , in File :  
Error: 123 Line: 0 END IF expected , in File :

Przy braku tylko jednej instrukcji End If bascom sygnalizuje dwa błędy o kodach: 7 (spodziewana instrukcja If....End) oraz 123 (brak instrukcji End If)

## Poniższa tabela zawiera listę błędów mogących się pojawić podczas sprawdzania składni lub kompilacji.

Kod błędu	Opis		
1	Nieznana instrukcja	35	Spodziewano się 3 parametrów
2	Nieznana struktura instrukcji EXIT	36	Spodziewano się THEN
3	Spodziewano się WHILE	37	Błędny operator relacji
4	Brak miejsca w pamięci IRAM na zmienną typu Bit	38	Nie można wykonać tej operacji dla zmiennych bitowych
5	Brak miejsca na zmienne typu Bit	39	Spodziewano się FOR
6	Spodziewana . (kropka) w nazwie pliku.	40	Ta zmienna nie może być parametrem instrukcji RESET
7	Spodziewana instrukcja IF..THEN	41	Ta zmienna nie może być parametrem instrukcji SET
8	Pliku źródłowego nie odnaleziono	42	Spodziewano się liczby jako parametru
9	Maksymalnie można użyć 128 instrukcji ALIAS	43	Pliku nie odnaleziono
10	Nieznany typ wyświetlacza	44	Spodziewano się 2 zmiennych
11	Spodziewano się INPUT, OUTPUT, 0 lub 1	45	Spodziewano się DO
12	Nieznany parametr instrukcji CONFIG	46	Błędne przypisanie
13	Ta stała już jest zdefiniowana	47	Spodziewano się UNTIL
14	Bajty mogą być tylko w IRAM	50	Liczba nie mieści się w zmiennej Integer
15	Błędny typ danych	51	Liczba nie mieści się w zmiennej Word
16	Nieznana definicja	52	Liczba nie mieści się w zmiennej Long
17	Spodziewano się 9 parametrów	60	Ta etykieta już istnieje
18	Zmienne bitowe umieszczone mogą być tylko w pamięci SRAM lub IRAM	61	Etykiety nie znaleziono
19	Spodziewano się określenia długości zmiennej typu String	62	Najpierw SUB lub FUNCTION
20	Nieznany typ danych	63	Parametrem funkcji ABS() może być liczba typu Integer lub Long
21	Brak wolnej pamięci IRAM	64	Spodziewany , (przecinek)
22	Brak wolnej pamięci SRAM	65	Urządzenie nie zostało otwarte
23	Brak wolnej pamięci XRAM	66	Urządzenie już jest otwarte
24	Brak wolnej pamięci EEPROM	68	Spodziewano się numeru kanału
25	Ta zmienna już jest zdefiniowana	70	Ta szybkość transmisji nie może być użyta
26	Spodziewano się AS	71	Typ przekazanych parametrów nie jest zgodny z zadeklarowanym
27	Spodziewano się parametru	72	Getclass error. Jest to błąd wewnętrzny.
28	Spodziewano się IF..THEN	73	Używanie PRINT w połączeniu z tą funkcją jeszcze nie działa
29	Spodziewano się SELECT..CASE	74	Spodziewano się 3 parametrów
30	Zmienne bitowe są zmiennymi globalnymi, nie można ich usuwać	80	Kod nie mieści się w pamięci tego układu
31	Błędny typ danych	81	Użyj funkcji HEX() zamiast PRINTHEX
32	Niezdefiniowana zmienna	82	Użyj funkcji HEX() zamiast LCDHEX
33	Zmienne globalne nie mogą być usuwane	85	Nieznane źródło przerwania
34	Błędna ilość parametrów	86	Błędny parametr w instrukcji CONFIG TIMER
35	Spodziewano się 3 parametrów	87	Nazwa podana jako parametr instrukcji ALIAS już jest używana
		88	Spodziewano się 0 lub 1
		89	Liczba musi zawierać się w przedziale 1-4



...  
242 Wystąpił brak zgodności zmiennych  
243 Numer bitu wykracza poza liczbę dopuszczalną dla tej zmiennej  
244 Nie możesz używać wskaźnika Y  
245 Zmienne tablicowe nie mogą być w pamięci IRAM  
246 Brak miejsca na definicje w pliku .DEF  
247 Spodziewano się kropki  
248 Powinien być użyty argument BYVAL w tej deklaracji  
249 Procedura obsługi przerwania jest już zdefiniowana  
250 Spodziewano się GOSUB  
251 Ta etykieta musi być nazwana SECTIC  
252 Spodziewano się zmiennej Integer lub Word  
253 Ta zmienna nie może być w pamięci ERAM  
254 Spodziewana zmienna  
255 Spodziewano się Z lub Z+  
256 Spodziewano się zmiennej typu Single  
257 Spodziewano się ""  
258 Spodziewano się SRAM  
259 Zmienne typu Byte nie mogą przyjmować wartości ujemnych  
260 Ciąg znaków nie zmieści się w tej zmiennej typu String  
261 Spodziewano się tablicy  
262 Spodziewano się ON lub OFF  
263 Indeks tablicy poza zakresem  
264 Zamiast tego użyj ECHO OFF i ECHO ON  
265 Spodziewano się offsetu w rozkazie LDD lub STD. Np. Z+1  
266 Spodziewano się TIMER0, TIMER1 lub TIMER2  
267 Spodziewano się stałej liczbowej  
268 Paramter musi zawierać się w granicach 0 - 3  
269 Spodziewano się END SELECT  
270 Ten adres już jest zajęty  
322 Ten typ danych nie jest obsługiwany przez tą instrukcję  
232 Etykieta posiada zbyt dużo znaków  
234 Ten układ nie jest obsługiwany przez bibliotekę I2C w trybie Slave  
325 Stopień podziału preskalera musi wynosić: 1, 8, 32, 128, 256 lub 1024  
326 Spodziewano się #ENDIF  
327 Maksymalna wielkość to 255  
328 Nie działa z programowym układem UART  
999 Wersje DEMO lub BETA generują kod tylko do 2 KB

**Uwaga!** Często zdarza się że kompilator raportuje błąd "File not found", który jest zwykle spowodowany przez błędne określenie parametrów instrukcji - zwłaszcza **CONFIG**.

**Kompilacja**

The screenshot shows the BASCOM-AVR IDE interface. The main window displays the following assembly code:

```
Dim A As Byte , B1 As Byte , C As Integer
A = 1
Print "print variable a " ; A
Print
Print "Text to print."
'new line
'constant to print

B1 = 10
Print Hex(b1)
'print in hexa notation
C = &HA000
'assign value to c%
Print Hex(c)
'print in hex notation
Print C
'print in decimal notation

C = -32000
Print C
Print Hex(c)
Rem Note That Integers Range From -32767 To 32768
End
```

An arrow points from the text below to the compilation button (a lightning bolt icon) in the toolbar.

1: 1 | Insert

Taskbar: Start, Microsoft Pow..., Magnesy - M..., XnView - [Bro..., Pawel Kowalc..., BASCOM-AVR, PL, 18:33

# Kompilacja programu

lub klawisz F7

# Kompilacja

**Podczas kompilacji, w zależności od ustawień w oknie konfiguracyjnym kompilatora, utworzone zostaną następujące pliki:**

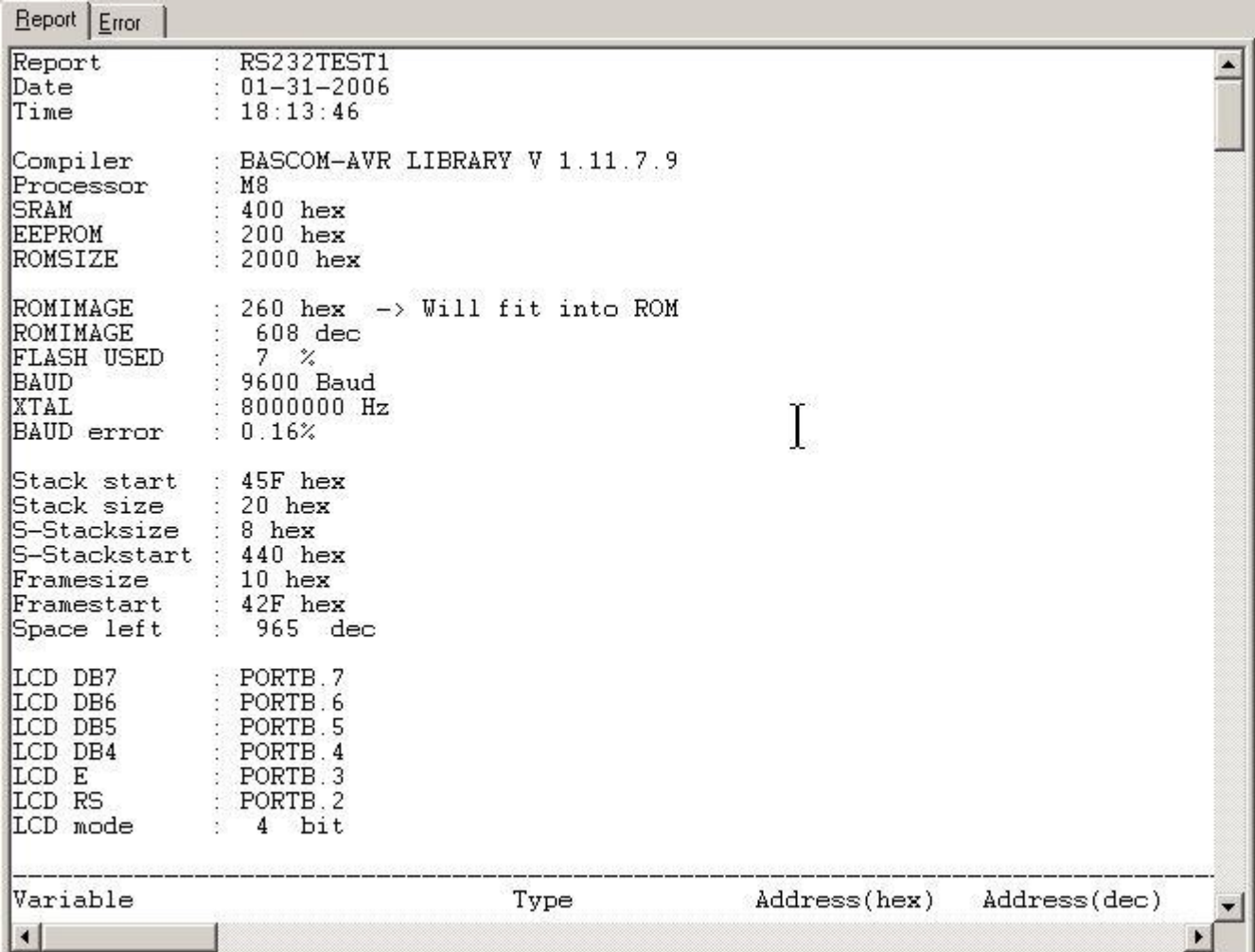
- xxx.BIN - Plik binarny, który można wpisać do mikroprocesora.
- xxx.DBG - plik wykorzystywany podczas symulacji.
- xxx.OBJ - Plik wykorzystywany przez symulator AVR Studio ale także przez wewnętrzny symulator Bascoma.
- xxx.HEX - plik hexadecimal wykorzystywany przez niektóre programy obsługujące programatory, plik ten może być także wpisany do pamięci programu mikrokontrolera przez bootloader.
- xxx.ERR - plik z komunikatem o błędach tworzonych po znalezieniu błędów składniowych.
- xxx.RPT - plik raportu z kompilacji.
- xxx.EEP - plik obrazu pamięci EEPROM.

---

**Uwaga:** pliki te powstają w tym samym katalogu gdzie został zapisany program

# Informacja o kompilacji

Wybierając z menu *Program* polecenie *Show result* lub klawiszem **Ctrl + W** możemy obejrzeć informacje dotyczące przebiegu kompilacji:



The screenshot shows a window titled "Report" with a tab labeled "Error". The main area contains a text-based report of compilation parameters and settings. On the right side of the window, there are three buttons: "Ok" (with a green checkmark), "Print" (with a printer icon), and "Copy" (with a document icon). At the bottom of the window, there is a table header with columns for "Variable", "Type", "Address(hex)", and "Address(dec)".

```
Report : RS232TEST1
Date   : 01-31-2006
Time   : 18:13:46

Compiler : BASCOM-AVR LIBRARY V 1.11.7.9
Processor : M8
SRAM    : 400 hex
EEPROM  : 200 hex
ROMSIZE : 2000 hex

ROMIMAGE : 260 hex -> Will fit into ROM
ROMIMAGE : 608 dec
FLASH USED : 7 %
BAUD      : 9600 Baud
XTAL      : 8000000 Hz
BAUD error : 0.16%

Stack start : 45F hex
Stack size  : 20 hex
S-Stacksize : 8 hex
S-Stackstart : 440 hex
Framesize   : 10 hex
Framestart  : 42F hex
Space left  : 965 dec

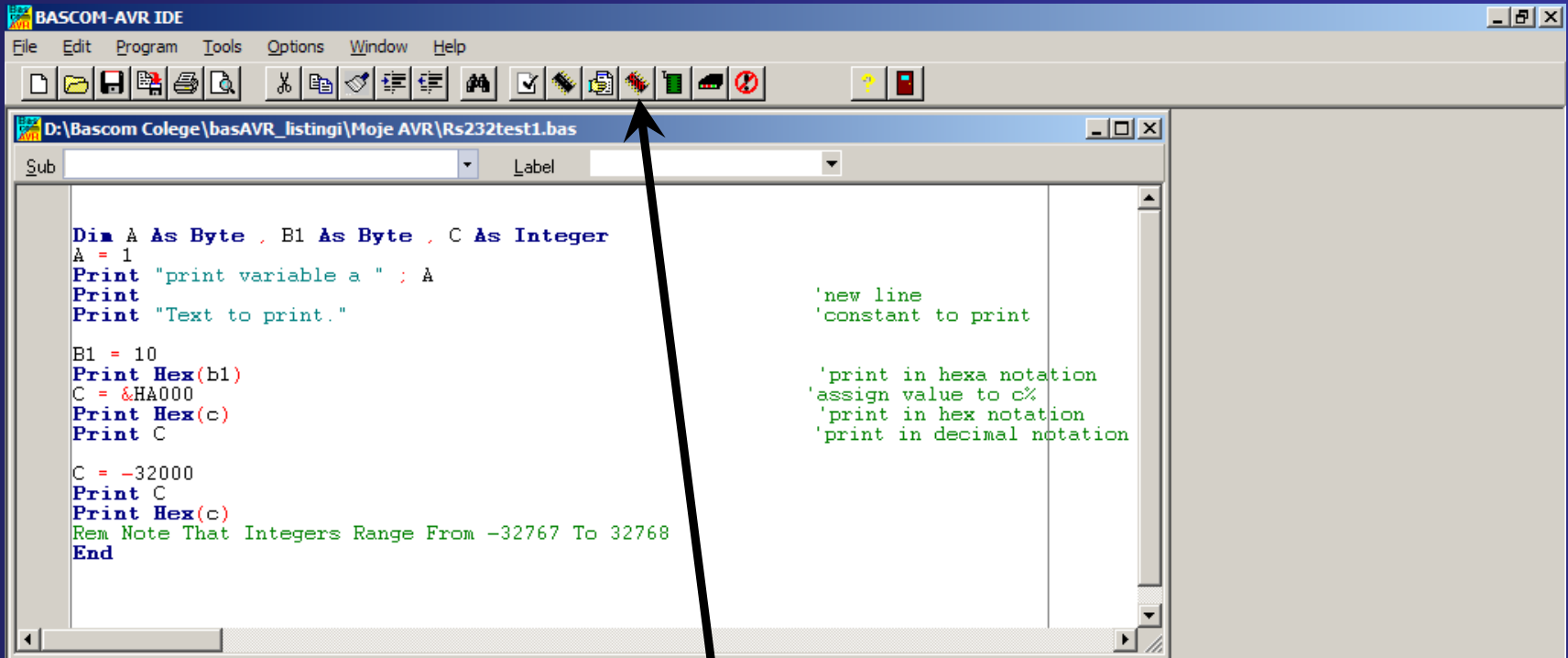
LCD DB7 : PORTB.7
LCD DB6 : PORTB.6
LCD DB5 : PORTB.5
LCD DB4 : PORTB.4
LCD E    : PORTB.3
LCD RS   : PORTB.2
LCD mode : 4 bit
```

Variable	Type	Address(hex)	Address(dec)
----------	------	--------------	--------------

**Symulacja programowa**



# Symulacja programowa



The screenshot shows the BASCOM-AVR IDE interface. The main window displays the following assembly code:

```
Dim A As Byte , B1 As Byte , C As Integer
A = 1
Print "print variable a " ; A
Print
Print "Text to print."

B1 = 10
Print Hex(b1)
C = &HA000
Print Hex(c)
Print C

C = -32000
Print C
Print Hex(c)
Rem Note That Integers Range From -32767 To 32768
End
```

The toolbar at the top contains various icons, including a simulation icon (a red and black chip) which is pointed to by a black arrow. The status bar at the bottom shows the current line as 1: 1 and the keyboard mode as Insert.

**Symulacja programowa**

**lub klawisz F2**

# Symulacja programowa

Wybierając z menu *Program* polecenie *Simulate* lub klawiszem F2 możemy przeprowadzić symulację działania napisanego programu bez konieczności programowania mikrokontrolera:

The screenshot shows a software interface for simulating a microcontroller program. At the top, there is a toolbar with various icons and a menu bar. Below the menu bar is a table for monitoring variables. The main area is divided into a code editor and a terminal window. The code editor contains a BASIC-style program with comments. The terminal window shows the output of the program. Annotations with arrows point to specific parts of the interface.

Variable	Value	Hex	Bin

```
1 |
2
3 Dim A As Byte , B1 As Byte , C As Integer
4 A = 1
5 Print "print variable a " ; A
6 Print
7 Print "Text to print."
8 If B1 < 1 Then
9 B1 = 10
10 Else
11 B1 = 11
12 End If
13 Print Hex(b1)
14 C = &HA000
15 Print Hex(c)
16 Print C
```

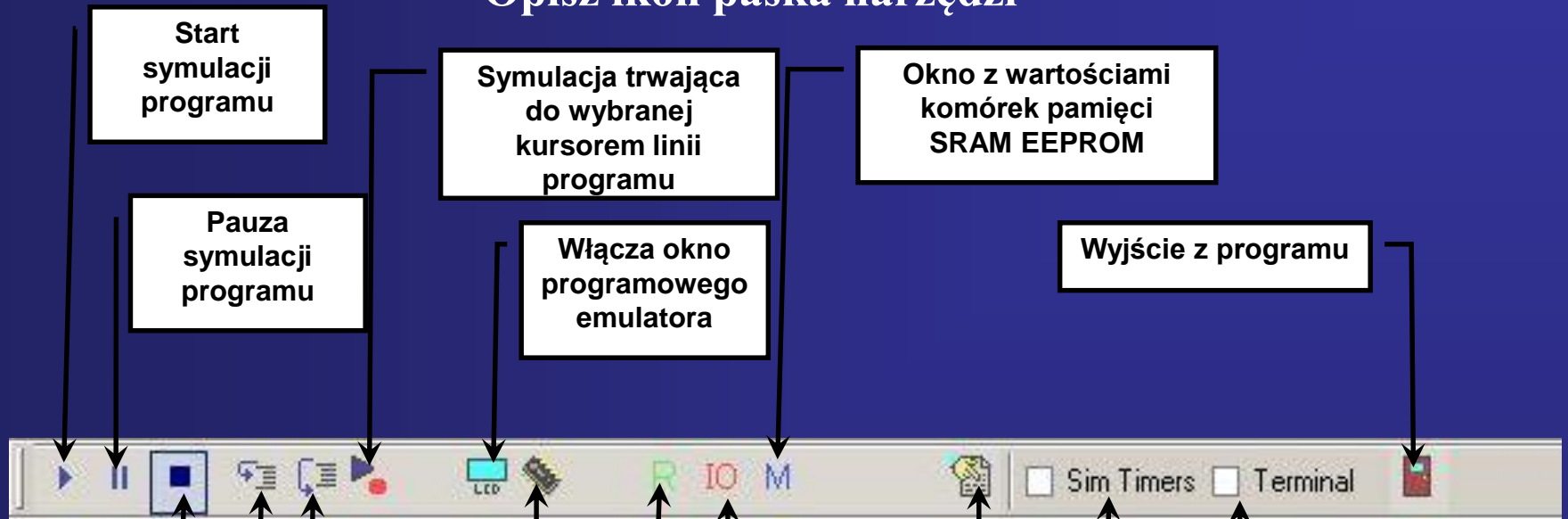
PC = 0    Cycles = 0    Stopped

Annotations:

- Pasek narzędzi (Toolbar)
- Okno emulatora terminala (Terminal window)
- Okno symulowanego programu (Code editor)

# Symulacja programowa

## Opisz ikon paska narzędzi



Start symulacji programu

Pauza symulacji programu

Symulacja trwająca do wybranej kursorem linii programu

Włącza okno programowego emulatora

Okno z wartościami komórek pamięci SRAM EEPROM

Wyjście z programu

Stop symulacji programu

Włą/wył sprzętową symulację

Włą/wył odświeżanie wartości zmiennych podczas symulacji

Praca krokowa symulacji

Okno z wartościami rejestrów R0..R31

Włącza sprawdzanie czy symulowane są Timery

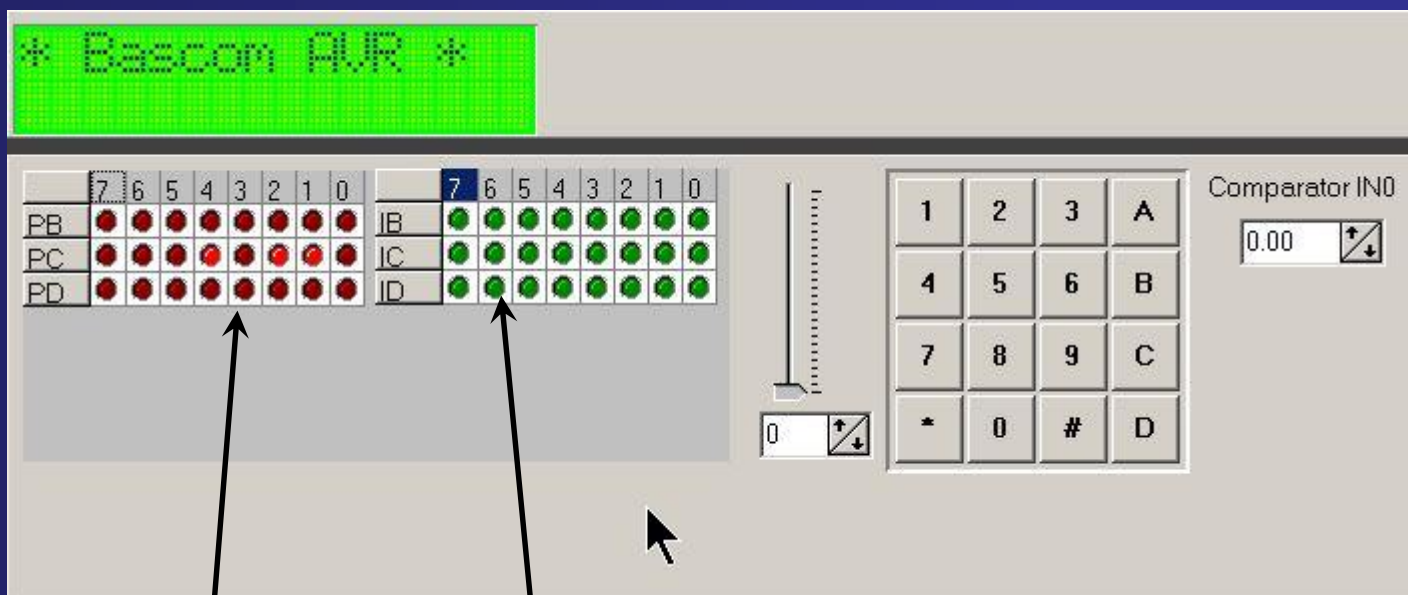
Praca krokowa symulacji z ominięciem linii podprogramów

Okno z wartościami rejestrów I/O mikrokontrolera

Włącza emulator terminala podczas symulacji

# Symulacja programowa

Okno symulacji programowej z wirtualnymi peryferiami mikrokontrolera

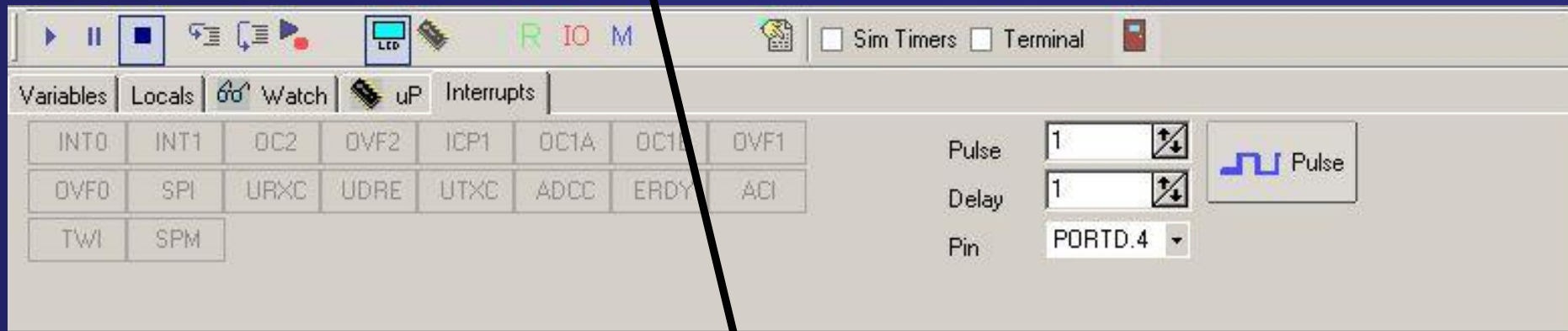
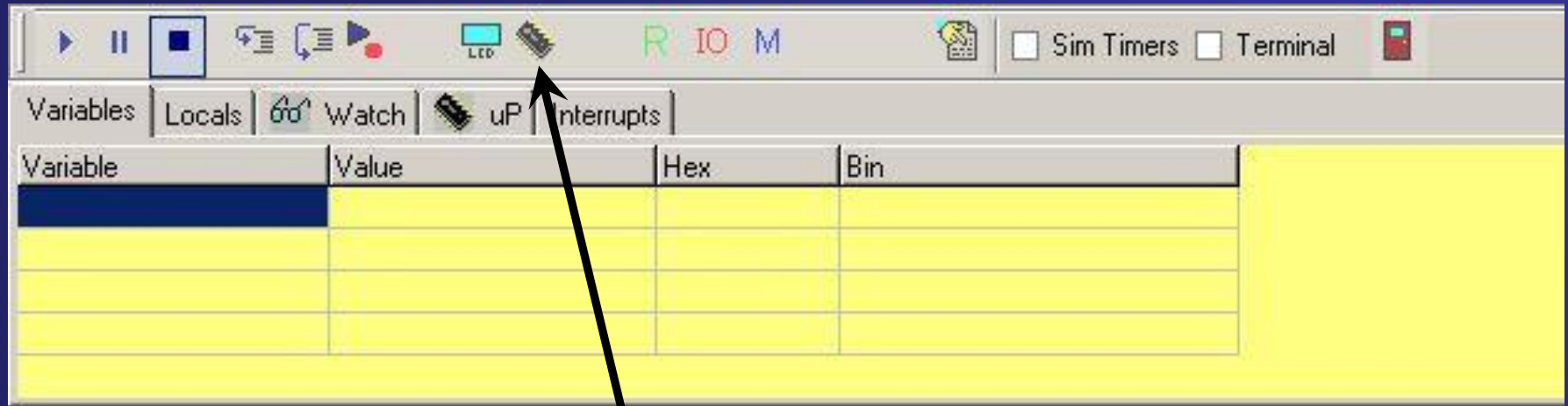


Czerwone diody dotyczą stanów linii portów wyjściowych

Zielone diody dotyczą stanów linii portów wejściowych

# Symulacja sprzętowa

Okno symulacji programowej z wirtualnymi peryferiami mikrokontrolera



Symulacja sprzętowa

# Symulacja sprzętowa

Do pamięci mikrokontrolera powinien być wpisany program będący rodzajem monitora, który komunikuje się z symulatorem Bascoma poprzez interfejs RS232

```
C:\Program Files\MCS Electronics\BASCOM-AVR\SAMPLES\basmon.bas
Sub
Label

'----- MCS Electronics -----
'          BASMON.BAS
'regfile must match the chip you use in your target system
$regfile = "m8def.dat"
'crystal must match too
$crystal = 8000000
'baudrate must match the baudrate of the terminal emulator
$bbaud = 19200

'[variables]
Dim Krk As Byte
Dim Address As Word
Dim Adrl As Byte , Adrh As Byte
Dim V1 As Byte
'[main program]
Print "BASMON Version 1.01"

Do
  Krk = Waitkey()
  If Krk = "T" Then
    Print Chr(13);
  ElseIf Krk = "W" Then
    Address = Waitkey()
    V1 = Waitkey()
    Out Address , V1
    Print Chr(13);
  ElseIf Krk = "R" Then
    Address = Waitkey()
    V1 = Inp(address)
    Print Chr(v1);
  ElseIf Krk = "O" Then
    Adrl = Waitkey()
    Adrh = Waitkey()
    V1 = Waitkey()
    Address = Adrh * 256
    Address = Address + Adrl
    Out Address , V1
    Print Chr(13);
  ElseIf Krk = "?" Then
    Print "?";
  End If
Loop

'command
'address
'for new OUT command
'value

'check if it is attached
'it is working

'wait for address
'wait for value
'write value
'confirm

'wait for address
'get value
'write back value

'wait for LSB of address
'wait for MSB
'wait for data

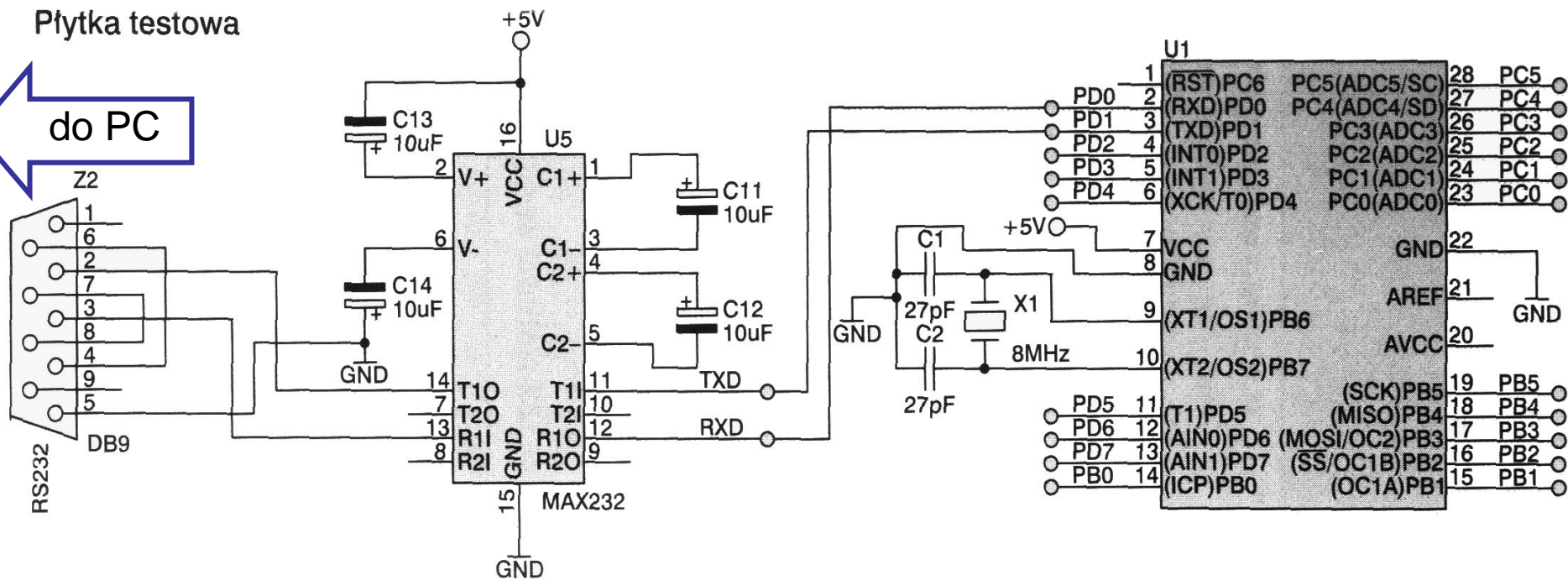
'just echo for test

'for ever
```



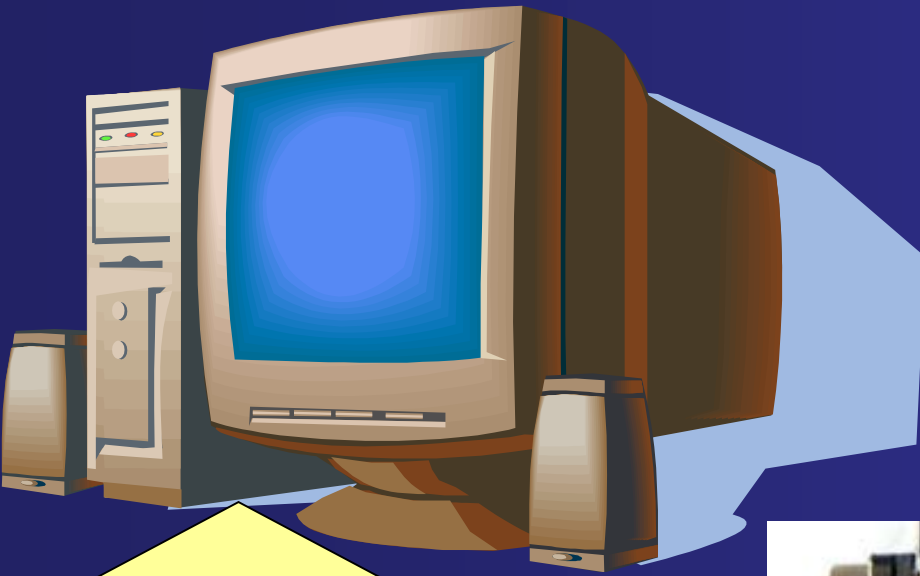
# Symulacja sprzętowa

Schemat dołączenia mikrokontrolera przez konwerter napięć dla standardu RS232 do komputera PC

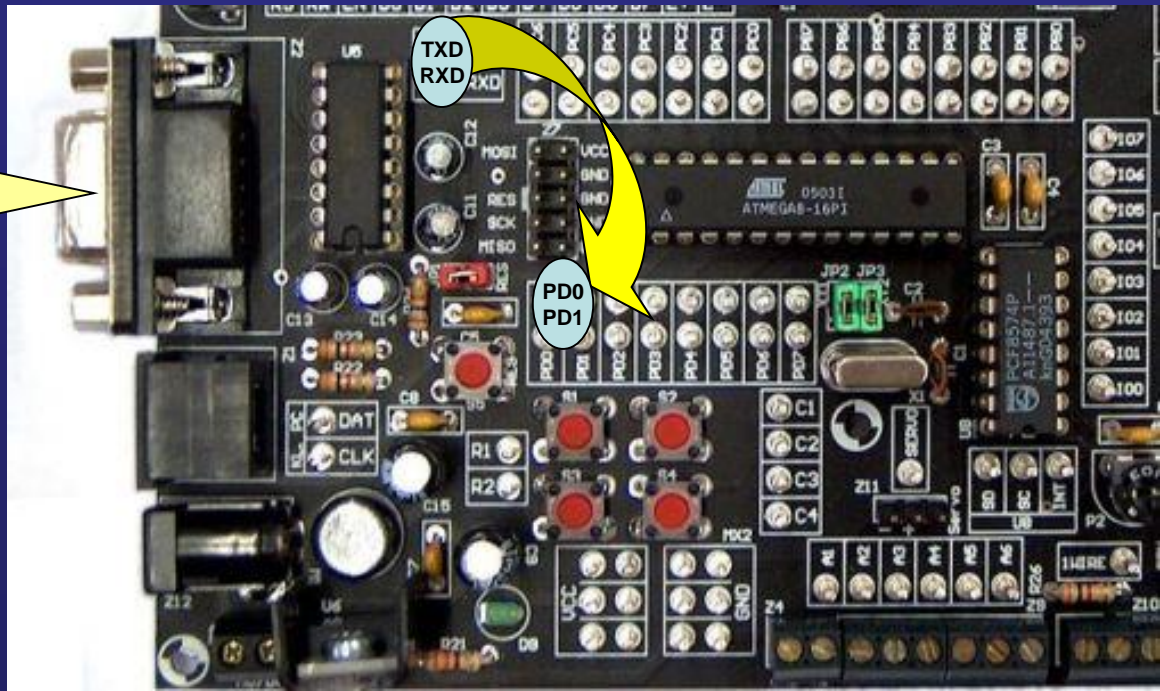


# Symulacja sprzętowa

## mikrokontrolera



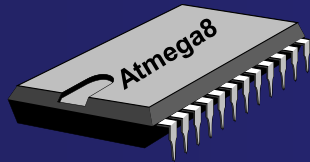
RS232



# Programowanie

## mikrokontrolera

# Programowanie mikrokontrolera



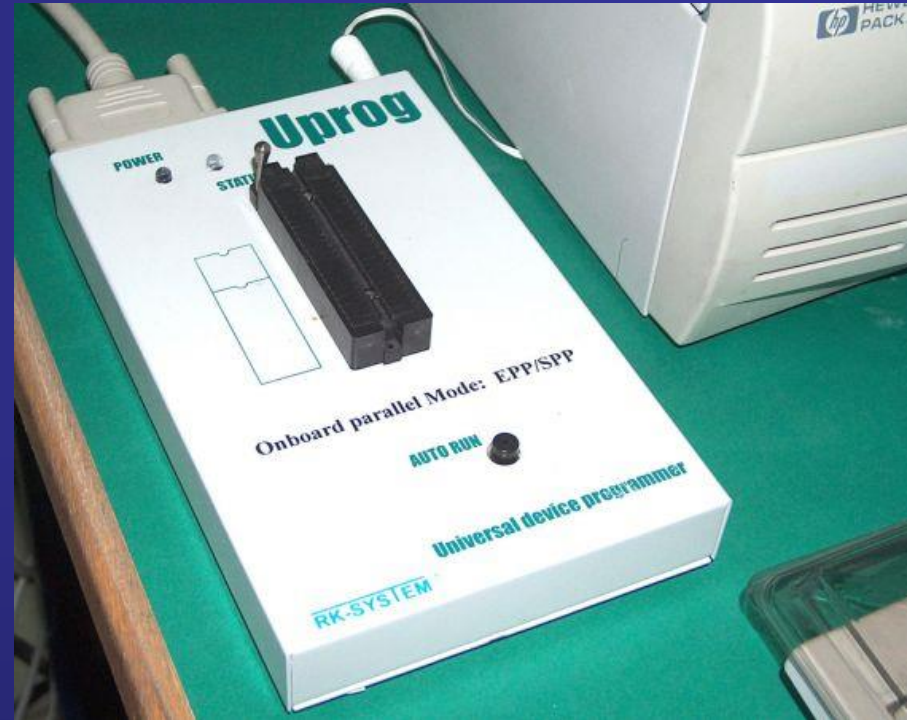
Wpisanie programu  
do pamięci Flash ROM

```
Atmega8

Progfile = wddat.dat
Scriptal = ROMM000
Shaud = 12000

"variables"
Dim i& As Byte
Dim adr As Word
Dim iadr As Byte : iadr As Byte
Dim i As Byte

"main program"
Print "SERVUS Version 1.01"
Do
  i& = Inkey()
  If i& = "" Then
    Print Chr(1)
  Elseif i& = "0" Then
    i& = Waitkey()
    i = i + 1
    Print Chr(1)
  Elseif i& = "1" Then
    i& = Waitkey()
    i = i + 1
    Print Chr(1)
  Elseif i& = "2" Then
    i& = Waitkey()
    i& = i + 1
    Print Chr(1)
  Elseif i& = "3" Then
    i& = Waitkey()
    i& = i + 1
    Print Chr(1)
  Elseif i& = "4" Then
    Print " "
  End If
Loop
```



Programator  
równoległy  
Uprog

# Programowanie

mikrokontrolera



**Uwaga:**

**Ładunki elektrostatyczne!!**





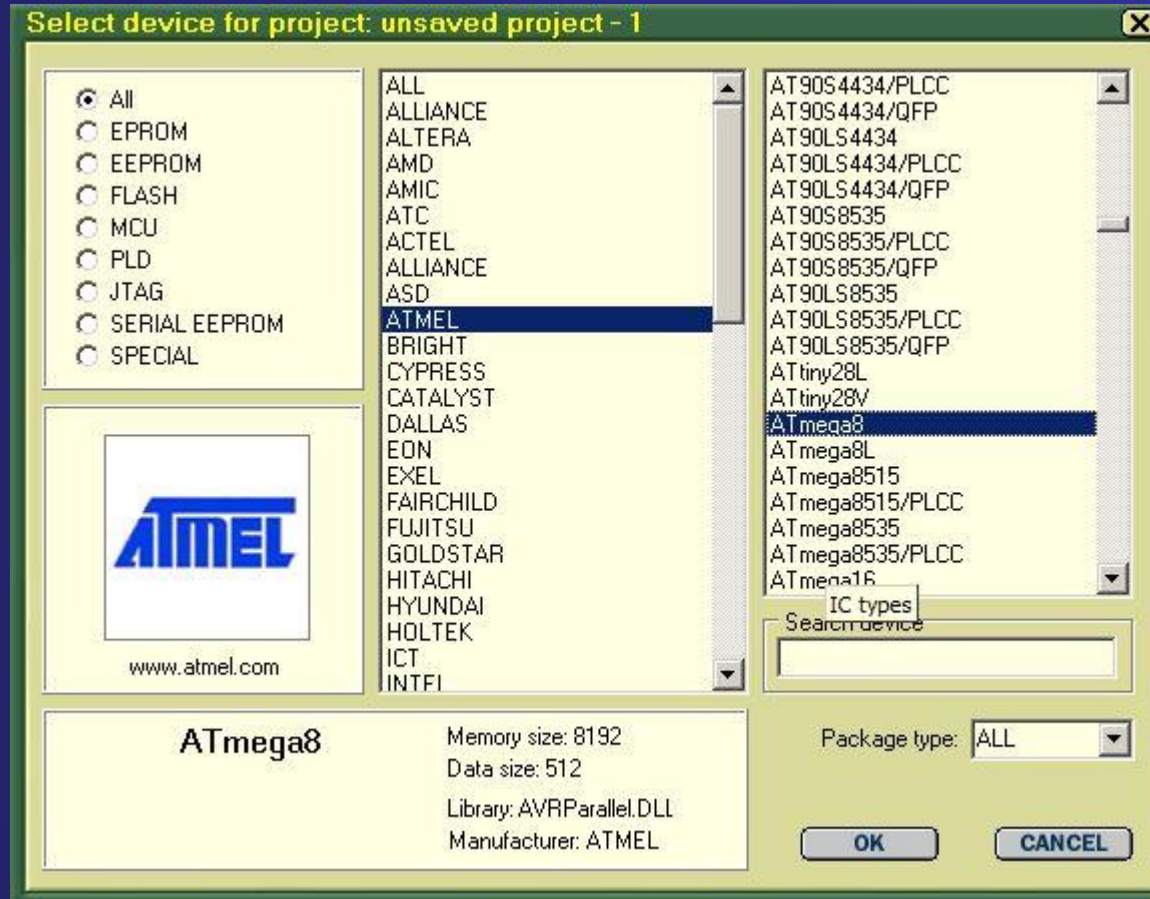




# Programowanie

## Mikrokontrolera

### Wybranie mikrokontrolera ATmega 8 firmy ATMEL



# Programowanie

Odczytanie zawartości pamięci mikroprocesora oraz konfiguracji bitów Fuse

The screenshot displays the 'Universal IC Programmer/Tester: DEMO mode' software interface. The main window shows a 'Project' section with buttons for 'NEW', 'OPEN', 'SAVE', and 'SAVE AS'. Below this is a 'Project name: Untitled' and 'File name: none' field. A 'Setup' dialog box is open, showing configuration options for the selected device, 'ATmega8'. The dialog is divided into four sections: 'Lock Bits', 'Fuse Bits', 'Fuse High Bits', and 'Fuse Extended Bits'. The 'Fuse Bits' section has 'SUTO' checked. The 'Fuse High Bits' section has 'BOOTSZ0' and 'BOOTSZ1' checked. The 'Fuse Extended Bits' section has all options set to 'unused'. Below these sections are 'Memory range' fields (Start address: 0000000, End address: 0001FFF, Address range: 0000000 - 0001FFF) and 'Calibration byte(s)' options (Write Calibration byte(s) at 0001FFB - 0001FFF). There are also checkboxes for 'Smart' and 'Pin Check', and 'OK' and 'CANCEL' buttons. The main window also shows a 'DEVICES' list with 'ATmega8' selected, a 'Device' section with a 'SETUP' button circled in red, and a 'File' section with 'LOAD' and 'SAVE' buttons. At the bottom, there is a status bar showing 'Code size: 8192', 'Data size: 512', and 'Checksum: E000'. The device information at the bottom reads 'Device: ATmega8', 'Type: MCU', and 'Manufacturer: ATMEL'.

Universal IC Programmer/Tester: DEMO mode

Project: NEW, OPEN, SAVE, SAVE AS

PROGRAM: PROGRAM, READ, ERASE, SINGLE

Chip ID Checking, Read, SKIP Verify

RUN, AUTO

Project name: Untitled, File name: none

Uprog, Close

DEVICES: AT89C2051, ATmega8

Device: +, -, SETUP, File: LOAD, SAVE, HELP, CONFIG, INFO

Lock Bits: Lock bit 1, Lock bit 2, Boot lock bit 01, Boot lock bit 02, Boot lock bit 11, Boot lock bit 12, unused, unused

Fuse Bits: CKSELO, CKSEL1, CKSEL2, CKSEL3, SUTO, SUT1, BODEN, BODLEVEL

Fuse High Bits: BOOTRST, BOOTSZ0, BOOTSZ1, EESAVE, CKOPT, SPIEN, WDTON, RSTDISBL

Fuse Extended Bits: unused, unused, unused, unused, unused, unused, unused, unused

Memory range: Start address: 0000000, End address: 0001FFF, Address range: 0000000 - 0001FFF

Calibration byte(s): Write Calibration byte(s) at 0001FFB - 0001FFF

Smart, Pin Check, OK, CANCEL

0 0 0 0 F 0 | FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

Code size: 8192, Data size: 512, Checksum: E000

Device: ATmega8, Type: MCU, Manufacturer: ATMEL, RK-SYSTEM



Table 4. Crystal Oscillator Operating Modes

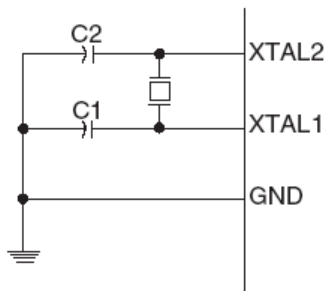
CKOPT	CKSEL3..1	Frequency Range(MHz)	Recommended Range for Capacitors C1 and C2 for Use with Crystals (pF)
1	101 <sup>(1)</sup>	0.4 - 0.9	-
1	110	0.9 - 3.0	12 - 22
<b>1</b>	<b>111</b>	3.0 - 8.0	12 - 22
0	101, 110, 111	1.0 ≤	12 - 22

Note: 1. This option should not be used with crystals, only with ceramic resonators.

The CKSEL0 Fuse together with the SUT1..0 Fuses select the start-up times as shown in Table 5.

Figure 11. Crystal Oscillator Connections

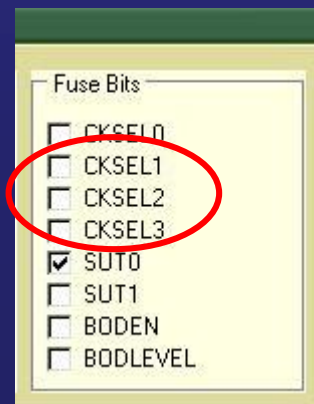
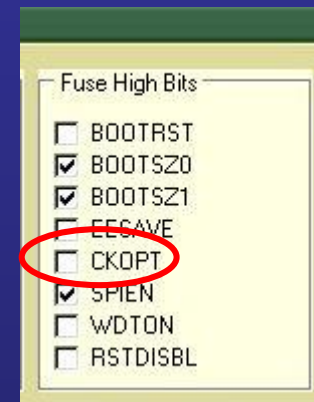
My będziemy używać oscylatora 8MHz



The Oscillator can operate in three different modes, each optimized for a specific frequency range. The operating mode is selected by the fuses CKSEL3..1 as shown in Table 4.

Nr bitu	Nazwa High Fuse Bitu	Wartość Fuse Bitu
0	BOOTRST	1
1	BOOTSZ0	0
2	BOOTSZ1	0
3	EESAVE	1
4	<b>CKOPT</b>	<b>1</b>
5	SPIEN	0
6	WDTON	1
7	RSTDISBL	1

Nr bitu	Nazwa Fuse Bitu	Wartość Fuse Bitu
0	CKSEL 0	1
1	CKSEL 1	1
2	CKSEL 2	1
3	CKSEL 3	1
4	SUT 0	0
5	SUT 1	1
6	BODEN	1
7	BODLEVEL	1



**Table 5. Start-up Times for the Crystal Oscillator Clock Selection**

CKSEL0	SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset ( $V_{CC} = 5.0V$ )	Recommended Usage
0	00	258 CK <sup>(1)</sup>	4.1 ms	Ceramic resonator, fast rising power
0	01	258 CK <sup>(1)</sup>	65 ms	Ceramic resonator, slowly rising power
0	10	1K CK <sup>(2)</sup>	–	Ceramic resonator, BOD enabled
0	11	1K CK <sup>(2)</sup>	4.1 ms	Ceramic resonator, fast rising power
1	00	1K CK <sup>(2)</sup>	65 ms	Ceramic resonator, slowly rising power
1	01	16K CK		Crystal Oscillator, BOD enabled
1	10	16K CK	4.1 ms	Crystal Oscillator, fast rising power
1	11	16K CK	65 ms	Crystal Oscillator, slowly rising power

- Notes:
1. These options should only be used when not operating close to the maximum frequency of the device, and only if frequency stability at start-up is not important for the application. These options are not suitable for crystals.
  2. These options are intended for use with ceramic resonators and will ensure frequency stability at start-up. They can also be used with crystals when not operating close to the maximum frequency of the device, and if frequency stability at start-up is not important for the application.

Nr bitu	Nazwa Fuse Bitu	Wartość Fuse Bitu
0	CKSEL 0	1
1	CKSEL 1	1
2	CKSEL 2	1
3	CKSEL 3	1
4	SUT 0	0
5	SUT 1	1
6	BODEN	1
7	BODLEVEL	1

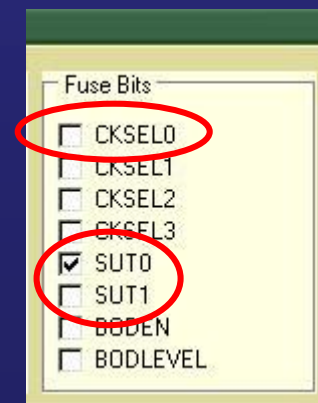


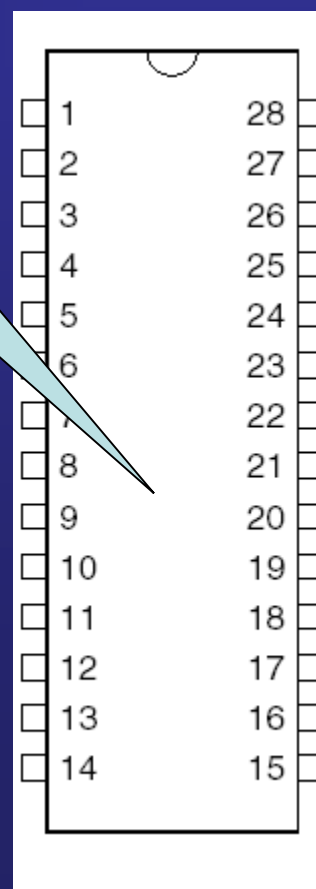
Table 9. Internal Calibrated RC Oscillator Operating Modes

CKSEL3..0	Nominal Frequency (MHz)
0001 <sup>(1)</sup>	1.0
0010	2.0
0011	4.0
0100	8.0

Note: 1. The device is shipped with this option selected.

Nr bitu	Nazwa Fuse Bitu	Wartość Fuse Bitu
0	CKSEL 0	0
1	CKSEL 1	0
2	CKSEL 2	1
3	CKSEL 3	0
4	SUTO 0	0
5	SUTO 1	1
6	BODEN	1
7	BODLEVEL	1

Oscillator RC inside





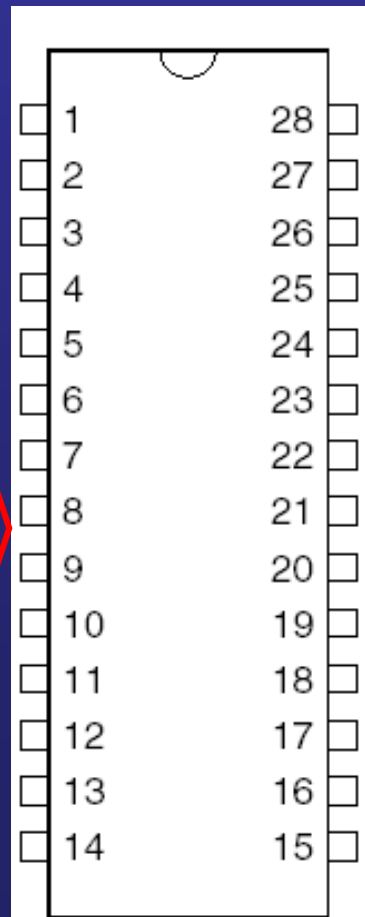
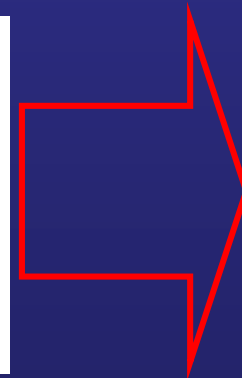
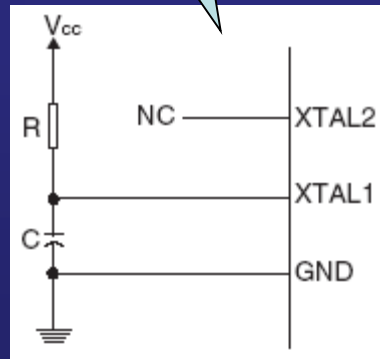
**Table 7. External RC Oscillator Operating Modes**

CKSEL3..0	Frequency Range (MHz)
0101	0.1 - 0.9
0110	0.9 - 3.0
0111	3.0 - 8.0
1000	8.0 - 12.0

When this Oscillator is selected, start-up times are determined by the SUT Fuses as shown in Table 8.

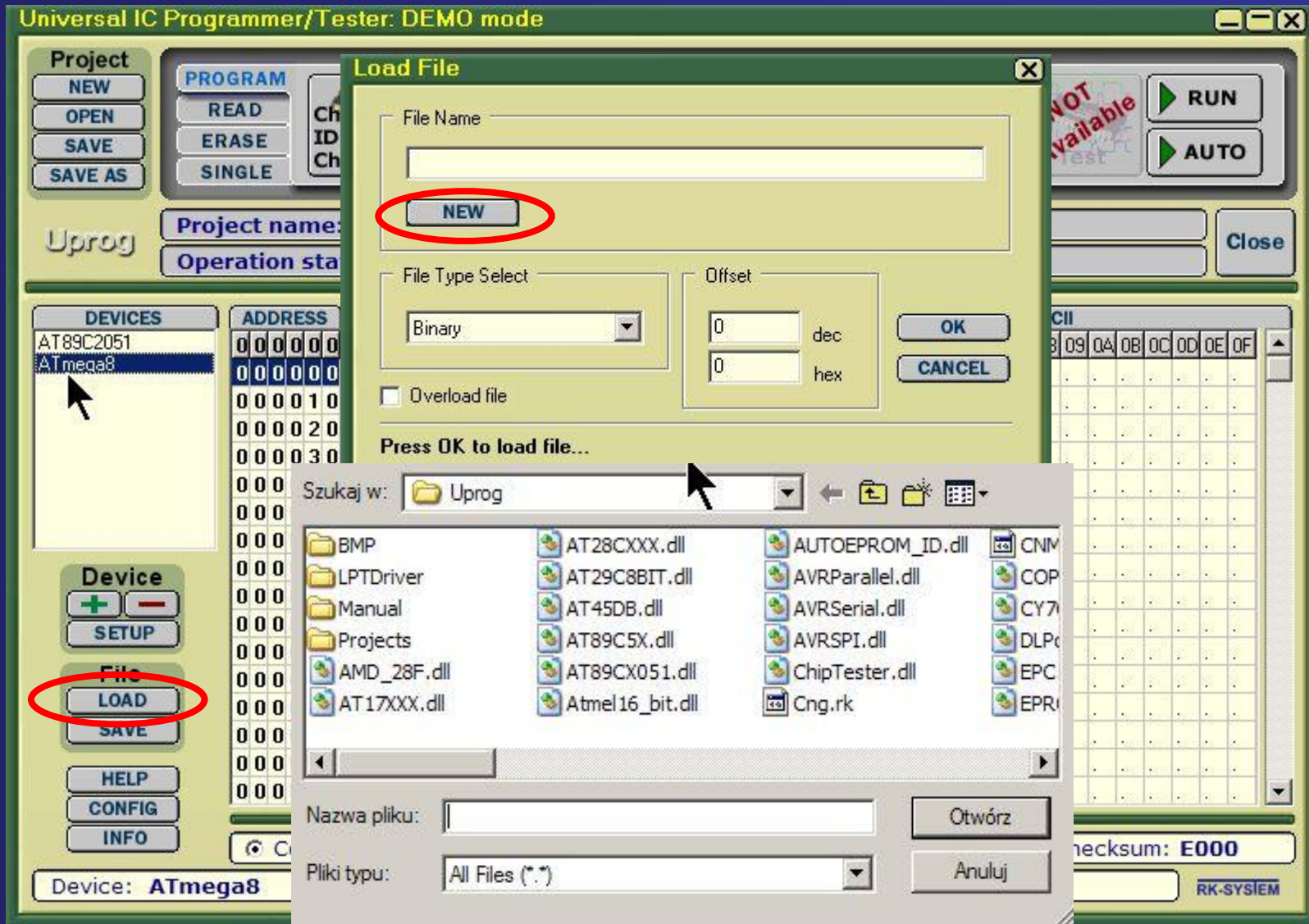
Nr bitu	Nazwa Fuse Bitu	Wartość Fuse Bitu
0	CKSEL 0	0
1	CKSEL 1	0
2	CKSEL 2	0
3	CKSEL 3	1
4	SUTO 0	0
5	SUTO 1	1
6	BODEN	1
7	BODLEVEL	1

Oscillator RC outside

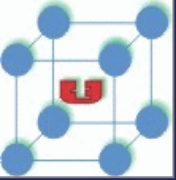


# Programowanie

Wczytywanie napisanego programu w postaci binarnej lub hexadecymalnej







# *Przyrządy używane na pracowni*





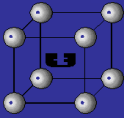
## Sonda Logiczna

„0” – zero logiczne

„1” – jedynka logiczna



# Przyrządy używane na pracowni



## Miernik cyfrowy

„0” – zero logiczne (0-0.8V)

„1” – jedynka logiczna (2-5V)

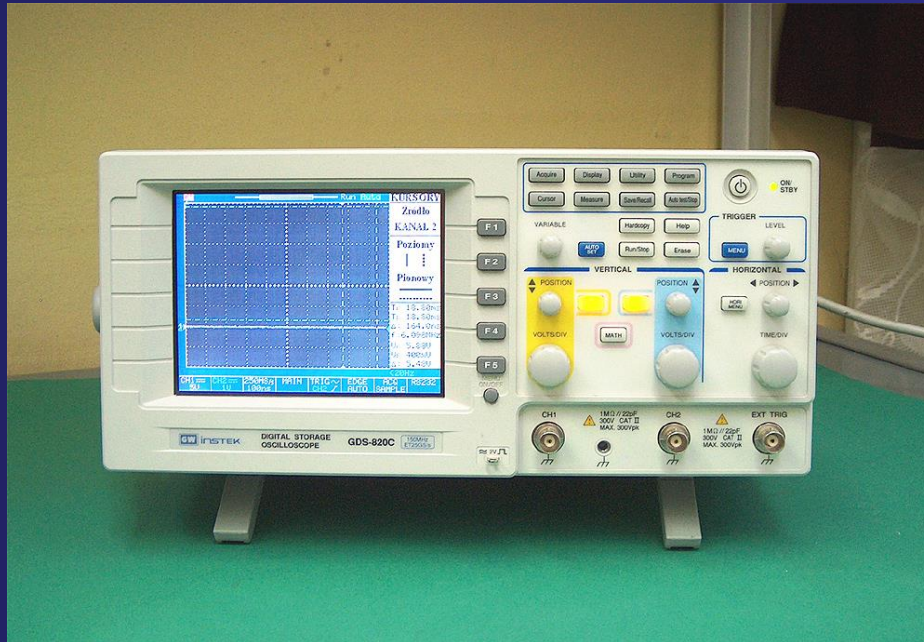




## Zasilacz stabilizowany

„0V – 30V” – napięcie

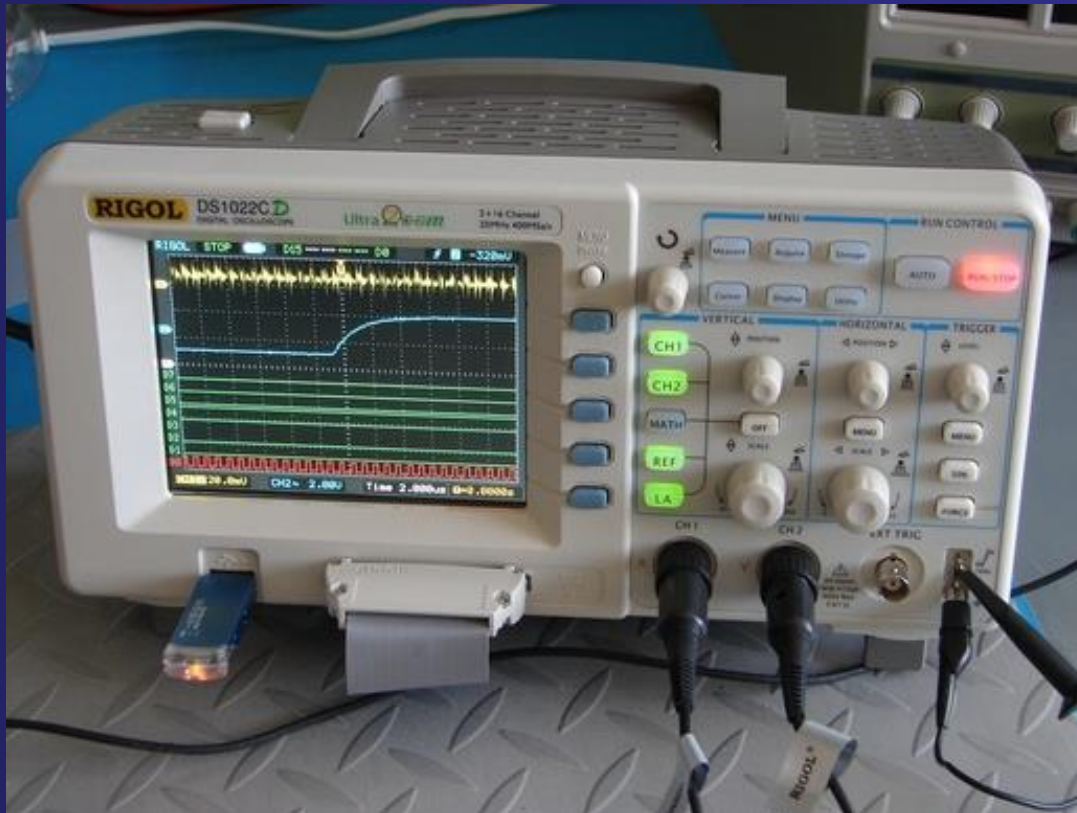
„0A – 1A” – ograniczenie prądowe



## Oscyloskop cyfrowy

Możemy rejestrować przebiegi napięć zmienne w czasie oraz zapisywać je w pamięci wewnętrznej oscyloskopu

## Analizator Stanów Logicznych

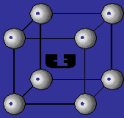


Możemy rejestrowane w czasie stany logiczne („0”, „1”) oraz zapamiętywać je w pamięci wewnętrznej analizatora

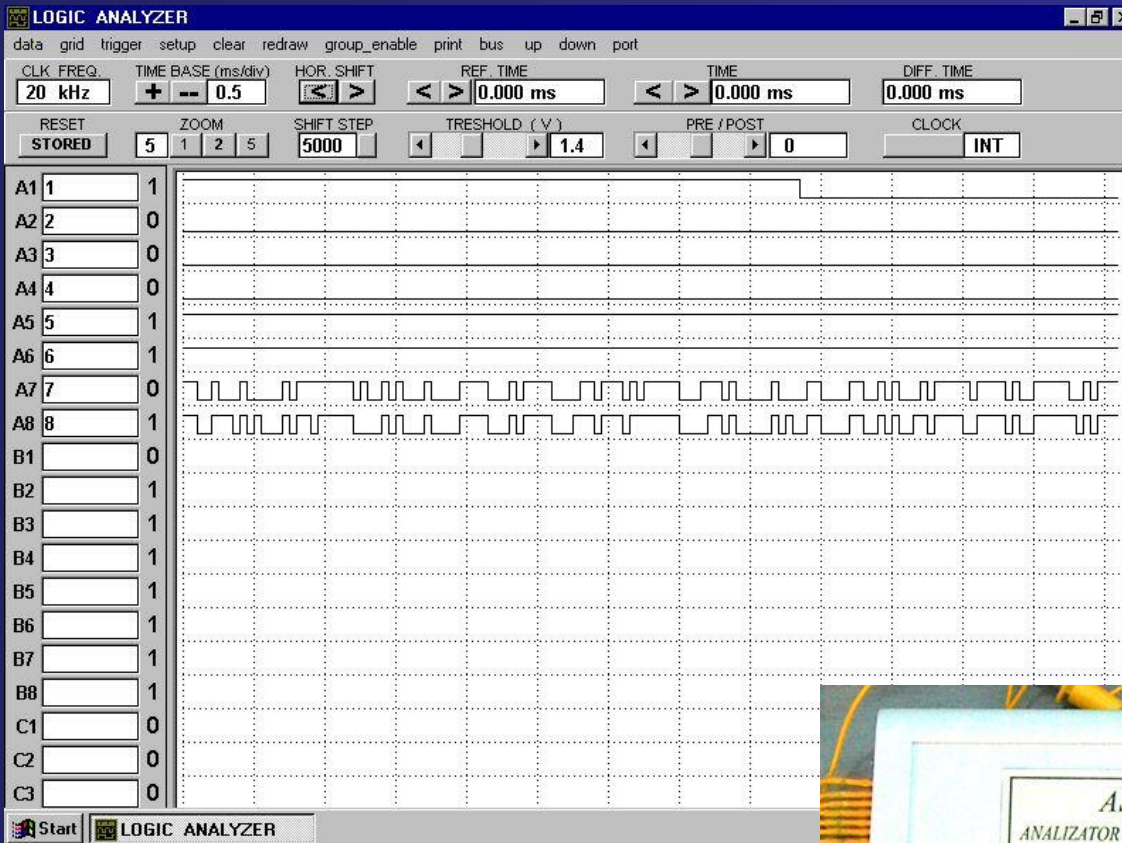




# Przyrządy używane na pracowni



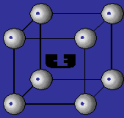
## Analizator Stanów Logicznych



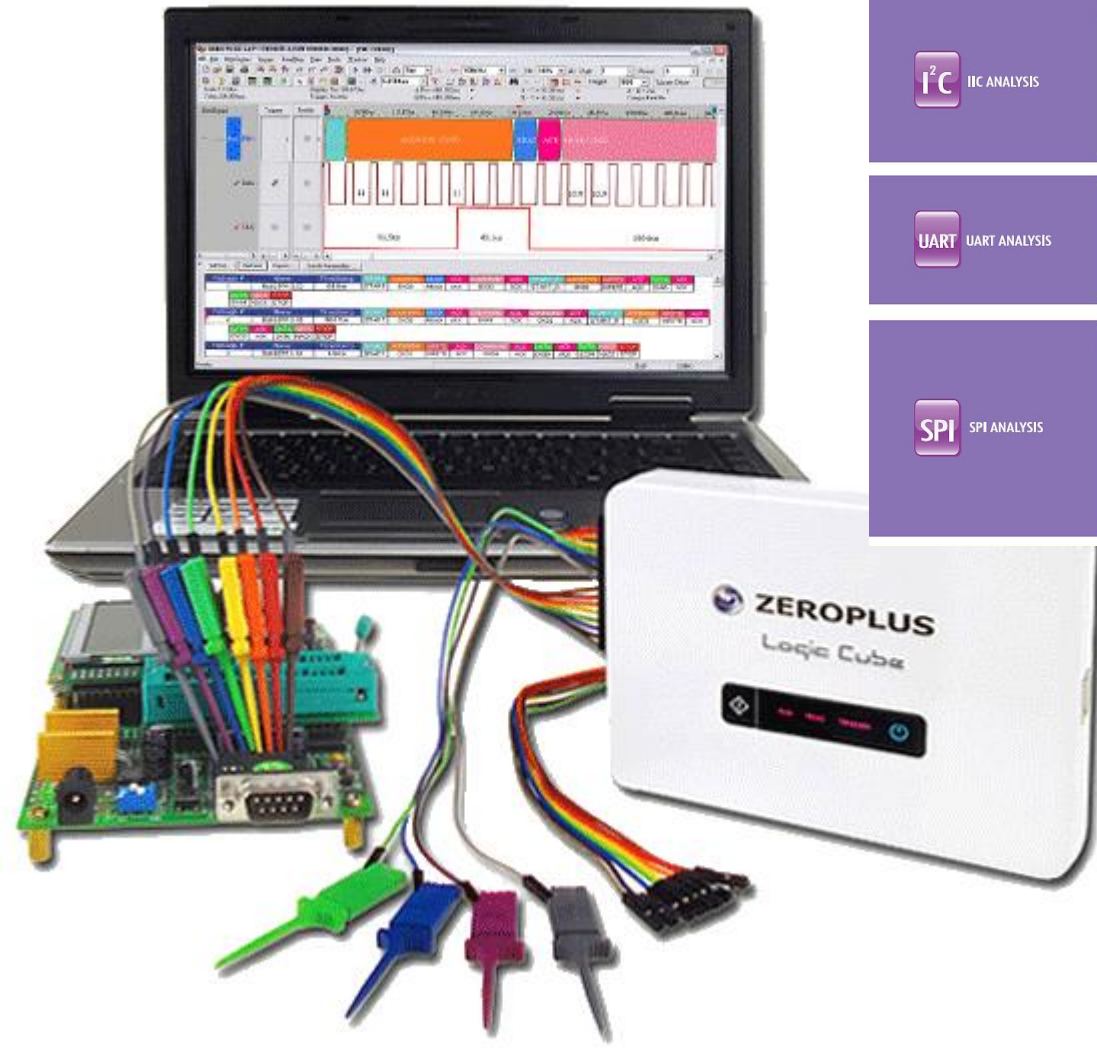
Rejestrowanie w czasie stanów logicznych

Możliwość zapamiętania danych w PC





## Analizator Stanów Logicznych



**UMOŻLIWIA**  
Rejestrowanie w czasie stanów logicznych i dekodowanie sygnałów różnych protokołów i magistral





Lab. Mikroprocesory

An iceberg floating in the ocean. The visible tip above the water is small and jagged, while the much larger, submerged part is hidden below the surface. A black circle highlights the tip, with a line connecting it to a grey box containing the text 'Lab. Mikroprocesory'. The text 'Mikrokontroler ATmega8' is overlaid on the submerged part of the iceberg.

**Mikrokontroler ATmega8**

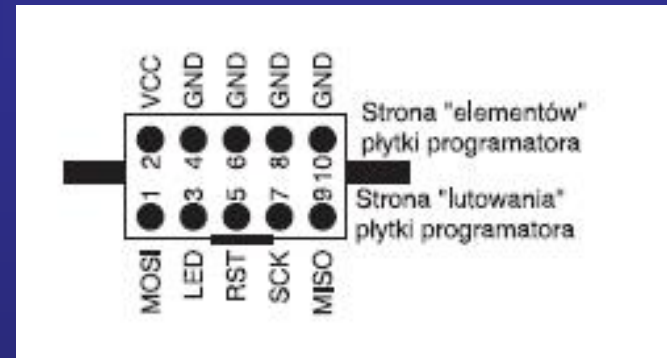
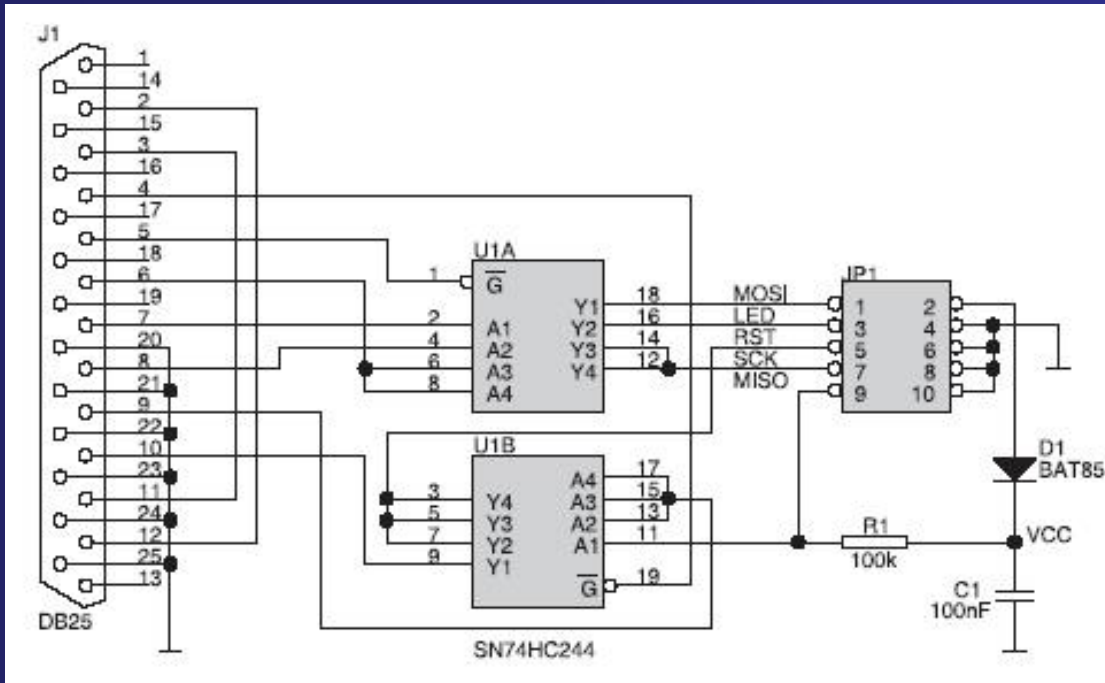
**Programowanie**

**Programowanie szeregowo  
poprzez interfejs ISP**

**Serial Peripheral Interface**

# Programowanie

Programowanie szeregowe poprzez interfejs ISP  
(Serial Peripheral Interface)



# Programowanie

Programowanie szeregowo poprzez interfejs ISP  
(Serial Peripheral Interface)





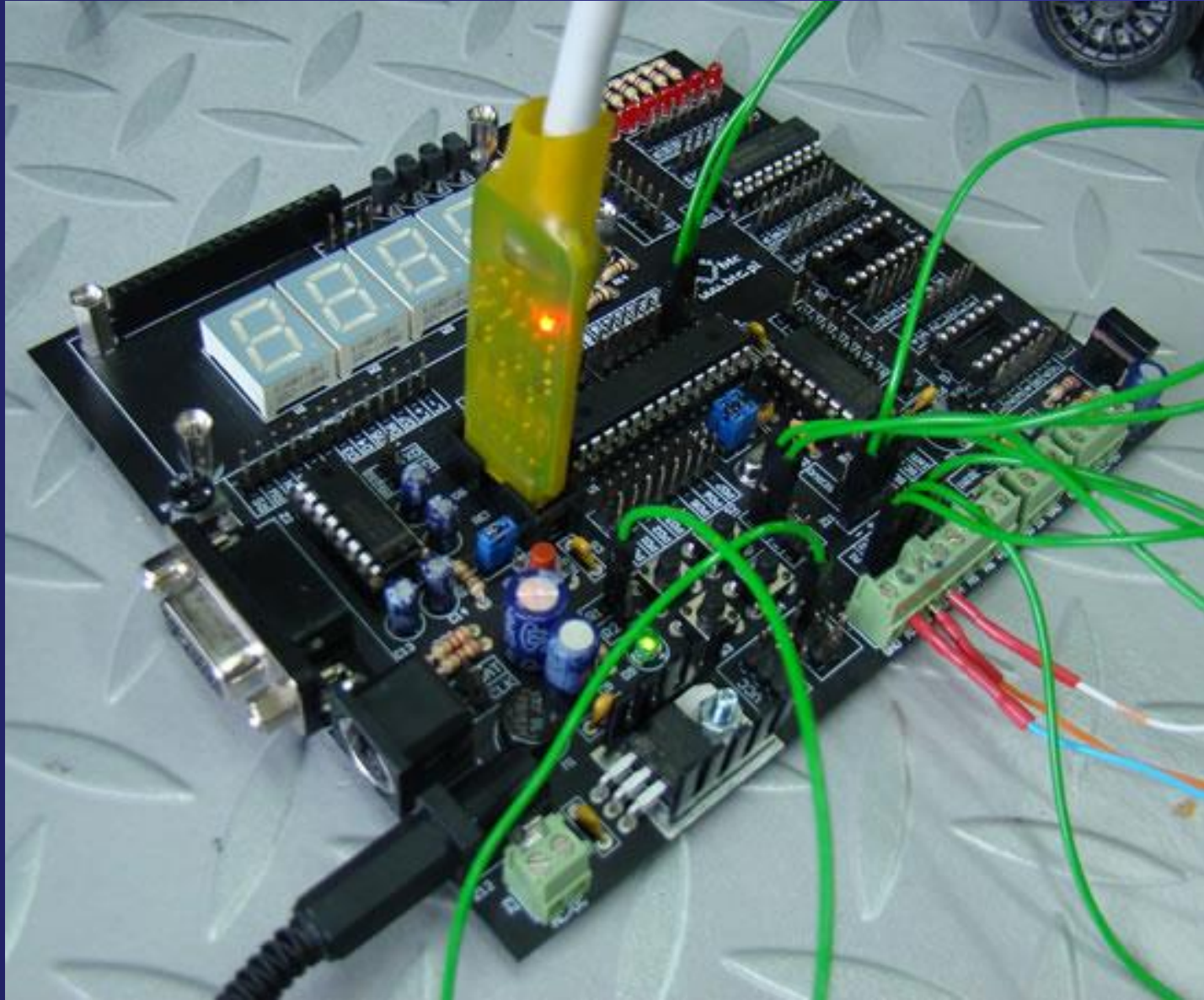
# Programowanie

Programowanie szeregowo poprzez interfejs ISP  
(Serial Peripheral Interface)



# Programowanie

Programowanie szeregowo poprzez interfejs ISP  
(Serial Peripheral Interface)





# Programowanie

Programowanie szeregowo poprzez interfejs ISP  
(Serial Peripheral Interface)

The image shows a screenshot of the 'BASCOM-AVR Options' dialog box, specifically the 'Programmer' tab. The dialog has a title bar and several tabs: 'Compiler', 'Communication', 'Environment', 'Simulator', 'Programmer' (selected), 'Monitor', and 'Printer'. The 'Programmer' section includes a dropdown menu for 'Programmer' set to 'STK200/STK300 Programmer', a 'Play sound' field with a 'Programmer Options' button and an ellipsis button, and four checkboxes: 'Erase warning' (unchecked), 'Auto Flash' (unchecked), 'AutoVerify' (checked), and 'Upload Code and Data' (unchecked). Below these are four sub-tabs: 'Parallel', 'Serial', 'Other', and 'Universal'. The 'Serial' sub-tab is active, showing an 'LPT-address' dropdown set to '378' with an adjacent '+' button, and a 'Port delay' text box set to '0'. At the bottom of the dialog are three buttons: 'Default', 'Ok' (with a green checkmark), and 'Cancel' (with a red X).

**BASCOM-AVR Options**

Compiler | Communication | Environment | Simulator | **Programmer** | Monitor | Printer

Programmer: STK200/STK300 Programmer

Play sound: [ ] Programmer Options [ ... ]

Erase warning    Auto Flash    AutoVerify    Upload Code and Data

Parallel | **Serial** | Other | Universal

LPT-address: 378 [ + ]

Port delay: 0

Default   **Ok**   **Cancel**

# Programowanie

Programowanie szeregowo poprzez interfejs ISP  
(Serial Peripheral Interface)

Ikona programatora

```
'$sim
$regfile = "m8def.dat"
$crystal = 8000000

Config Portd = Output
Config Pinb.0 = Input

Dim Kierunek As Bit

Przycisk Alias Pinb.0

Set Portb.0

Portd = &B11111110
Reset Kierunek

Do
  If Przycisk = 0 Then
    Waitms 50
    Toggle Kierunek
  Do
```

# Programowanie

Programowanie szeregowo poprzez interfejs ISP  
(Serial Peripheral Interface)

Ikona programatora

The screenshot shows the BASCOM-AVR IDE interface. The main window is titled 'AVR ISP STK programmer'. The 'Chip' dropdown menu is set to 'ATmega8'. The 'Manufacturer' is 'Atmel' and the 'Flash ROM' size is '8 KB'. The 'EEPROM' size is '512'. The 'Programmed' count is '324'. The 'FlashROM' tab is selected, showing a memory map table.

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
0000	C0	12	95	18	95	18	95	18	C1	46	95	18	95	18	95	18	R...AF...
0010	95	18	95	18	95	18	95	18	95	18	95	18	95	18	95	18	.....
0020	95	18	95	18	95	18	E5	8F	BF	8D	E4	C0	E4	E0	2E	4E	...iZzTaRaf.N
0030	E0	84	BF	8E	E0	D4	E0	F4	2E	5F	EF	EE	E0	F3	E6	A0	f_zZr0r0_dif06
0040	E0	B0	27	88	93	8D	97	31	F7	E9	D0	37	24	66	E0	88	r'rT-1+eD7\$fr
0050	BD	82	E0	85	BD	85	B7	89	64	80	BF	89	94	78	E6	A6	"f...%d02%"xc
0060	E0	B0	E3	EE	E0	F3	D0	F9	D0	B2	E6	AF	E0	B0	E4	E8	r'air000_0Zf'a0
0070	E0	F3	D0	F3	D0	B1	D0	19	D0	A3	E6	AF	E0	B0	D0	0D	r000000_0t0Zf'0.
0080	D0	12	D0	9E	E6	A6	E0	B0	D0	08	E6	84	E0	90	D0	89	0_0z0f'0_0_r0%
0090	CF	F2	94	F8	CF	FF	E2	A8	E0	B4	91	8D	23	88	F0	11	0kr0'ar'rT#id.
00A0	D0	4E	CF	FB	95	08	EC	80	C0	50	E0	81	D0	4E	D0	02	0N00_00APr0N0.
00B0	E8	80	C0	4B	E0	83	27	99	C0	74	EF	8A	E0	90	D0	71	00AKr'r'RidS'r0q
00C0	9A	A3	9A	A2	9A	A1	9A	A0	9A	A5	9A	A4	98	AC	E0	85	0t000000000-f...
00D0	E0	90	D0	67	98	AD	98	AB	98	AA	9A	A9	9A	A8	9A	AD	r00qH0IS0000-

AVR ISP STK programmer  
850 ROM    0 EPROM    PROGRAM17.BIN

Chip	
Name	MEGA8
Calibration 0	B7
Calibration 1	B7
Calibration 2	B3
Calibration 3	B4
Lockbits	
Lockbit 65	11:No restrictions for SPM or LPM accessing the boot loader section
Lockbit 43	11:No restrictions for SPM or LPM accessing the application section
Lockbit 21	11:No memory lock features enabled for parallel and serial programming
Fusebits	
Fusebit C	1:BODLEVEL 2.7V
Fusebit B	1:BODEN disabled
Fusebit KL	10:6 CK, 64 mS delay
Fusebit A987	0100:Internal RC oscillator 8 MHz
Fusebits High	
Fusebit M	1:PIN PC6 is RESET
Fusebit J	1:WDT enabled by WDTCR
Fusebit I	0:SPI enabled
Fusebit H	1:CKOPT 1
Fusebit G	1:Erase EEPROM when chip erase
Fusebit FE	00:1024 words boot size, C00
Fusebit D	1:Reset vector is \$0000

- Refresh
- Write LB
- Write FS
- Write FSH
- Write FSE
- Write PRG



AVR ISP STK programmer

File Buffer Chip

Chip ATmega8

Manufacturer **Atmel** Flash ROM **8 KB** Size  
Chip **ATmega8** EEPROM **512** Programmed:324

FlashROM | EEPROM | Lock and Fuse Bits

Chip	
Name	MEGA8
Calibration 0	B7
Calibration 1	B7
Calibration 2	B3
Calibration 3	B4

Lockbits	
Lockbit 65	11:No restrictions for SPM or LPM accessing the boot loader section
Lockbit 43	11:No restrictions for SPM or LPM accessing the application section
Lockbit 21	11:No memory lock features enabled for parallel and serial programming

Fusebits	
Fusebit C	1:BODLEVEL 2.7V
Fusebit B	1:BODEN disabled
Fusebit KL	10:6 CK, 64 mS delay
Fusebit A987	0100:Internal RC oscillator 8 MHz

Fusebits High	
Fusebit Hi	1001:1001 external low freq XTAL
Fusebit M	1010:1010
Fusebit J	1011:1011
Fusebit I	1100:1100
Fusebit L	1101:1101
Fusebit H	1110:1110 external XTAL
Fusebit G	1111:1111 external XTAL
Fusebit FE	00:1024 words boot size, CPU
Fusebit D	1:Reset vector is \$0000

Refresh  
Write LB  
Write FS  
Write FSH  
Write FSE  
Write PRG

AVR ISP STK programmer

850 ROM | 0 EPROM | PROGRAM17.BIN



# Programowanie

## Programowanie szeregowo poprzez interfejs ISP (Serial Peripheral Interface)

The screenshot shows the AVR ISP STK programmer interface. The 'Chip' dropdown menu is set to 'ATmega8'. The 'Flash ROM' size is 8 KB and the 'EEPROM' size is 512. The 'Programmed' count is 371. The 'FlashROM' tab is selected, displaying a memory table with columns for address (00 to 0F) and data. A red circle highlights the 'FlashROM' tab icon in the toolbar.

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
0000	C0	12	95	18	95	18	95	18	95	18	95	18	95	18	95	18	Á.....
0010	95	18	95	18	95	18	95	18	95	18	95	18	95	18	95	18	.....
0020	95	18	95	18	95	18	E5	8F	BF	8D	E4	C0	E4	E0	2E	4E	...iZzTáRár.N
0030	E0	84	BF	8E	E0	D4	E0	F4	2E	5F	EF	EE	E0	F3	E6	A0	f_zZr0ró_difóó
0040	E0	B0	27	88	93	8D	97	31	F7	E9	24	66	EF	8F	BB	81	f*rT-1+e\$fdZ»
0050	98	B8	9A	C0	EF	8E	BB	82	91	80	00	60	77	8F	93	80	l,sRdZ»,e.wZ»e
0060	00	60	27	00	27	BB	E3	A6	91	8C	FB	80	F4	0E	E0	01	!>á!S00ó.f.
0070	E0	40	17	04	F0	09	C0	1B	E3	82	E0	90	D0	3F	E6	A0	f@..d.R.ã.rD?ó
0080	E0	B0	91	8C	FB	87	27	88	F9	80	95	80	FB	80	E6	A0	f*S0#100+000ó
0090	E0	B0	91	8C	F9	87	93	8C	27	00	27	BB	E3	A6	91	8C	f*S0#S!>á!S
00A0	FB	80	F4	0E	E0	01	E0	41	17	04	F0	09	CF	F5	27	00	00ó.f.rA..d.Dó.
00B0	E6	A0	E0	B0	91	8C	FB	87	F4	0E	E0	01	E0	41	17	04	ó f*S0#0ó.f.rA..
00C0	F0	09	C0	07	E0	91	B3	82	27	BB	E1	A8	D0	26	BB	82	d.R.fT!>áD&».
00D0	C0	06	E0	91	B3	82	27	BB	E1	A8	D0	29	BB	82	E6	84	R.fT!>áD!>ó..

AVR ISP STK programmer

322 ROM      0 EPROM      WAZ\_PRZYCISK.BIN

