Asembler

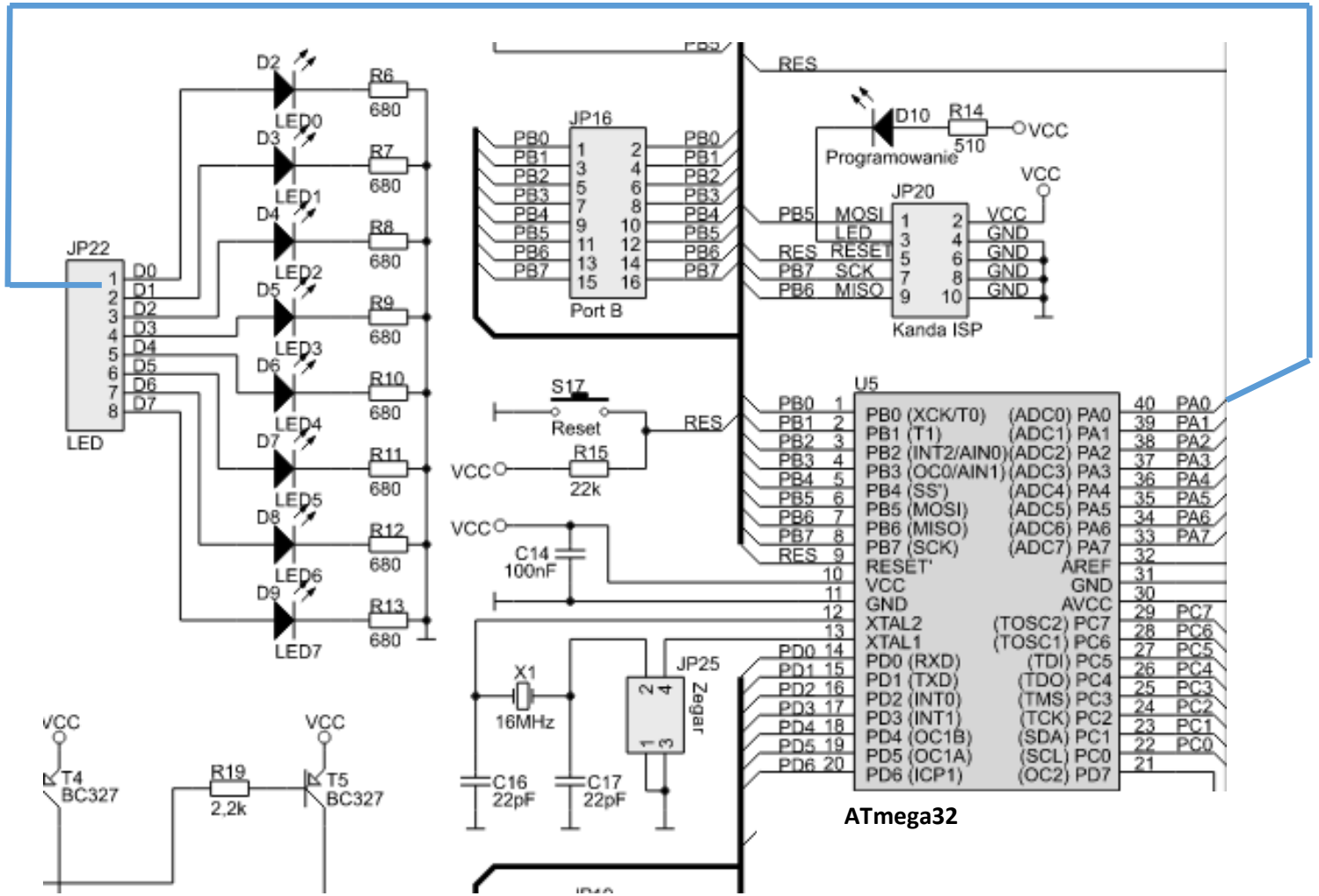
Program pierwszy– włączenie i wyłączenie diody

CELE

1. **Podstawowe informacje o portach**
2. **Programowanie portów jako wyjście**
3. **Program włączający i wyłączający naprzemiennie diodę LED D2**
4. **Przykład użycia języka maszynowego**

WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

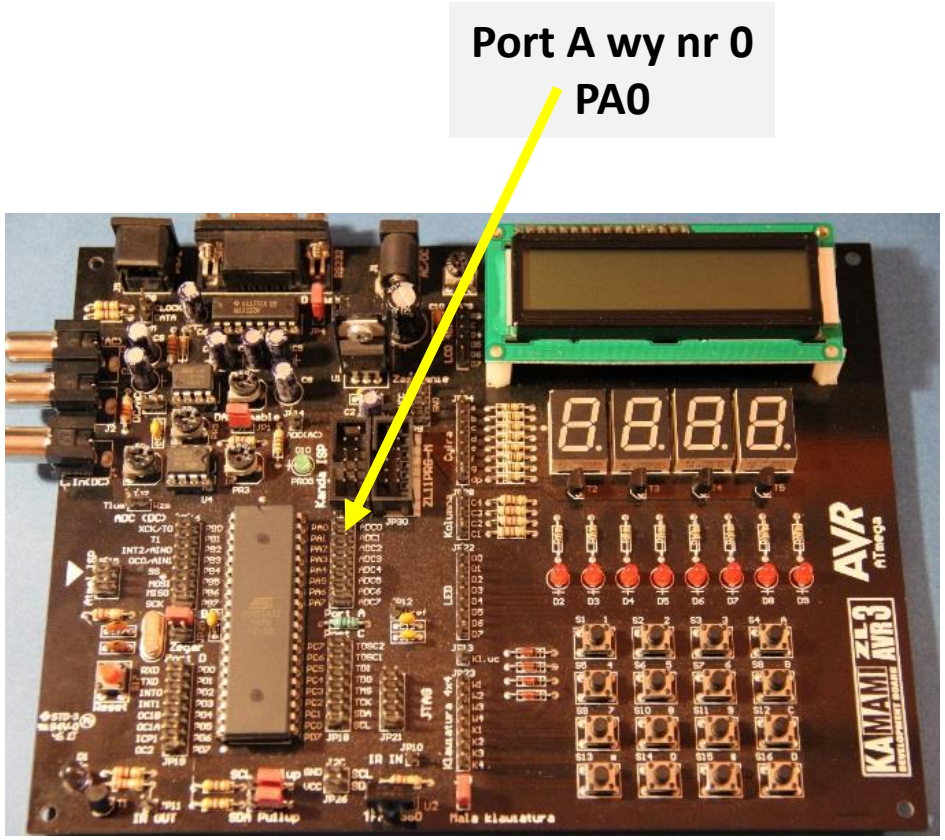
Operacje na portach – schemat połączeń



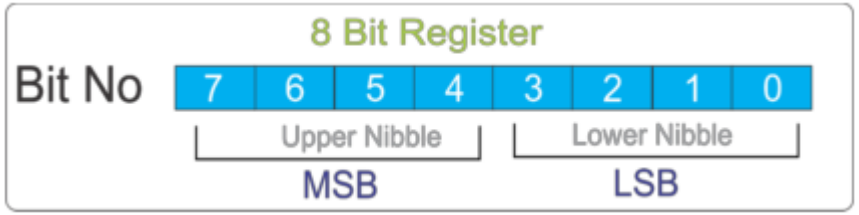
Pierwszy program z użyciem portów

```
.nolist
.include"m32def.inc"
.list

ldi r16, 0b00000001        ; załaduj do rejestru R16 wartość 1
out ddra,r16               ; Port A jego wyprowadzenie nr 0 jako port wyjściowy (output)
ldi r17, 0b00000000        ; załaduj do rejestru R17 wartość 0


loop:                      ; etykieta pętli

out porta,r16              ; zapisz do Portu A wartość rejestru R1
out porta,r17              ; zapisz do Portu A wartość rejestru R1

jmp loop                   ; skok bezpośredni do adresu w pamięci danych etykiety loop


.exit
```

Program – wystawienie naprzemienne wartości logicznej „1” „0” na porcie A



Język wewnętrzny (maszynowy)



Mnemonik	Rozwinięcie	Operacja	SREG I T H S V N Z C
ldi	**Load Immediate** Ładuje rejestr bezpośrednio stałą	PC ← PC+1 Rh ← C255	- - - - - - - -

R16 – adres 0x10 = 00010000

Kod Rozkazu ldi:

1	1	1	0	c	c	c	c	h	h	h	h	c	c	c	c

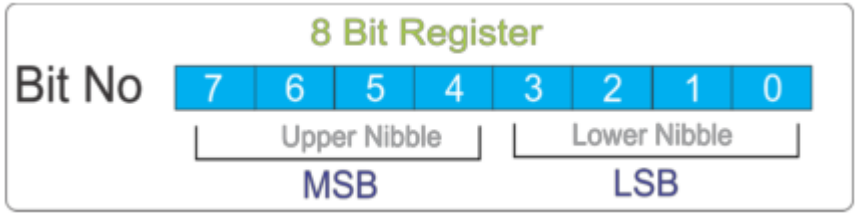
ldi r16, 0b00000001

1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1

Kod Rozkazu: **E0 01**

Jest to pierwszy rozkaz kodu maszynowego, który zostanie umieszczony w zerowej komórce pamięci FLASH

Język wewnętrzny (maszynowy)



Mnemonik	Rozwinięcie	Operacja	SREG I T H S V N Z C
out	**Out**put port Zapisz wartość rejestru w przestrzeni we-wy	(P+0x20) ← Rs PC ← PC+1	- - - - - - - -

R16 – adres 0x10 = 00010000

Kod Rozkazu out:

1	0	1	1	1	p	p	s	s	s	s	s	p	p	p	p

out ddra, r16

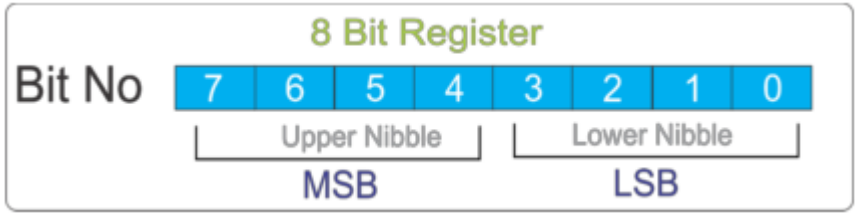
ddra – adres 0x1a = 00011010

1	0	1	1	1	0	1	1	0	0	0	0	1	0	1	0

Kod Rozkazu: **BB 0A**

Jest to drugi rozkaz kodu maszynowego, który zostanie umieszczony w pierwszej komórce pamięci FLASH

Język wewnętrzny (maszynowy)



Mnemonik	Rozwinięcie	Operacja	SREG I T H S V N Z C
ldi	**Load Immediate** Ładuje rejestr bezpośrednio stałą	PC ← PC+1 Rh ← C255	- - - - - - - -

R16 – adres 0x10 = 00010001

Kod Rozkazu ldi:

1	1	1	0	c	c	c	c	h	h	h	h	c	c	c	c

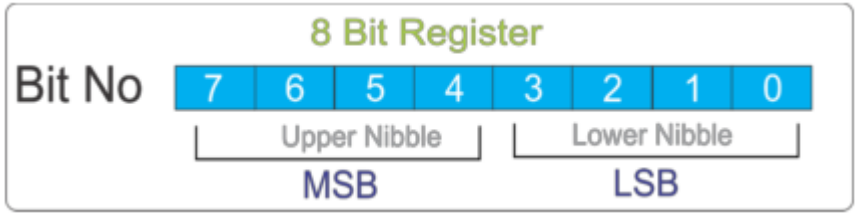
ldi r17, 0b00000000

1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0

Kod Rozkazu: **E0 10**

Jest to pierwszy rozkaz kodu maszynowego, który zostanie umieszczony w zerowej komórce pamięci FLASH

Język wewnętrzny (maszynowy)

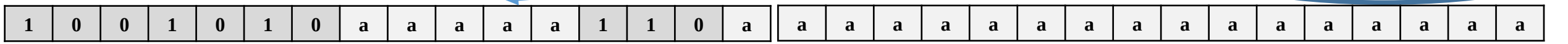


Mnemonik	Rozwinięcie	Operacja	SREG I T H S V N Z C
jmp	JuMP Skok adresowany bezpośrednio Liczba słów 2 (4bajty)	PC ← PC+adr4M	- - - - - - - -

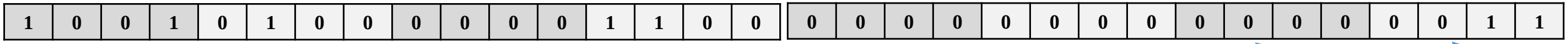
loop – adres 0x0003 =

0000000000000011

Kod Rozkazu jump:



jump loop

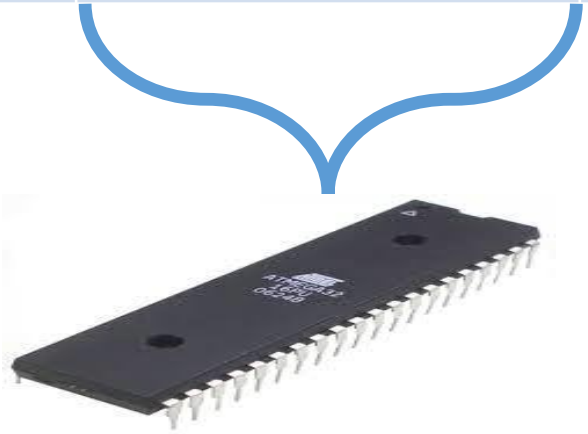


Kod Rozkazu: **94 0C 0003**

Jest to pierwszy rozkaz kodu maszynowego, który zostanie umieszczony w zerowej komórce pamięci FLASH

Język wewnętrzny (maszynowy)

Adres w pamięci FLSH kodów programu	Kod maszynowy	Mnemoniki
000000	E001	Ldi R16 ,0b00000001
000001	BB0A	out ddra, R16
000002	E010	ldi R17,0b00000000
		Loop:
000003	BB0B	out porta, R16
000004	BB1B	out porta, R17
000005	940C 0003	jmp loop



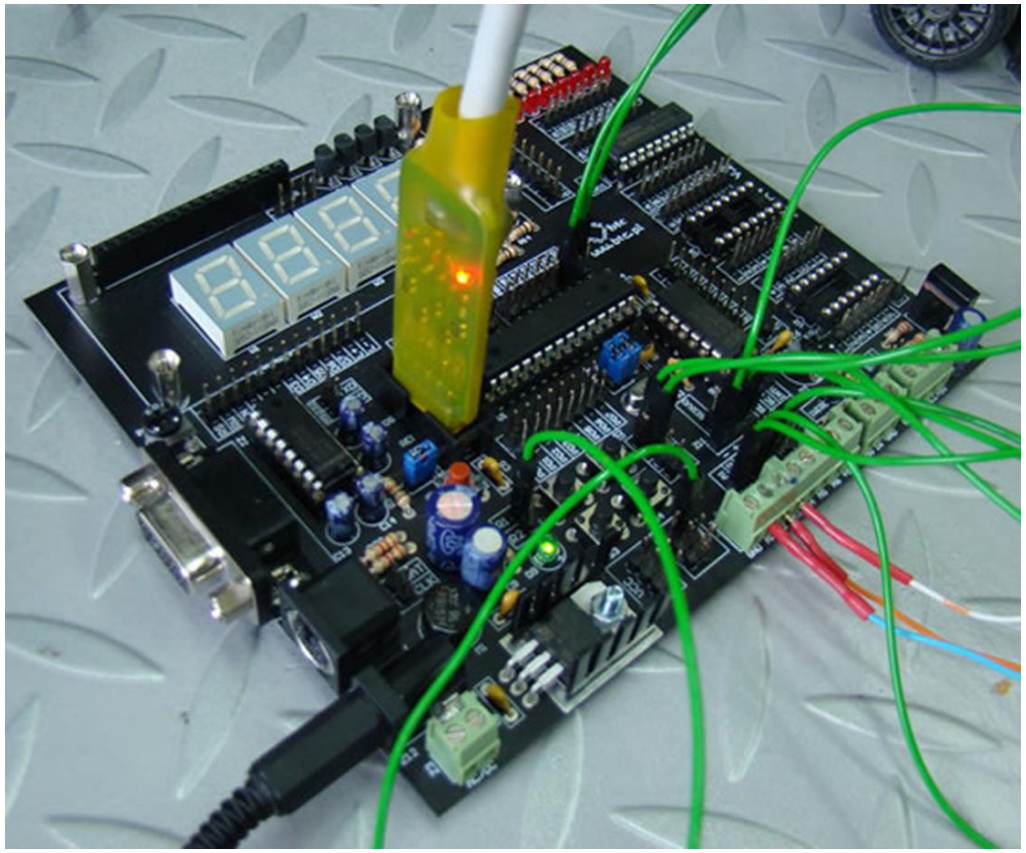
Instrukcje zamienione na kod maszynowy musimy umieścić w pamięci flash mikrokontrolera Atmega 32

Język wewnętrzny (maszynowy)

Programowanie

Programowanie szeregowe poprzez interfejs ISP Serial Peripheral Interface)

Adres w pamięci FLSH	Kod maszynowy
000000	E001
000001	BB0A
000002	E010
000003	BB0B
000004	BB1B
000005	940C 0003



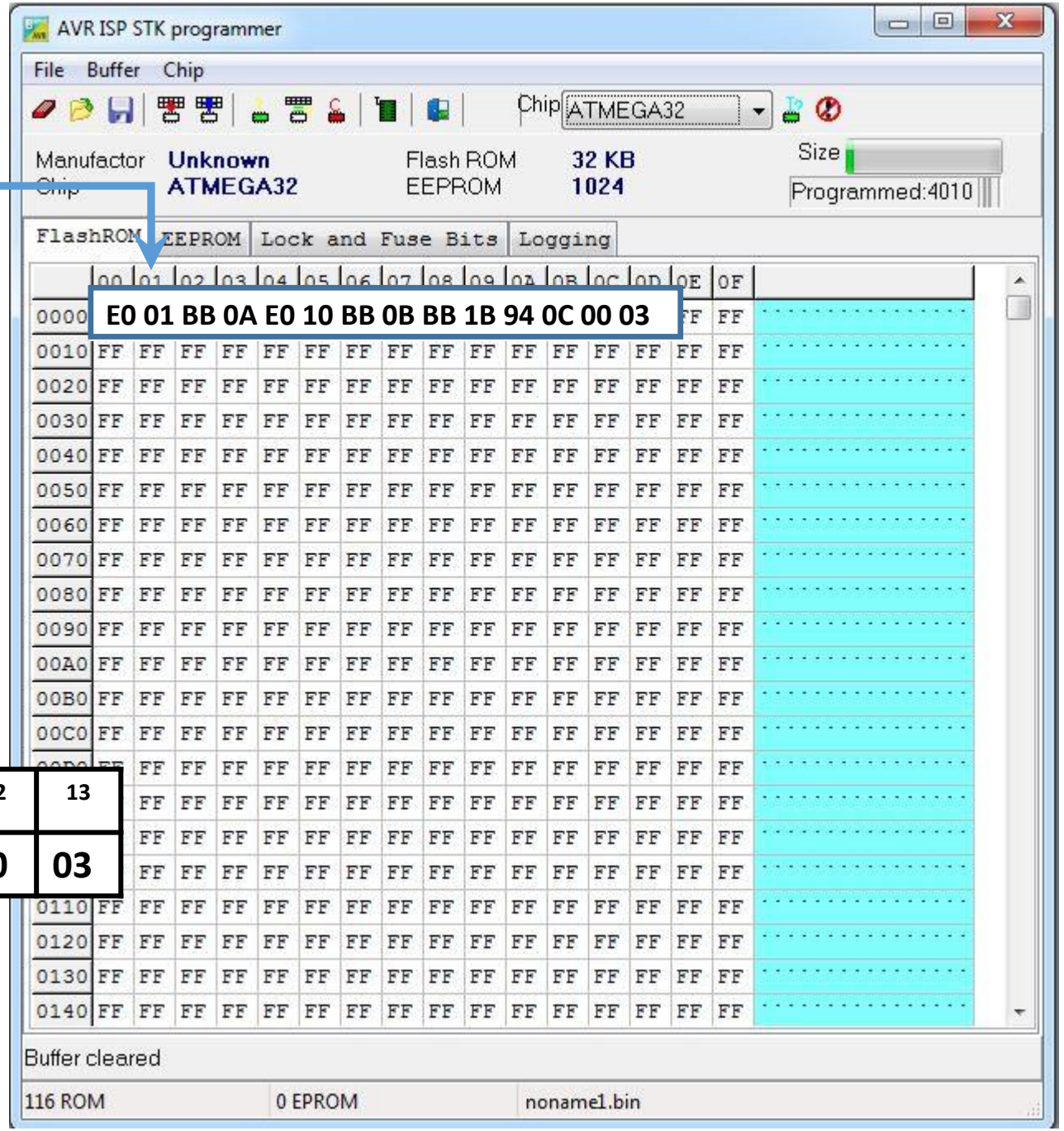
Instrukcje zamienione na kod maszynowy musimy umieścić w pamięci flash mikrokontrolera Atmega 32 użyjemy do tego programatora AVR ISP STK

Język wewnętrzny (maszynowy)

Adres w pamięci FLSH	Kod maszynowy
000000	E001
000001	BB0A
000002	E010
000003	BB0B
000004	BB1B
000005	940C 0003

0	1	2	3	4	5	6	7	8	9	10	11	12	13
E0	**01**	**BB**	**0A**	**E0**	**10**	**BB**	**0B**	**BB**	**1B**	**94**	**0C**	**00**	**03**

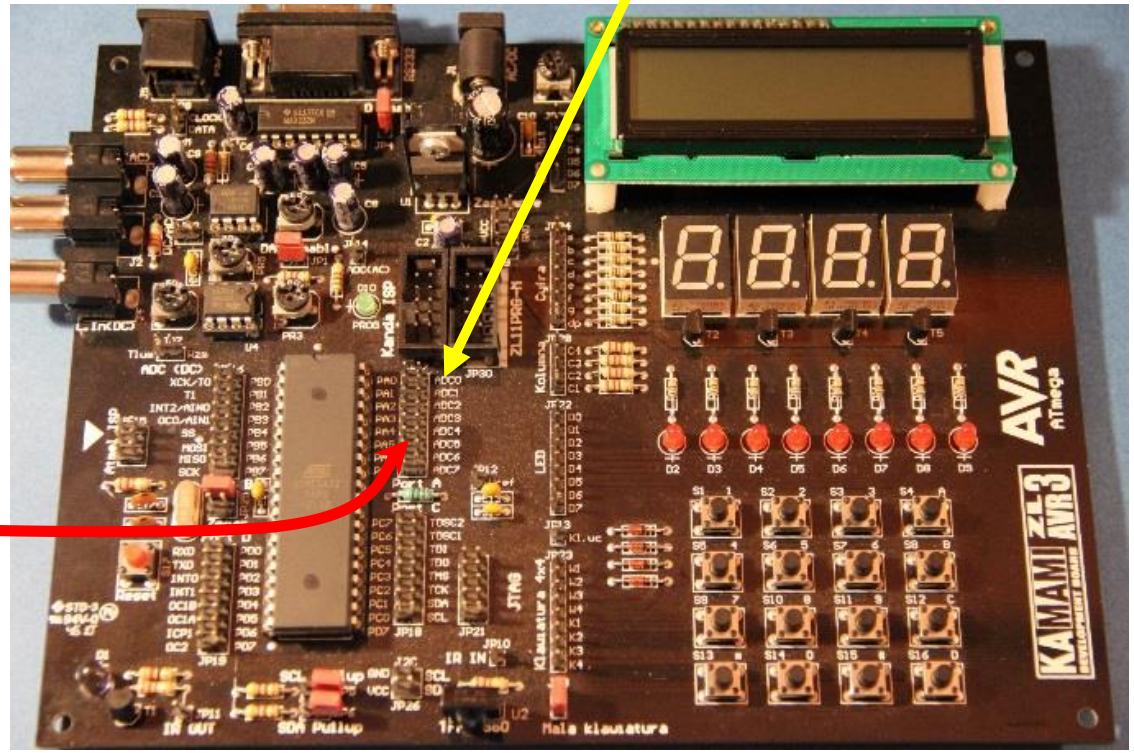
Instrukcje zamienione na kod maszynowy musimy umieścić w pamięci flash mikrokontrolera Atmega 32 jeśli komórki pamięci są przedstawione w postaci 8-bitowej dzielimy słowo 16-bitowe na połowę



Pierwszy program z użyciem portu A kod maszynowy

Program – wystawienie naprzemienne wartości logicznej „1” „0” na porcie A

Port A wy nr 0
PA0



Pierwszy program z użyciem portu A kod maszynowy

Mimo to że kod jest bardzo prosty i zajmuje tylko 14 bajtów to można go jeszcze zoptymalizować !!!!!

Optymalizacja polega na wykorzystaniu instrukcji rjmp zamiast jmp

Mnemonik	Rozwinięcie	Operacja	SREG I T H S V N Z C
jmp	**JuMP** Skok adresowany bezpośrednio Liczba słów 2 (4bajty)	PC ← PC+adr4M adr4M=zakres 0…262143	- - - - - - - - - -

Mnemonik	Rozwinięcie	Operacja	SREG I T H S V N Z C
rjmp	**Relative JuMP** Skok adresowany względnie Liczba słów 1 (2bajty)	PC ← PC+adr2k+1 adr2k= zakres -2048…2047	- - - - - - - - -

Instrukcja rjmp działa najszybciej, dlatego zalecane jest jej stosowanie wszędzie tam, gdzie jest to możliwe (gdy długość skoku jest mniejsza lub równa ±2 ksłowa). Instrukcja rjmp wykonywana jest w duch cyklach zegara szybciej niż jmp tu przepadają trzy cykle zegara