ATmega328p PLONKEC

BITNE OPERACIJE

Nastavljanje bita	BYTE |= (1<<BIT);
Odstavljanje bita	BYTE &= ~(1<<BIT);
Spreminjanje bita	BYTE ^= (1<<BIT);
Bit na 1?	if (BYTE &= (1 << BIT))
Bit na 0?	if (!(BYTE &= (1 << BIT)))

DIGITALNI OUTPUT

Primer (make45)

```
#define F_CPU 16000000L
#include <avr/io.h>
#include <util/delay.h>
int main(void) {
  DDRB |= 0b0000001; //smer (output)
  while (1) {
    PORTB = 0b00000001; //pin 0 porta B
    _delay_ms(1000);
    PORTB = 0b00000000;
    _delay_ms(1000);
  }
  return (0);
}
```

DIGITALNI INPUT

Nastavljanje smeri (make111)

```
DDRB &= ~(1 << BIT); //port B.BIT
```

Je input low? (make108)

```
if ((PORTB & (1 << BIT)) == 0)  oz.
if (!(PORTB & (1 << BIT)))
```

SERIJSKA KOMUNIKACIJA

Inicializacija (potrebuje util/setbaud.h) (make89)

```
UBRR0H = UBRRH_VALUE;
UBRR0L = UBRRL_VALUE; //baud rate
UCSR0B = (1 << TXEN0) | (1 << RXEN0); //enable tx, rx
UCSR0C = (1 << UCSZ01) | (1 << UCSZ00); //8 bits, 1 stop
```

Pošiljanje byta (make89)

```
while ( !( UCSR0A & (1<<UDRE0)) );
UDR0 = data;
```

Sprejemanje byta (make89)

```
while ( !(UCSR0A & (1<<RXC0)) );
data =UDR0;
```

ČASOVNIKI (TIMERJI)

Inicializacija časovnika 1 (štetje ure) (make179)

```
TCCR1B |= (1 << CS12) | (1 << CS10);
cas = TCNT1;
```

PREKINITVE

Inicializacija prekinitve 0 (PD2) (make157)

```
EIMSK |= (1 << INT0);
EICRA |= (1 << ISC00);
sei();
```

Prekinitvena rutina (make160)

```
ISR(INT0_vect) {
  ....
}
```

PULZNO ŠIRINSKA MODULACIJA

Nastavitev časovnikov (make205)

```
 /* Fast PWM mode, 8-bit */
TCCR1A |= (1 << WGM10);
/* Fast PWM mode, pt.2 */
TCCR1B |= (1 << WGM12);
/* PWM Freq = F_CPU/8/256 */
TCCR1B |= (1 << CS11);
 /* PWM output on OCR1A */
TCCR1A |= (1 << COM1A1);
/* PWM output on OCR1B */
TCCR1A |= (1 << COM1B1);
```

start (make205)

```
OCR1A = vrednost_med_0_in_255;
```

AD PRETVORBA

Inicializacija ADC0 (make135)

```
ADMUX |= (1 << REFS0);
ADCSRA |= (1 << ADPS1) | (1 << ADPS0);
ADCSRA |= (1 << ADEN);
```

Zajem (make136)

```
ADCSRA |= (1 << ADSC);
while (! (ADSCRA &= (1 << ADCS)) );
adcValue = ADC;
```

KNJIŽNICA

Definicije pinov/portov	avr/io.h
Funkcije za čakanje	util/delay.h
Serijska komunikacija	util/setbaud.h
Prekinitve	avr/interrupt.h

ATmega328p PLONKEC - PRIMERI

BLINK

```
#define F_CPU 16000000L

#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
 DDRB |= (1 << PB5);
    while(1)
    {
        PORTB |= (1 << PB5);
  _delay_ms(200);
  PORTB &= ~(1 << PB5);
  _delay_ms(200);
    }
}
```

ZUNANJA PREKINITEV NA PD2

```
#include <avr/io.h>
#include <avr/interrupt.h>

ISR(INT0_vect) {
 PORTD ^= (1 << PD6);
}

int main(void)
{
 DDRD &= ~(1 << DDD2);
 DDRD |= (1 << DDD6);

 EIMSK |= (1 << INT0);
 EICRA |= (1 << ISC01);
 sei();
    while(1) { }
}
```

PWM S TIMERJEM 0

```
#define F_CPU 16000000L

#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
 DDRD |= (1 << DDD6);

 //fast PWM
 TCCR0A |= (1 << WGM01);
 TCCR0A |= (1 << WGM00);
 //zaganjanje timerja (prescaler)
 TCCR0B |= (1 << CS00);

 TCCR0A |= (1 << COM0A1);

    while(1) {
 OCR0A = 50;
 _delay_ms(1000);
 }
}
```

SERIJSKA KOMUNIKACIJA

```
#define F_CPU 16000000
#define BAUD 9600

#include <avr/io.h>
#include <util/setbaud.h>
#include <util/delay.h>

char data;

int main(void) {
 DDRB |= (1<<PB0);
 UBRR0H = UBRRH_VALUE;
 UBRR0L = UBRRL_VALUE;
 UCSR0B = (1 << TXEN0) | (1 << RXEN0);
 UCSR0C = (1 << UCSZ01) | (1 << UCSZ00);

 DDRB |= (1<<DDB5);

 while(1)
 {
  //while ( !( UCSR0A & (1<<UDRE0)) );
  //UDR0 = data;
  while ( !(UCSR0A & (1<<RXC0)) );
  data =UDR0;
  if(data=='o') {
   PORTB |= (1<<PORTB5);
  }
  else
    PORTB &= ~(1<<PORTB5);
    _delay_ms(1);
 }
}
```

SERIJSKA KOMUNIKACIJA UTILITIES

```
void transmitByte(uint8_t data) {
  loop_until_bit_is_set(UCSR0A, UDRE0);
  UDR0 = data;
}

uint8_t receiveByte(void) {
  loop_until_bit_is_set(UCSR0A, RXC0);
  return UDR0;
}

void printString(const char myString[]) {
  uint8_t i = 0;
  while (myString[i]) {
    transmitByte(myString[i]);
    i++;
  }
}

void printByte(uint8_t byte) {
  transmitByte('0' + (byte / 100));
  transmitByte('0' + ((byte / 10) % 10));
  transmitByte('0' + (byte % 10));
}

void printWord(uint16_t word) {
  transmitByte('0' + (word / 10000));
  transmitByte('0' + ((word / 1000) % 10));
  transmitByte('0' + ((word / 100) % 10));
  transmitByte('0' + ((word / 10) % 10));
  transmitByte('0' + (word % 10));
}
```