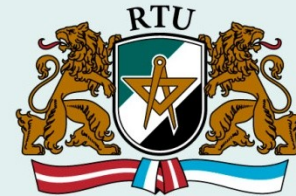


1. Praktiskā nodarbība

Iepazīšanās ar Charon II

Skaitļu pieraksts



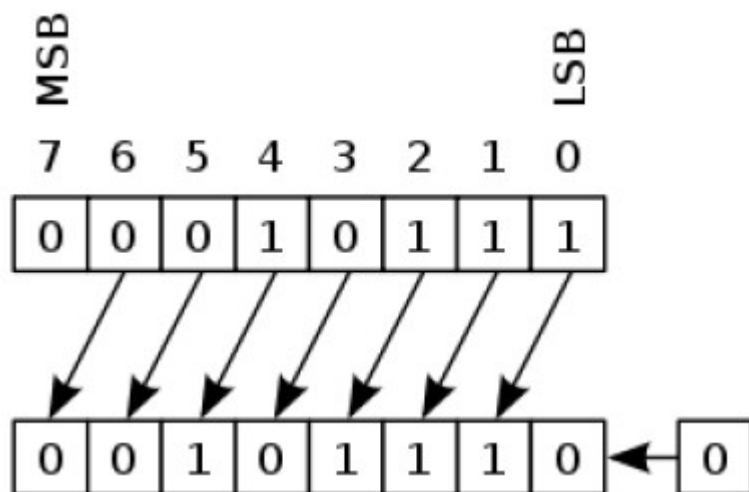
- Decimālā sistēma
255, 122, 84
- Heksadecimālā sistēma
0xff, 0x7A = 0x7a, 0x54
- Binārā sistēma
0b11111111, 0b01111010, 0b01010100

Bitu logika

NOT		OR			XOR			AND		
0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	1	1	0	1	1	0	0
		0	1	1	0	1	1	0	1	0
		1	1	1	1	1	0	1	1	1
~					^			&		

- $212 \text{ AND } (12 \text{ XOR } 122) \square 212 \text{ AND } 118 \square 84$
- $11010100 \text{ AND } (00001100 \text{ XOR } 01111010) \square ?$

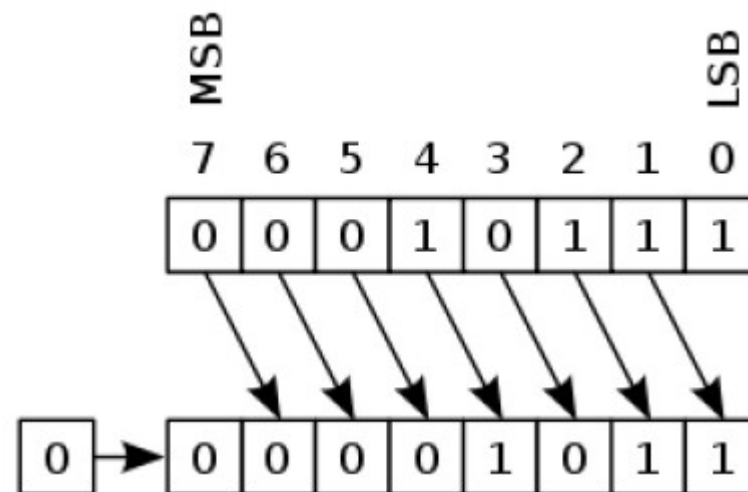
Bitu nobīdes operācijas



Bitu nobīde pa kreisi

$01111010 \ll 1 = ?$

$01111010 \ll 5 = ?$

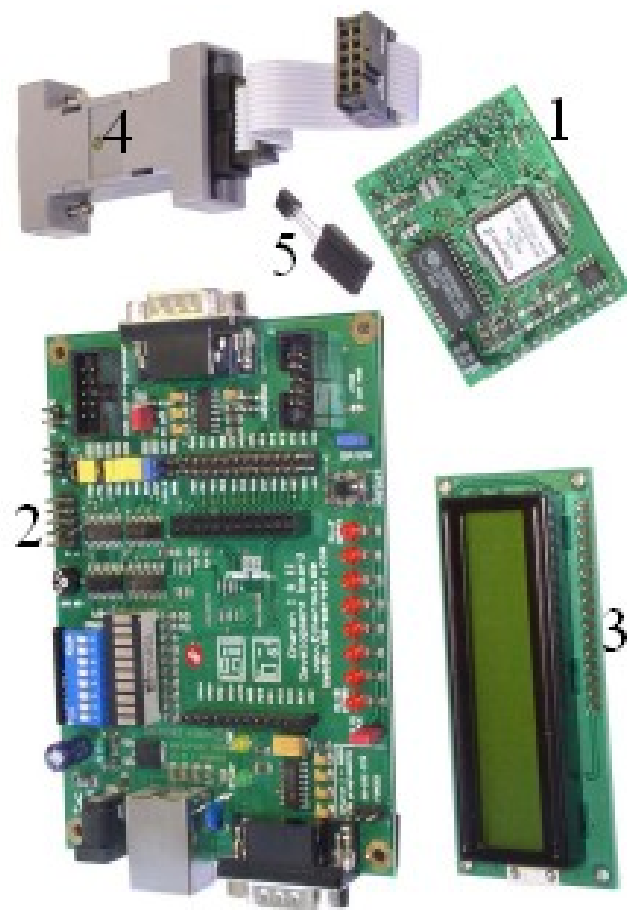


Bitu nobīde pa labi

$01111010 \gg 1 = ?$

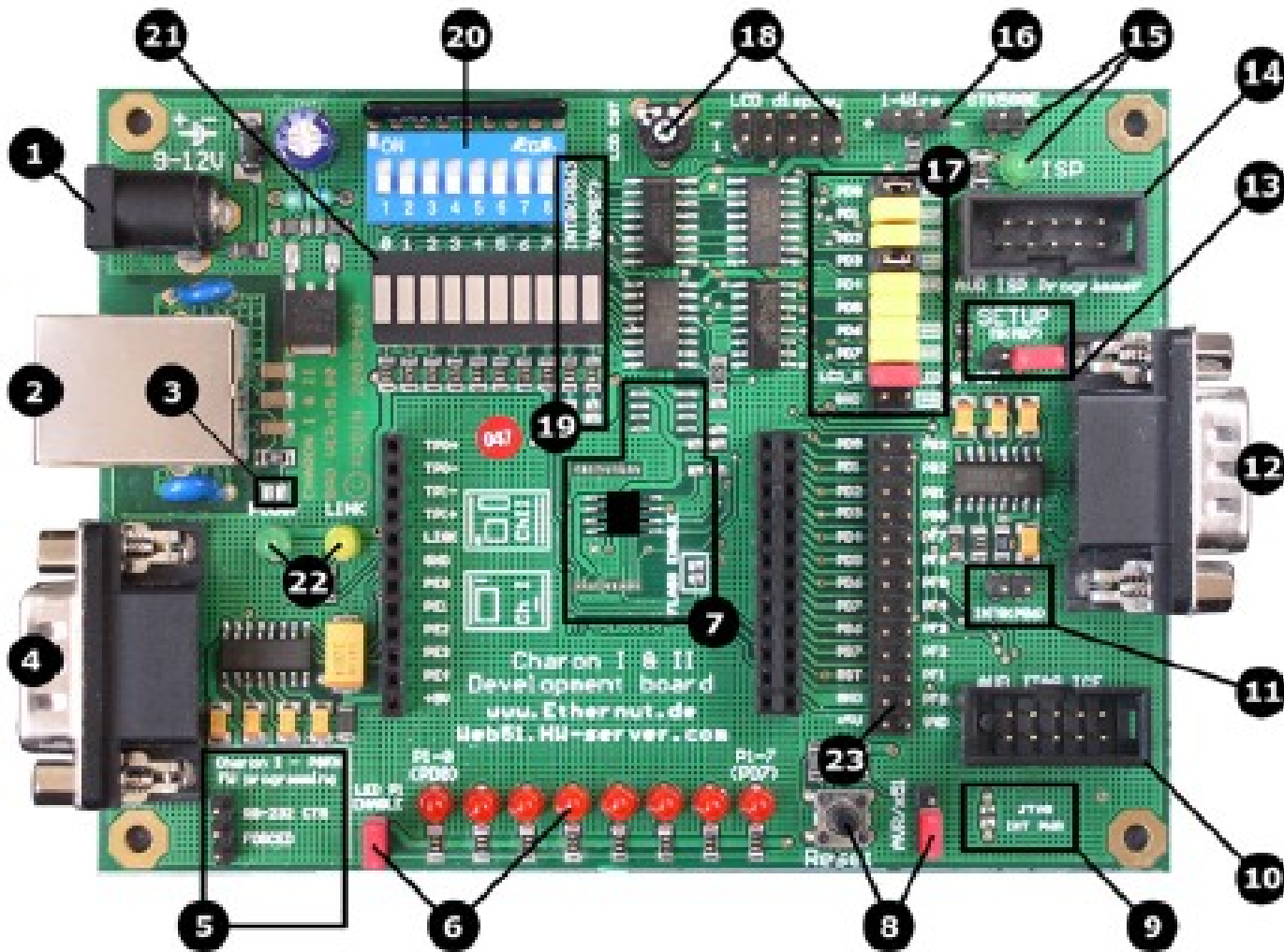
$01111010 \gg 3 = ?$

Charon II



1. Charon II moduļa.
2. Izstrādes plates.
3. LCD displeja.
4. HW STK-500 ISP programmatora.
5. 1-Wire temperatūras sensora.

Charon II



ATmega128



<http://www.atmel.com> – datu lapa
(datasheet)

Flash	128 Kbaiti
Pinu skaits	64 pini (7 porti)
Frekvence	14.7456 MHz
CPU	8-bit
I/O	53
Programmēšanas valoda	Assembler vai C

Programmas piemērs



```
1
2  /***** Standarta C un speciało AVR biblioteeku ieklaushana*****/
3  #include <avr/io.h>
4  #include <avr/iom128.h>
5  #include <avr/interrupt.h>
6  #include <avr/signal.h>
7  #include <math.h>
8  #include <stdlib.h>
9  #include <avr/delay.h>
10 #include <stdint.h>
11
12 #include <stdio.h>
13 /*****
```

- Bibliotēku pievienošana programmai

Programmas piemērs (II)



```
void port_init(void)
{
    DDRA  = 0x00; //visas porta A linijas uz IEvadi
    DDRB  = 0x00; //visas porta B linijas uz IEvadi
    DDRC  = 0x00; //visas porta C linijas uz IEvadi
    DDRD  = 0xFF; //visas porta D linijas uz IZvadi
    DDRE  = 0x00; //visas porta E linijas uz IEvadi
    DDRF  = 0x00; //visas porta F linijas uz IEvadi
    DDRG  = 0x00; //visas porta G linijas uz IEvadi

    PORTA = 0x00; //porta A atsienoshie rezistori pret +Vcc NETiek izmantoti
    PORTB = 0x00; //porta B atsienoshie rezistori pret +Vcc NETiek izmantoti
    PORTC = 0x00; //porta C atsienoshie rezistori pret +Vcc NETiek izmantoti
    PORTD = 0x00; //porta D izejas liniju limenji uz 0
    PORTE = 0x00; //porta E atsienoshie rezistori pret +Vcc NETiek izmantoti
    PORTF = 0x00; //porta F atsienoshie rezistori pret +Vcc NETiek izmantoti
    PORTG = 0x00; //porta G atsienoshie rezistori pret +Vcc NETiek izmantoti

    return;
}
```

- Portu inicializācijas funkcija

Programmas piemērs (III)



```
void init_devices(void)
{
    cli(); //aizliedz visus partraukumus
    XDIV   = 0x00; //takts impulsu dalitajs NE tiek izmantots
    XMCRA  = 0x00; //arejo atminju NEizmanto
    MCUCR  = 0x00; //NE tiek izmantoti nekadi energiju tauposhi stavokli

    port_init(); //inicialize portus

    return;
}
```

- Mikrokontrollera inicializācijas funkcija

Programmas piemērs (IV)



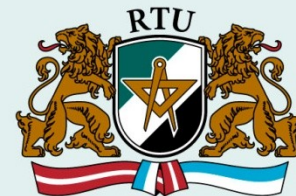
```
int main (void)
{
    unsigned char i,port_data;

    init_devices(); //Inicializejam kontrolleri

    while(1) //Muzigais cikls, lai programma nekad nebeigtos
    {
        port_data = 0b00000001;
        PORTD=~port_data;
    }
}
return 1;
}
```

- Galvenā funkcija

Kas tas tāds?



- DDRx – reģistra nosaukums, kurā ierakstot 1 vai 0 attiecīgo pinu uzstāda uz ievadi vai izvadi
- PORTx – reģistra nosaukums, kurā ierakstot 1 vai 0 uz attiecīgā pina izvada 1 vai 0 (piem., 0V vai +5V)
- PINx – reģistra nosaukums, kuru nolasot var iegūt 1, ja uz pina padots spriegums +5V, un 0, ja uz pina padots spriegums 0V









































































Uzdevums



- Iepazīties ar ATmega128 un Charon II;
- Nokompilēt piedāvātu izejas kodu un ierakstīt *.hex* programmfailu mikrokontrollera atmiņā. Pārbaudīt testa programmas korektu darbību;
- Pārrakstīt programmu tā, lai visas gaismas diodes ieslēgtos pēc kārtas, izveidot „skrejošas gaismas” efektu;

Uzdevums (II)



1.Iterācija								
2.Iterācija								
3.Iterācija								
4.Iterācija								
5.Iterācija								
6.Iterācija								
7.Iterācija								
8.Iterācija								
9.Iterācija								
n.iterācija								...