

# Programming i C

Programming af microcontroller i C  
(4 af 4)

12. april 2007

*Mads Pedersen, OZ6HR*  
*mads@oz6hr.dk*

# Plan i dag



- Afrunding af OZ3VB's program
- Fra "almindelig C" til "microcontroller C"
  - Lighederne
    - Struktur, løkker, if/else, variabler, ...
  - Forskelle
    - Input, output, registre, ...
- Eksempler
  - MSP430 (LED, indbygget temperatur-sensor)
  - AVR (seriel kommunikation)
  - PIC (LM75 temperatur-sensor)

# OZ3VB's program



- Kan findes på klubbens hjemmeside
- Debug programmet
  - Sæt "breakpoint" ved at klikke med musen på en linje i den sorte margin (eller Ctrl + F5)
  - Vælg Debug → Debug (F8)
  - Step over hver linje (F7)
  - Vælg en variabel → Højreklik → Add Watch

# "Almindelig" C til "microcontroller C" (1)

- Ikke den store forskel
- Strukturen (layout) er stadig den samme

```
#include "msp430x20x3.h"

void main(void)
{
    WDTCTL = WDTPW + WDTHOLD;           // Stop watchdog timer
    P1DIR |= 0x01;                       // Set P1.0 to output direction

    while(1 == 1)
    {
        volatile unsigned int i;        // volatile to prevent optimization

        P1OUT ^= 0x01;                   // Toggle P1.0 using exclusive-OR

        i = 10000;                        // SW Delay
        while(i != 0)
            i--;
    }
}
```

# "Almindelig" C til "microcontroller C" (2)

- Lighederne:
  - Main-funktionen
  - Løkker
    - *for*
    - *while*
  - Conditions
    - *if*
    - *if / else*
    - *if / else if / else*
  - Variabler
    - *int, short, long, float, double, char*

# "Almindelig" C til "microcontroller C" (3)

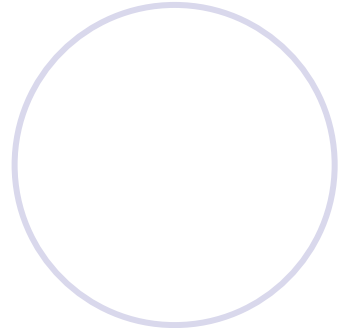
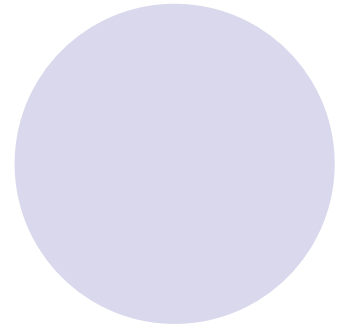
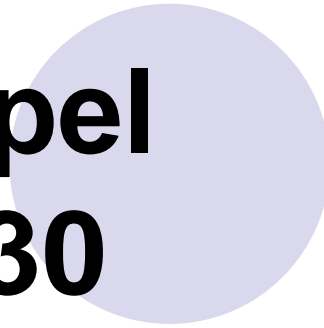
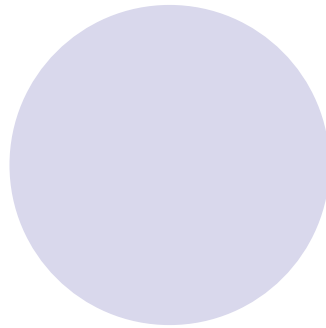
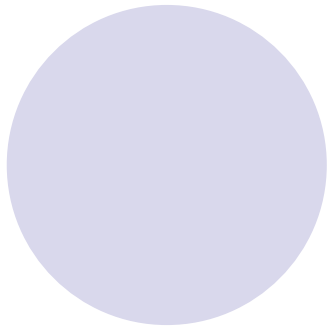
- Forskellene

- Input (før *scanf*)
- Output (før *printf*)
- Giver ikke længere mening, for hvordan skal man indtaste data og hvor skal man præsentere data?

- I stedet

- Registre kan bruges som input (f.eks. fra ADC eller temperatur-sensor)
- Display (LCD) eller seriel kommunikation kan bruges som output

# **Eksempel MSP430**



# MSP430 (1)

- OZ2JN John introducerede Texas Instruments MSP430 (Ultra low-power). Præsentationen ligger på [www.oz6hr.dk](http://www.oz6hr.dk)
- Eksperimenteret med "Developer Tool"
  - [http://www.ti-estore.com/Merchant2/merchant.mvc?Screen=PROD&Product\\_Code=EZ430-F2013](http://www.ti-estore.com/Merchant2/merchant.mvc?Screen=PROD&Product_Code=EZ430-F2013)
- Hurtigt og simpelt at komme i gang
- Både C og Assembler
- CD med software
  - "IAR Embedded Workbench IDE"



# MSP430 (2)

- Brochure og User's Guide

- [http://focus.ti.com/mcu/docs/mcuprooverview.tsp?sectionId=95&tabId=140&familyId=342&DCMP=MCU\\_other&HQS=Other+IL+msp430](http://focus.ti.com/mcu/docs/mcuprooverview.tsp?sectionId=95&tabId=140&familyId=342&DCMP=MCU_other&HQS=Other+IL+msp430)

- Assembler-instruktioner i kapitel "RISC 16-Bit CPU"

# MSP430 (3): LED-test

```
#include "msp430x20x3.h"

void main(void)
{
    WDTCTL = WDTPW + WDTCTL;           // Stop watchdog timer
    P1DIR |= 0x01;                     // Set P1.0 to output direction

    while(1 == 1)
    {
        volatile unsigned int i;      // volatile to prevent optimization

        P1OUT ^= 0x01;                 // Toggle P1.0 using exclusive-OR

        i = 10000;                      // SW Delay
        while(i != 0)
            i--;
    }
}
```

- I include-filen "msp430x20x3.h" defineres adresser, konstanter mv. for netop denne  $\mu$ C
- Info om registre kan findes i User's Guide
- Ellers standard C

# MSP430 (4): LED-test

- Vi kører programmet fra "IAR Embedded Workbench IDE"
- Project → Options
  - Target (MSP430F2013)
  - Debugger
    - Simulator
    - FET Debugger (FET = Flash Emulation Tool)
- Debugger/simulator meget nyttig under udvikling!

# MSP430 (5): Temperatur-sensor

```
#include <msp430x20x3.h>
#include <stdio.h>

#define ADCDeltaOn 31 // ~0.5 Deg C delta
static unsigned int LastADCVal; // holds ADC temperature result

void main(void)
{
    BCSCTL2 |= DIVS_3; // SMCLK/8
    WDTCTL = WDT_MDLY_32; // WDT Timer interval
    IE1 |= WDTIE; // Enable WDT interrupt
    P1DIR |= 0x01; // P1.0 to output direction
    SD16CTL = SD16REFON +SD16SSEL_1; // 1.2V ref, SMCLK
    SD16INCTL0 = SD16INCH_6; // A6+/-
    SD16CCTL0 = SD16SNGL + SD16IE ; // Single conv, interrupt

    _BIS_SR(LPM0_bits + GIE); // Enter LPM0 with interrupt
}

// SD16_A interrupt service routine
#pragma vector=SD16_VECTOR
__interrupt void SD16ISR(void)
{
    // Make the compare (the ADC value is in SD16MEM0)
    if (SD16MEM0 > LastADCVal + ADCDeltaOn)
        P1OUT |= 0x01; // LED on
    else
        P1OUT &= ~0x01; // LED off
    LastADCVal = SD16MEM0; // Store value
}

// Watchdog Timer interrupt service routine
#pragma vector=WDT_VECTOR
__interrupt void watchdog_timer(void)
{
    SD16CCTL0 |= SD16SC; // Start SD16 conversion
}
```

Se User's Guide  
for info om registre

**Nye ting i dette  
program:**

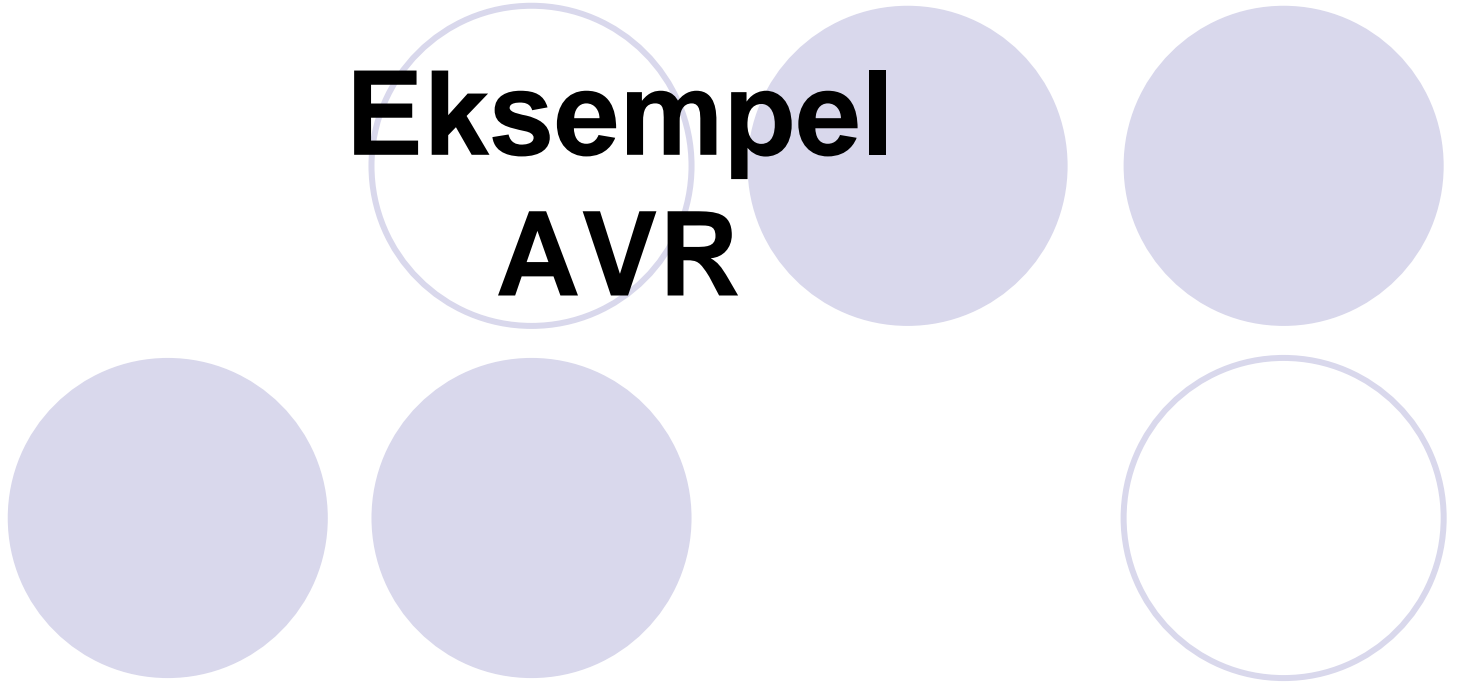
Kaldes automatisk,  
når ADC'en har  
foretaget en  
konvertering

En Watchdog Timer  
bruges normalt til at  
resette et system,  
hvis alt andet kikser...  
Her starter WDT  
ADC'en

# MSP430 (6): Temperatur-sensor

- Vi kører programmet fra "IAR Embedded Workbench IDE"
- Sæt breakpoint ved at dobbeltklikke i margin
- Programmet startes i "FET Debugger"
- Tryk F5 for at fortsætte til breakpoint
- Højreklik på variabel → "Add to Watch"
  - F.eks. LastADCVal og SD16MEM0

# Eksempel AVR



# AVR (1)



- Flere af klubbens medlemmer har selv bygget et "developer board" med en AVR microcontroller (AT90S2313)
- Datasheet:
  - [http://www.atmel.com/dyn/products/product\\_card.asp?family\\_id=607&family\\_name=AVR+8%2DBit+RISC+&part\\_id=1993](http://www.atmel.com/dyn/products/product_card.asp?family_id=607&family_name=AVR+8%2DBit+RISC+&part_id=1993)
- Vi har primært programmeret i BASCOM, som et et Basic-lignende sprog
- Det er også et fint højniveau-sprog, men ikke nær så udbredt og standardiseret et sprog som C
- Man kan sagtens programmere AVR i C
  - F.eks. WinAVR (<http://sourceforge.net/projects/winavr>)

# AVR (1): Serial kommunikation

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

/* Prototypes */
void InitUART( unsigned char baudrate );
unsigned char ReceiveByte( void );
void TransmitByte( unsigned char data );

/* Main - a simple test program */
int main( void )
{
    InitUART( 11 ); /* Set the baudrate to 19,200 bps using a 3.6864MHz crystal */

    for(;;)          /* Forever */
    {
        TransmitByte( ReceiveByte() ); /* Echo the received character */
    }

    return 0;
}

/* Initialize UART */
void InitUART( unsigned char baudrate )
{
    UBRR = baudrate; /* Set the baud rate */
    UCR = ( 1<<RXEN ) | ( 1<<TXEN ); /* Enable UART receiver and transmitter */
}

/* Read and write functions */
unsigned char ReceiveByte( void )
{
    while ( !(USR & (1<<RXC)) ) /* Wait for incoming data */
        ; /* Return the data */
    return UDR;
}

void TransmitByte( unsigned char data )
{
    while ( !(USR & (1<<UDRE)) )
        ; /* Wait for empty transmit buffer */
    UDR = data; /* Start transmission */
}
```

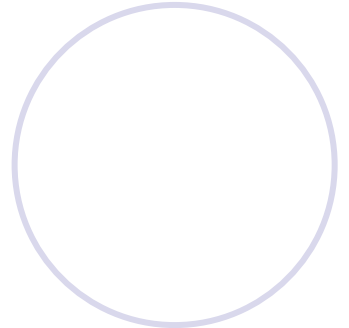
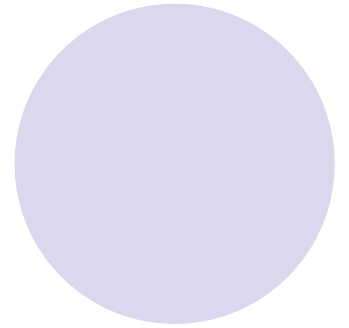
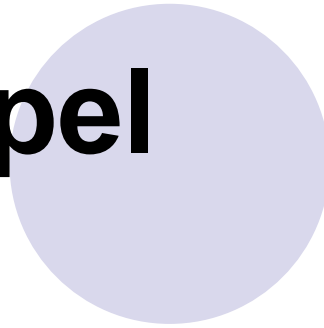
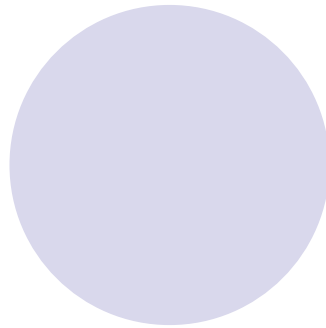
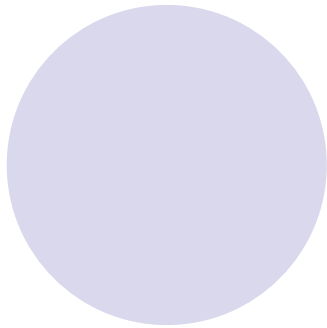
Alternativ til

*while(1 == 1)*

Funktioner



# Eksempel PIC

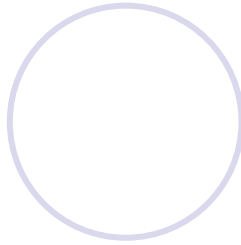


# PIC (1)



- Vi har arbejdet med PIC tidligere
- PIC kan også sagtens programmeres i C, det kræver blot at der installeres en udvidelse (C compiler) til det normale "MPLAB"

# PIC (2)



LM75 er den samme temperatur-sensor, som blev brugt i AVR-projektet

Der kan læses mere om I2C i præsentationen af AVR på [www.oz6hr.dk](http://www.oz6hr.dk)

```
#include <16f877.h>
#define LM75_SDA PIN_C4
#define LM75_SCL PIN_C3
char result; // The result where the temperature value goes

void main()
{
    LM75_init();
    while (1)
    {
        result = LM75_read_temperature();
        printf("Temperature = %d", result); // Sent to remote debugger
        delay_ms(1000);
    }
}

void LM75_init(void)
{
    output_float(LM75_SDA);
    output_float(LM75_SCL);

    i2c_start();
    i2c_write(I2C_WRITE_COMMAND);
    i2c_write(LM75_CONFIG_REG_ADDR);
    i2c_write(LM75_CONFIG_VALUE);
    i2c_stop();
}

char LM75_read_temperature(void)
{
    char msb;
    char lsb;

    i2c_start();
    i2c_write(I2C_WRITE_COMMAND); //0b10011110 LM75_I2C_WRITE_ADDR
    i2c_write(0x00);
    i2c_start();
    i2c_write(I2C_READ_COMMAND); //0b10011111 LM75_I2C_READ_ADDR
    msb = i2c_read();
    lsb = i2c_read(0);
    i2c_stop();

    return msb; // Ignore lsb
}
```



# Slut!

- Det var det!
- **Spørgsmål?**
- Jeg besvarer naturligvis spørgsmål senere
  - Kodetekniske spørgsmål
  - Værktøjer
  - Microcontroller-relateret (platforme, værktøjer, ...)
- Alt ligger tilgængeligt på [www.oz6hr.dk](http://www.oz6hr.dk)
- Afsat 4 aftener i næste kvartal til microcontroller-projekter