Programmering i C

Intro og grundlæggende C

5. marts 2007

Mads Pedersen, OZ6HR

mads@oz6hr.dk

Plan for kurset

- Ma. 5/3: Formål, intro, grundlæggende
- Ma. 19/3: Videre, sprogkonstruktioner
- Ma. 2/4: Praksis (opgave eller vælg selv)
- To. 12/4: Microcontroller-specifikt

Plan i dag

- Formål
- Introduktion
- Værktøjet Dev-C++
- Kodeprocessen
- Layout i et C-program
- ”Hello World”
  – det berømte begyndereksempel!
- Grundlæggende
  - Datatyper
  - Variabler
  - Input/output
  - Afsluttende eksempel: Konvertering fra ºF til ºC

Formål

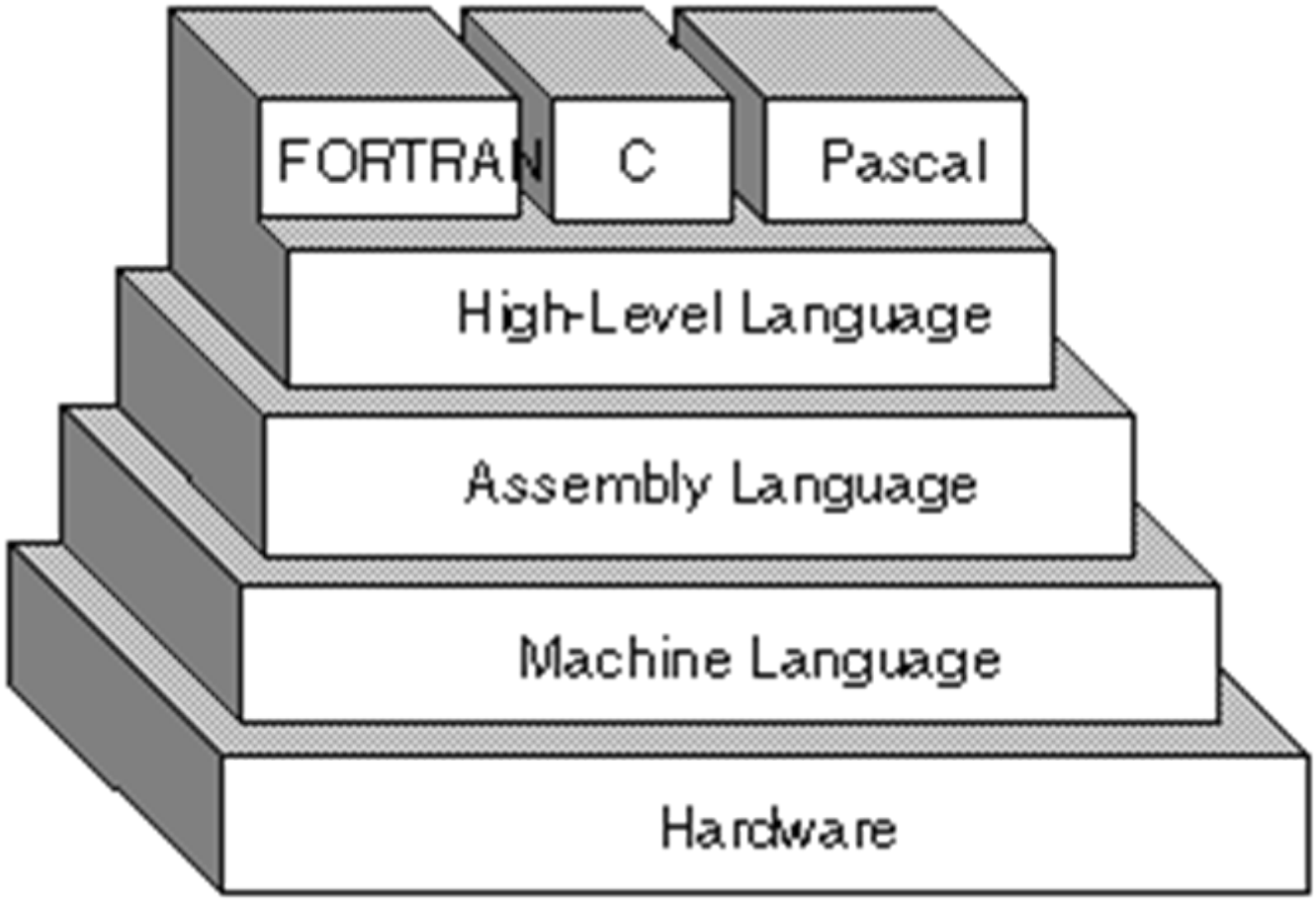
- Vi har haft gang i flere microcontrollere
- Programmeret i assembler, basic og C
- Ønske om mere C
  - Almindelige Windows-programmer
  - Microcontroller

Forbehold…

- Det tager LANG tid at blive en god programmør
- Vi kan kun introducere en lillebitte brøkdel
- Der findes masser af gode tutorials på nettet

- Værktøj på dansk, men alt kode er engelsk (engelske *keywords*)
  - det kommer man ikke uden om ☹

Introduktion til C



C i forhold til andre sprog

- C-programmer kan ikke eksekveres direkte som visse andre sprog (*Basic, javascript, PHP, ASP: VBscript* m.fl.)
- Skal oversættes til en .exe fil
- Andre sprog skal køres i en fortolker (*Java, C#* m.fl.), men C (og C++) kan altså afvikles direkte fra .exe filen.

- C++ er en objektorienteret modernisering af C.

Hvor bruges C i dag? (1)

- Operativsystemer
- Sprog-kompilere
- Sprog-fortolkere
- Teksteditorer
- Drivere (f.eks. til printere og netværk)
- Databaser
- Moderne applikationer
- … Alt muligt andet

- Generelt:
  - Næsten lige så hurtigt som Assembler-kode og **meget** mere effektivt at kode i

Hvor bruges C i dag? (2)

- Mange nyere højniveau-sprog som C++, C# og Java har elementer fra C
- Fint med kendskab til C, når man skal lære disse nye sprog
- Nye Windows-programmer vil typisk blive skrevet i et af disse nyere højniveau-sprog

- Microcontrollere (PIC, AVR, 8051, MSP430 mv.) programmeres stadig primært i C

Historie

- 1972: Dennis Ritchie ved Bell Labs skriver C
- 1978: *C Programming Language* publiceres af Kernighan & Ritchie
- 1983-88: "*ANSI C*" standard af *American National Standards Institute*

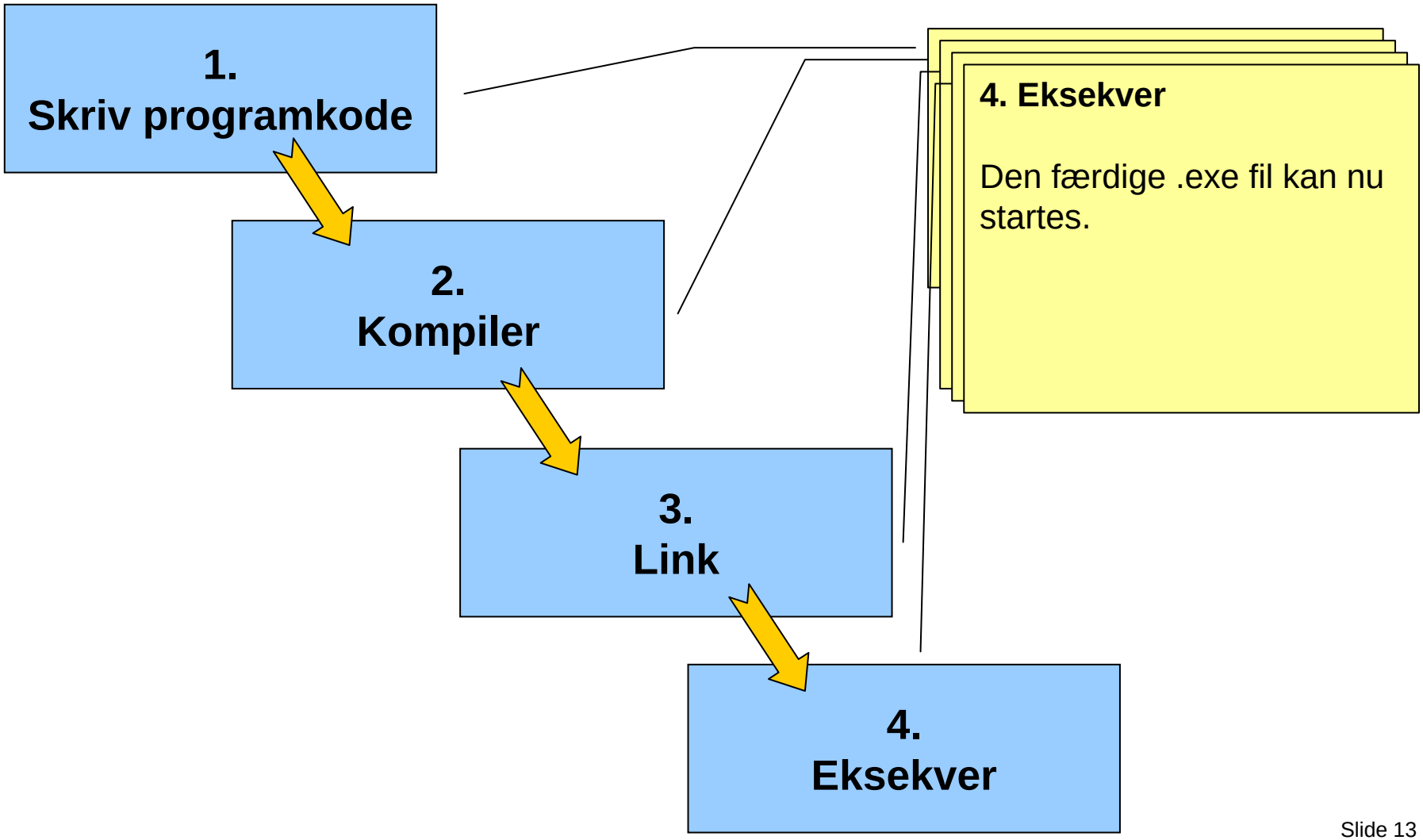
Hvad skal vi bruge?

- C compiler
  - Omsætter fra C-kode til noget som computeren kan forstå
- Mange muligheder
  - Kommandoprompt
    - DOS
    - Linux-lignende miljø (Cygwin)
  - **I**ntegrated **D**evelopment **E**nvironment (IDE)
    - Grafisk brugerflade
    - Hjælpefunktioner
  - Vi vælger **Dev-C++**
    - http://www.bloodshed.net/dev/devcpp.html
    - Både C og C++
    - Bl.a. DANSK!

Intro til værktøjet Dev-C++

- Installation
- Opsætninger
- Projekter – filer
- Kompiler, kør, debug
- Short-cuts gode!

Kode-proces


1.
Skriv programkode
2.
Kompiler
3.
Link
4.
Eksekver
4. Eksekver
Den færdige .exe fil kan nu startes.

Layout i en C-fil

```
preprocessor directives
global declarations

main()
{
   local variables to function main;
   statements associated with function main;
}

f1()
{
   local variables to function 1;
   statements associated with function 1;
}

f2()
{
   local variables to function f2;
   statements associated with function 2;

}
.
.
.
```

- ( ) bruges I forbindelse med funktioner
- { } bruges til at afgrænse en funktion
- ; (semikolon) bruges til at terminere C statements
- Preprocessor directives:
  F.eks. inkludering af eksterne funktioner, som `#include <stdio.h>`

Første C-program

- Det populære "Hello World" eksempel

```
#include <stdio.h>

main()

{
   /* Print to the screen */
   printf("Hello World\n");
}
```

- Inkludér andre biblioteker
- Definition af main-funktionen, som starter programmet
- Kommandoer indkapsles i {…}
- Kommentarer skrives i /* … */
- Kald af standard-funktionen printf, som printer til skærmen (\n er et linjeskift)

Datatyper

- Basale datatyper:
  - int - integer: Heltal, f.eks. 42
  - short - short integer
  - long - long integer
  - float - floating point tal: Kommatal, f.eks. 42.5
  - double - double-precision floating point
  - char - character: Enkelt karakter, f.eks. 'a'
- Størrelsen er maskin- og kompiler-afhængig. Normalt er en int 16 eller 32 bit (dvs. 2 eller 4 bytes)

Variabler og aritmetik

```
type variabelnavn;

int a;                  /* Variabel kaldet a af typen integer */
a = 10;                 /* Tildel variabel a værdien 10 */
a = a + 5;              /* Tildel variabel a værdien af a + 5 */

double sum;             /* Variabel kaldet sum af typen double */
sum = 12.50             /* Tildel variable sum værdien 12.50 */

char c;                 /* Variabel kaldet c af typen character */
c = 'A';                /* Tildel variabel c karakteren A */

int a = 10;             /* Initialiser variabel med værdi i starten */
int a, b, c;            /* 3 variabler af typen integer */
```

- Bør være lower-case
- Skal begynde med bogstav eller _ (ikke tal)

```
int a;
int b;
int resultat;

a = 10;
b = 3;

resultat = a + b;       /* Addition, resultat = 13 */
resultat = a – b;       /* Subtraktion, resultat = 7 */
resultat = a * b;       /* Multiplikation, resultat = 30 */
resultat = a / b;       /* Division, resultat = 3, ikke 3.333333 (heltal) */
```

Input og output, *scanf* og *printf* (1)

- Input *scanf* og output *printf* er en del af ANSI-specifikationen og ligger i biblioteket *stdio.h*

```
printf(string, variable, variable, variable ...)    /* Generel form */

printf("Hello World");              /* printf uden variabler */

int total = 42;
printf("Total = %d", total);        /* Print signed integer (%d) */
```

```
scanf(control string, variable, variable,...)     /* Generel form */

int a;
scanf("%d", &a)         /* Input lægges som signed integer (%d)
                           i variabel a */
```

% Format Specifiers

Format	Datatype	Display
%c	char	single character
%d (%i)	int	signed integer
%e (%E)	float or double	exponential format
%f	float or double	signed decimal
%g (%G)	float or double	use %f or %e as required
%o	int	unsigned octal value
%p	pointer	address stored in pointer
%s	array of char	sequence of characters
%u	int	unsigned decimal
%x (%X)	int	unsigned hex value

Input og output, *scanf* og *printf* (2)

```
#include <stdio.h>

main()
{
   int a, b, c;

   printf("\nThe first number is ");
   scanf("%d", &a);

   printf("The second number is ");
   scanf("%d", &b);

   c = a + b;

   printf("The answer is %d \n", c);
}
```

Afsluttende eksempel

- Konvertering fra Fahrenheit til Celsius

$$°C = \frac{5}{9} \cdot (°F - 32)$$

```
/* Konverterer fra Fahrenheit til Celsius
   vha. formlen °C = (5/9)(°F-32) */

#include <stdio.h>

main()
{
    int fahr, celsius;

    printf("\nSkriv Fahrenheit: ");
    scanf("%d", &fahr);

    celsius = 5 * (fahr-32) / 9;

    printf("%d F => %d C\n", fahr, celsius);
}
```

Brush-up

- Introduktion (historie, hvor bruges C)
- Værktøjet Dev-C++ (gratis og bl.a. på dansk)
- Kodeprocessen
  (Kode → Kompiler → Link → Eksekver)
- Layout i en C-fil (*#include*, *main*, …)
- Datatyper (*int, short, long, float, double, char*)
- Variabler og aritmetik
- Input og output (*scanf* og *printf*)

Links (danske)

- Dansk introduktion til C
  - http://our-site.dk/service/kurser/C/C.php
- Kursus på Aalborg Universitet
  - http://www.cs.auc.dk/~normark/c-prog-04/html/notes/index.html
- PDF til ovenstående kursus
  - http://www.cs.auc.dk/~normark/c-prog-04/html/notes/leno-print-page.html

Links

- Kursus på University of Washington
  - http://www.eskimo.com/~scs/cclass/cclass.html
- theForger's Win32 API Tutorial
  - http://www.winprog.org/tutorial/
- Cprogramming.com
  - http://www.cprogramming.com/
  - http://www.cprogramming.com/tutorial.html#ctutorial
  - http://www.cprogramming.com/tutorial.html#advanced
- FoosYerDoos (brugergrænseflader i Win32 API)
  - http://www.foosyerdoos.fsnet.co.uk/

Næste gang

- Videre med C
  - Kontrolløkker (*while*, *for*, …)
  - Conditional Execution (*if, if/else*)
  - Funktioner
  - Så er vi faktisk godt i gang
    ➔ vi kan lave næsten alle programmer

- Næste gang igen:
  - Praksis…
    - Kom gerne med egen computer
    - Opgave eller vælg selv