

DST 1

Nicholas Wickström

IDE, Högskolan i Halmstad

2009

Översikt

- 1 **Introduktion**
 - Hårdvarunära programmering
 - C grunder

Outline

- 1 **Introduktion**
 - Hårdvarunära programmering
 - C grunder

Hårdvarunära programmering

- Vad är Hårdvara? (Datorsystemmodell; processor m. periferi, IO, Minne)
- Typiskt för hårdvarunära programmering (datatablad, register, datastrukturer,)
- Språk - Assembler, C, C++??, java??
- Verktyg (utvecklingsmiljöer, debugger, minne, kodkontroll)

Hårdvarunära programmering

- Vad är Hårdvara? (Datorsystemmodell; processor m. periferi, IO, Minne)
- Typiskt för hårdvarunära programmering (datatablad, register, datastrukturer,)
- Språk - Assembler, C, C++??, java??
- Verktyg (utvecklingsmiljöer, debugger, minne, kodkontroll)

Hårdvarunära programmering

- Vad är Hårdvara? (Datorsystemmodell; processor m. periferi, IO, Minne)
- Typiskt för hårdvarunära programmering (datatablad, register, datastrukturer,)
- Språk - Assembler, C, C++??, java??
- Verktyg (utvecklingsmiljöer, debugger, minne, kodkontroll)

Hårdvarunära programmering

- Vad är Hårdvara? (Datorsystemmodell; processor m. periferi, IO, Minne)
- Typiskt för hårdvarunära programmering (datablad, register, datastrukturer,)
- Språk - Assembler, C, C++??, java??
- Verktyg (utvecklingsmiljöer, debugger, minne, kodkontroll)

Hårdvarunära programmering

- Ofta inget OS, problem man kan bara lite på sig själv och det gäller att göra rätt.
- Prestanda, Minnesbegränsningar
- Det blir lätt krångligt med hårdvara - ny dimension

Hårdvarunära programmering

- Ofta inget OS, problem man kan bara lite på sig själv och det gäller att göra rätt.
- Prestanda, Minnesbegränsningar
- Det blir lätt krångligt med hårdvara - ny dimension

Hårdvarunära programmering

- Ofta inget OS, problem man kan bara lite på sig själv och det gäller att göra rätt.
- Prestanda, Minnesbegränsningar
- Det blir lätt krångligt med hårdvara - ny dimension

C grunder

- Struktur, reserverade ord
- Variabler, typer, typomvandling
- Lagringsklasser *auto*, *register*, *volatile*, *static*, *extern*

C grunder

- Struktur, reserverade ord
- Variabler, typer, typomvandling
- Lagringsklasser *auto*, *register*, *volatile*, *static*, *extern*

C grunder

- Struktur, reserverade ord
- Variabler, typer, typomvandling
- Lagringsklasser *auto*, *register*, *volatile*, *static*, *extern*

enkel.c

```
/* enkel.c */
#include <stdlib.h> /* Common used functions in here */
#include "enkelsum.h" /* CalculateSum defined here */

void main(void)
{
    int nCounter, n;
    float fSum, f;
    ...
    fSum = CalculateSum(nArg1, fArg2);
    ...
    return;
}
```

enkelsum.h - enkelsum.c

```
/* enkelsum.h */  
extern float CalculateSum(int nArg1, float fArg2);  
  
/* enkelsum.c */  
float CalculateSum(int nArg1, float fArg2)  
{  
    return( (float)nArg1+fArg2 )  
}
```

Reserverade ord

**auto, break, case, char, const, continue, default, do, double,
else, enum, extern, float, for, goto, if, int, long, register,
return, short, signed, sizeof, static, struct, switch, typedef,
union, unsigned, void, volatile, while**

Variabler - begrepp

- Lokala, Globala
- Livstid
- Minnesutrymme - Maskinberoende (*limits.h*, *float.h*)
- Typ
- Deklaration/Definition/Initiering

Typer

- Grundtyper - **heltal, flyttal**
- Typer baserade på grundtyperna - **vektorer, poster, unioner, uppräknande**
- Pekare till andra typer
- Specialtyp - **void**
- Heltaltstyper - *signed/unsigned* (**int, short, char, long**)
- Flyttaltstyper - **float, double, long double**
- Konstanter **const**

Typer (forts.)

```
int nVect[100]; /* Vektor av heltal */
struct car_spec {
    int nCyls;
    float fHorsePower;
} rSaab, rVolvo;
/* Definition av rSaab, rVolvo med
   typ: struct car_spec */
rSaab.nCyls = 4; /* Punktoperatorn */

enum months {JAN = 1, FEB, MAR, APR, MAY, JUN, JUL,
             AUG, SEP, OCT, NOW, DEC};
/* Uppräknande */
```

Typomvandling

```
int    nApples;
float  fParts;
float  fResult;

fResult = nApples/fParts; /* Implicit */
/* Resultatet blir av typen float */

/* Notera! 5/2 blir 2 medan 5.0/2.0 blir 2.5 */

fResult = (float) nApples; /* Explicit */
```

Lagringsklasser

```
int      nApples;
/* Egentligen auto int nApples,
   minne lagras automatiskt */

register int nCnt;
/* Ex. loop räknare skall lagras i register */

volatile int nInterrupt;
/* Får ej optimeras bort eller läggas i register */

static int nLocal; /* Intern global variabel */
extern int nGlobal; /* En definition,
flera deklARATIONER */
```