

CHALMERS Maskinorienterad programmering

Pekare till C-funktioner på fast adress i minnet

```
void func(void) { /* funktion utan parameter och returvärde */ }
void * func(void) { /* funktion utan parameter, returnerar pekare */ }
void (*func)(void) { /* pekare till funktion utan parameter och returvärde */ }
```

EXEMPEL:
En parameterlös subrutin `portinit` för att initiera hårdvaran i MC407 finns med ingång på adress `0x08000201` i minnet. Visa hur denna kan anropas:
a) från ett C-program
b) i assemblerpråk
Vi löser på tavlan...

Assemblerprogrammering - fördjupning 2

a)

```
typedef void (*simplefunc)(void);
/* obs Thumb-bit i adressbit 0 */
#define portinit ((simplefunc) 0x08000201 )
void xxx( void )
{
    ...
    portinit();
    ...
}

b)

...
LDR R0,=0x08000201 @ obs Thumb-bit i A0...
BLX R0
```

Alt 2: R7 är "framepointer"
sub:
@ parametrar finns i register,
@ spara dessa på stacken

```
PUSH {LR,R7}
SUB SP,SP,#16
MOV R7,SP
STR R0,[R7,#12]
STR R1,[R7,#8]
STR R2,[R7,#4]
STR R3,[R7]
...
MOV SP,R7
ADD SP,SP,#16
POP {PC,LR}
```

CHALMERS Maskinorienterad programmering

Parameteröverföring via register

Konventionerna säger att vi använder register R0,R1,R2,R3, (i denna ordning) för parametrar som skickas till en subrutin. Övriga parametrar läggs på stacken.

Exempel:
int a,b,c,d;
Följande funktionsanrop görs:
sub(a,b,c,d);

Tänkbar komplikation: "Registerspill"

Dvs. register behövs i den anropade subrutinen.
Visa hur en prolog skapar en aktiveringspost med kopior av parametrarna. Dvs. Skapa "tillfälliga variabler" – och använd stacken för temporär lagring.
Vi löser på tavlan

Assemblerprogrammering - fördjupning 10

Alt 1: SP är "framepointer"
sub:
@ parametrar finns i register,
@ spara dessa på stacken

```
PUSH {LR}
SUB SP,SP,#20
STR R0,[SP,#12]
STR R1,[SP,#8]
STR R2,[SP,#4]
STR R3,[SP]
...
ADD SP,SP,#20
POP {PC}
```

Förklaring till instruktionerna LDMIA och STMIA


Maskinorienterad programmering

Returvärden via stack

Krävs typiskt då en subrutin ska returnera en komplett post, eller ett helt fält. Den anropande funktionen ska då först reservera utrymme för returvärdet. En pekare till detta utrymme läggs sedan som första parameter vid anropet:

Exempel:

```

typedef struct word COORD;
typedef struct word {
    int x,y,z;
} COORD;

COORD *start;

COORD sub( void )
{
    static COORD start;
    return start;
}

```

GCC genererar följande kod:

```

@ Reservad stack för returvärde
SUB    SP, SP, #16
MOV    R0, SP
BL     sub
--

sub:
PUSH    {R4,R7}
LDR     R2, =start
MOV     R4, R0
LDMIA   R2, {R0,R1,R2,R3}
STMIA   R4, {R0,R1,R2,R3}
MOV     R0, R4
POP     {R4,R7}
BX      LR

```

LDMIA: *Load Multiple Increment After*
 STMIA: *Store Multiple Increment After*

R2 = start

LDMI R2, {R0, R1, R2, R3}

->

(start) ->R0

(start +4) ->R1

(start +8) ->R2

(start +12) ->R3

R2 <- start + 16

R4 = dest

STMIA R4, {R0, R1, R2, R3}

->

(dest) <- R0

(dest +4) <- R1

(dest +8) <- R2

(dest +12) <- R3

R4 <- dest + 16