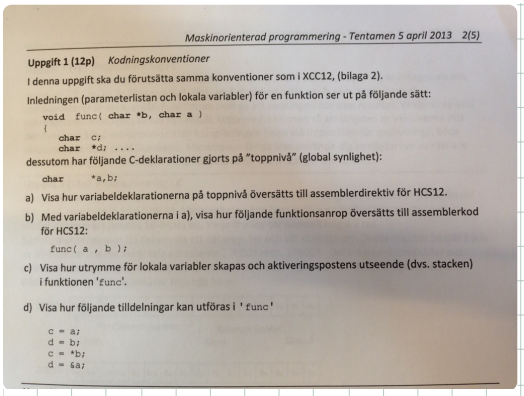


Maskinorienterad programmering - Tentamen 5 april 2013 2(5)

Uppgift 1 (12p) *Kodningskonventioner*

I denna uppgift ska du förutsätta samma konventioner som i XCC12, (bilaga 2).

Inledningen (parameterlistan och lokala variabler) för en funktion ser ut på följande sätt:

```
void  func( char *b, char a )
{
   char  c;
   char  *d; ....
```

dessutom har följande C-deklarationer gjorts på "toppnivå" (global synlighet):

```
char     *a,b;
```

a) Visa hur variabeldeklarationerna på toppnivå översätts till assemblerdirektiv för HCS12.

b) Med variabeldeklarationerna i a), visa hur följande funktionsanrop översätts till assemblerkod för HCS12:

```
func( a , b );
```

c) Visa hur utrymme för lokala variabler skapas och aktiveringspostens utseende (dvs. stacken) i funktionen 'func'.

d) Visa hur följande tilldelningar kan utföras i 'func'

```
c = a;
d = b;
c = *b;
d = &a;
```

a) Toppnivå: Globalt scope: char *a,b;

```
_a   RMB  2
_b   RMB  1
```

b) 3 saker att göra: 1. PUSHA argumenten till stacken med början till höger
2. Hoppa till subrutin
3. Återställ stackpekaren så att den pekar på returaddressen.

```
LDAB   _b
PSHB
LDD    _a
PSHD
JSR    _func
LEAS   3,SP
```

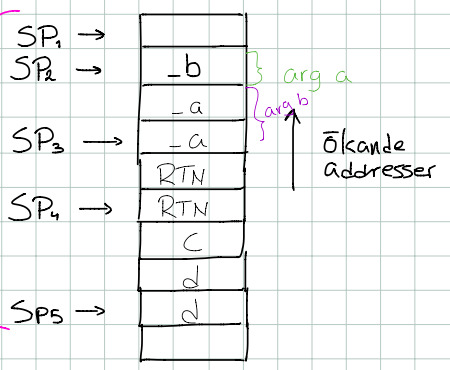
Eftersom B är för litet. Rita så du inte tappar dig.

c)

```
LEAS   -3,SP   ← prolog
```

För att veta hur SP kommer stå kan vi titta upp PSHB

Aktiveringsposten
Stackframe



d) _func:

```
LEAS   -3,SP
LDAB   7,SP
STAB   2,SP
LDD    5,SP
STD    0,SP
LDX    5,SP
LDAB   0,X
STAB   2,SP
STS    0,SP
LDD    0,SP
ADDD   #7
STD    0,SP
(LEAS  3,SP)
RTS
```

LEAX 7,SP
STX 0,SP

Exempeltenta 1.
Uppgift 4

Uppgift 4 (8p) *Programmering med pekare*

Skriv en C-funktion med följande deklaration:

```
char *xcpy(const char *q1, char *q2);
```

Funktionen skall kopiera den text `q1` pekar på till det utrymme som pekas ut av `q2`, men vid kopieringen skall alla *inledande och avslutande* s.k. *vita tecken* utelämnas. Med vita tecken menas mellanslag, tabulator och nyradstecken. Tänk på att det kan finnas vita tecken inne i den text som skall kopieras. Dessa tecken skall förstås tas med i kopieringen.

Funktionen `xcpy` skall som resultat ge en pekare till kopian av texten. Du får inte använda dig av någon av C:s standardfunktioner, utan du måste skriva all kod själv.

Tips. Leta först framifrån efter första icke vita tecken. Sök sedan upp slutet på texten och leta därifrån bakåt efter sista icke vita tecken. Kopiera sedan den mellanliggande texten till det utrymme som pekas ut av `q2`.

```
Char *xcpy(const char *q1, char *q2)
{
    Char *end, *kopiastart;

    while(*q1 == ' ' || *q1 == '\n' || *q1 == '\t')
    {
        q1++;
    }
    end = q1;
    while(*end)
    {
        end++;          end pekar på null
    }
    kopiastart = q2;
    end--;              end pekar på "en innan null"
    while(end >= q1 && (*end == ' ' || *end == '\t' || *end == '\n'))
    {
        end--;          q1 pekar på första, end på sista
    }
    while(q1 <= end)
    {
        *q2 = *q1;
        q1++;
        q2++;
    }
    *q2 = '\0';
    return kopiastart;
}
```

Vi får inte använda q1 för att skriva:
Minnet vi pekar på är skyddat!
`*q1 = k;` är INTE ok.

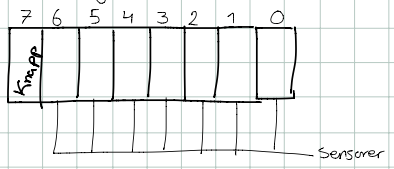
```
const char *q1;

char* const q3;
```

pekaren är konstant.
`*q3 = k` är okay,
`q3++;` är inte okay!

Övervakningslarm

DATR 0x500



STYRR 0x501

endast skrivbart

#define IRQ_VEC_ADR 0xFFE0

Void alarm_on(void)

Void alarm_off(void)