

Praktikumsversuch 1: Programmierung des NF300

Im ersten Praktikumsversuch sollen Sie einerseits das verwendete Mikrocomputersystem und den Umgang mit den Entwicklungs- und Test-Tools kennenlernen, andererseits die in Vorlesungen erlernten Eigenschaften eines Mikrocomputersystems (Adressierungsarten, Befehlssatz ..) an einem realen System nachvollziehen. Der verwendete Mikrocontroller 68332 verwendet eine CPU32, einen 32-Bit Prozessor von Motorola.

1. Vorbereitung:

Die Aufgaben sind in der zur Verfügung stehenden Zeit nur zu bewerkstelligen, wenn Sie gut vorbereitet sind. Dazu gehört, dass Sie die Programme bereits zuhause codiert und fehlerfrei übersetzt haben. Außerdem sollten Sie die Befehle des Monitors und Debuggers *BD32* beherrschen.

- 1.1 Analysieren Sie das folgende Assemblerprogramm „blink.asm“. und geben Sie an den angegebenen Testpunkten die Inhalte der entsprechenden Register an.
- 1.2 Ändern Sie das Programm „blink.asm“ in ein Programm „lauf.asm“ ab. das ein Lauflicht mit 4 LED's realisiert. Erweitern Sie dann das Programm so. daß durch Betätigen einer der Tasten des Boards das Lauflicht vorwärts oder rückwärts läuft (Hinweis: Geeignet für diese Aufgabe sind die Rotationsbefehle ROL und ROR).
- 1.3 Entwerfen Sie ein Assemblerprogramm, das einen über Konstanten einstellbaren Speicherbereich des NF300 nach einem bestimmten 16-Bit Wort (z.B. \$1234) absucht und die Anzahl der gefundenen Worte sowie die Adressen in dem Speicherbereich ab \$1000 ablegt.

2. Praktikumsversuch:

- 2.1 Übersetzen Sie das gegebene Assemblerprogramm „blink.asm“. Laden Sie mit Hilfe des Monitors BD32 den HEX-Code (im S-Format). Starten Sie das Programm im Single-Step-Mode und überprüfen Sie an den gegebenen Testpunkten die Inhalte der entsprechenden Register und den Status analog zur Vorbereitung 1.2.
- 2.2 Übersetzen und starten Sie nun das von Ihnen erstellte Programm „lauf.asm“.
- 2.3 Geben Sie das vorbereitete Suchprogramm ein und bringen Sie es fehlerfrei zum Laufen. Versuchen Sie die Ablaufgeschwindigkeit des Programms zu optimieren. Zum Austesten des Programms sollten Sie den Breakpoint-Mechanismus des BD32 verwenden.



Programm "blink.asm"

```

*****
*
* Project:      Blinking LED
* Filename:    blink.asm
*
*****

* Description:*****
*
* Blinking LED's on Port E
*
*****

* Addressmap with EQUates:*****
*
RAM   EQU   $0           ;RAM section
STACK EQU   RAM+$800     ;Stack section
PGM   EQU   RAM+$C00     ;Program start
ROM   EQU   $800000      ;ROM section
*
*****

* Definitions:*****
*
*** 68332 global registerdefinitions ***
*
* SIM
SIMCR EQU   $FFFA00      ;SIM Configuration Register
SYPCR EQU   $FFFA20      ;System Protection Register
CSBARBT EQU  $FFFA48      ;Chip Select Base Address Boot Register
CSBAR0 EQU  $FFFA4C      ;Chip Select Base Address Register 0

*PORT E
PEPAR EQU   $FFFA16      ;Pin Assignment Register
DDRE EQU   $FFFA14      ;Data Direction Register
PORTE EQU   $FFFA12      ;Port E is decoded on 2

*PORT F
PFPAR EQU   $FFFA1E      ;Pin Assignment Register
DDRF EQU   $FFFA1C      ;Data Direction Register
PORTF EQU   $FFFA1A      ;Port F is decoded on 2

*
* Globals
SL EQU 1024 ;Stack length
*
*****

*** Init Presets *****
*
*** Place FEPROM at $800000, because unused
*
    ORG CSBARBT ;set on CSBARBT
    DC.W $8003 ;at 8M, size 64KB
*
*** Place ext. RAM at $0, 256KB, 0WS ***
*
    ORG CSBAR0 ;Set on SIM CSs
    DC.W $0005 ;CSBAR0, ext. RAM_RD
    DC.W $6830 ;CSOR0
    DC.W $0005 ;CSBAR1, ext. RAM_WR_LO
    DC.W $3030 ;CSOR1

```



```

DC.W $0005 ;CSBAR2, ext. RAM_WR_HI
DC.W $5030 ;CSOR2
*
*** Initialize SIM and system protection
* Switch off watchdogs, also while FREEZE is active
*
ORG SIMCR
DC.W $60CF ;FREEZE settings
ORG SYPCR
DC.W $0000 ;no system protection
*
*** Define Stack ***
*
ORG STACK ;Stackarea starts at $800
ST DS.B SL ;Length is 1KByte
*
*****

* Program: *****
*
*** Used processor-registers:
* D0: LED-on-register
* D1: LED-off-register
* D2,D3: wait time
*
*** Start in SRAM ***
ORG PGM ;Programcode at $C00
*
*** Initialize registers ***
*
* Initialize stack pointer
BLINK MOVE.L #ST+SL,A7 ;load Stack Pointer
* Initial Program Counter is initialized by the corresponding DO-File
*
* Initialize Port E
MOVE.W #$0000,PEPAR ;I/O instead of systembus
MOVE.W #$000F,DDRE ;4 bit as output
*
* Initialize Port F
MOVE.W #$0000,PFPAR ;I/O instead of systembus
MOVE.W #$0000,DDRF ;all 8 bit as input
*
* Initialize CPU32 registers
MOVE.W #$0000,D0 ;initialize with LED = on
MOVE.W #$FFFF,D1 ;initialize with LED = off
*
*** Blink: main **
*
START MOVE.W D0,PORTE ;LED on -->Testpunkt (SP,PC)
BSR WAIT ;enjoy it
MOVE.W D1,PORTE ;LED off
BSR WAIT ;enjoy it
BRA START ;do forever
*
*** Subroutine ***
*
WAIT MOVE.W #$05,D3 ;Zaehler aussen -->Testpunkt (SP,PC, D3)
LOOP2 MOVE.W #$8000,D2 ;Zaehler innen
LOOP1 DBRA D2,LOOP1
DBRA D3,LOOP2
RTS ;return -->Testpunkt (D2,D3)
*
*** For the Assembler... ***
END BLINK

```