

# UE04\_hexup.c

```

/*****
 * Filename      : UE04_hexup.c      *
 * Created on    : Nov 17, 2018     *
 * Author       : Christian Zahner*
 *****/

#pragma compact_abi

#include "UART0.h"
#include "support_common.h" // include peripheral declarations and more;
#include "uart_support.h"   // universal asynchronous receiver transmitter,
                           // (d.h. die serielle Schnittstelle)
#include "terminal_wrapper.h"
#include <stdio.h>

#include "UE04_HexUp.h"

void hexUP(int zahl){

    asm{

        bra      start

    hexup: // Unterprogramm Hex Ausgabe

        link     a6,#0           //Stackframe aufbauen, 0 lokale Variablen

        adda     #-16, sp        //Platz auf Stack schaffen zum Sichern der
                                //Register (4x Long)
        movem.l   d2-d5, (sp)    //Register sichern

        move.w    8(a6),d5       //Parameter holen => d5
        move.l    #12,d4        //Schleifenzähler in d4

    loop:

        move.w    d5,d3          //kopieren nach d3
        lsr.l     d4,d3          //12/8/4/0 Bit nach rechts schieben

        andi.l    #0x000f,d3     // Maske zum löschen
                                // der linken 12 Bit
        addi.l    #'0',d3        // in ASCII umwandeln
        cmpi.l    #'9',d3        // 9 Ascii ist 57
                                // Compare zieht 57 von d3 ab
                                // anschließend werde je nach
                                // Ergebnis das CCR gesetzt
        ble.b     kein_buchstabe // Sprung wenn Zahl
        addi.l    #'A'-'0'-10,d3 // Hex-Ziffern A - F ermitteln
                                // ASCII Muster von A
                                // Von vorher Binär Muster 0
                                // ASCII abziehen.
                                // -10 da A = 10 ansonsten Buchstaben ab J

    kein_buchstabe:

        move.b    d3,-(sp)       //Parameter auf den Stack (Byte)
        jsr       TERM_Write    //Nibble-Ziffer ausgeben
        adda     #1, sp         //Stack freigeben

        subi.l    #4,d4          //Shift-Register um 4 erniedrigen
        bge       loop          //insgesamt 4 mal wiederholen

        jsr       TERM_WriteLn   //neuZeile
    }
}

```

# UE04\_hexup.c

```

movem.l (sp),d2-d5      //gesicherte Register restaurieren
adda    #16, sp         //Platz auf Stack wieder freigeben (4x Long)

unlk    a6              //Stackframe abbauen
rts     //Return from Subroutine Rücksprung

start:                  //"Hauptprogramm"

    //pea zahl          //Push effective address auf Stack
                        //geht nicht INOUT WriteInt braucht
                        //Zahl(Int 16) nicht Adresse(32)

    move.w zahl,-(sp)    //jsr INOUT_WriteInt //jsr INOUT_WriteInt
                        //adda #2,sp           //Stackbereinigen
                        //jsr TERM_WriteLn     //neuZeile

    //move.w zahl,-(sp)  //Parameter auf den Stack (Word=16 Bit)
    jsr    hexup         //Aufruf des Unterprogramms hexup
    adda    #2,sp        //Clear Stack (Word=16 Bit)

}

```