

```

/*****
*                               V 8 0 6 0 C . C                               *
**-----**
* Task                        : Demonstrates programming in 800x600 graphics *
*                            mode on Super VGA cards with 16 colors. This   *
*                            program requires V8060CA.ASM assembly          *
*                            language module.                               *
**-----**
* Author                     : MICHAEL TISCHER                             *
* Developed on               : 01/14/91                                     *
* Last update                : 03/07/92                                     *
**-----**
* Memory model               : SMALL                                       *
**-----**
* (BORLAND TURBO C)
* Compilation                : Create a project file using the following:   *
*                            v8060c.c                                       *
*                            v8060ca.asm                                    *
**-----**
* Call                       : v8060c                                     *
*****/

```

```

#include <dos.h>
#include <stdarg.h>
#include <stdlib.h>
#include <io.h>
#include <stdio.h>
#include <conio.h>

```

```

/*-- Type declarations -----*/

typedef unsigned char BYTE;

```

```

/*-- External references to the assembler routines -----*/

extern int  init800600( void );
extern void setpix( int x, int y, unsigned char pcolor);
extern BYTE getpix( int x, int y );
extern void far * getfontptr( void );

/*-- Compiler-dependent declarations -----*/

#ifdef __TURBOC__
    #define random(x) ( rand() % (x+1) )           /* Random function */
#endif

/*-- Constants -----*/

#define MAXX      799                /* Maximum X- and Y-coordinates */
#define MAXY      599
#define NUMLINES  2500              /* Number of lines */
#define XSPACING  40                /* Distance of line box from border */
#define YSPACING  30
#define X1        ( 2 * XSPACING )  /* Coordinates of the line box */
#define Y1        ( 2 * YSPACING )
#define X2        ( MAXX-XSPACING )
#define Y2        ( MAXY-YSPACING )
#define XRAND     random( X2 - X1 - 1 ) + X1 + 1  /* Random */
#define YRAND     random( Y2 - Y1 - 1 ) + Y1 + 1  /* coordinates */

/*****
*  IsVga: Determines whether a VGA card is installed.
*  *****/
**-----**
*  Input   : None
*

```

```

*   Output   : 0, if no VGA exists, otherwise < 0
*****/

BYTE IsVga( void )
{
    union REGS Regs;          /* Processor registers for interrupt call */

    Regs.x.ax = 0x1a00;        /* Function 1AH applies to VGA only */
    int86( 0x10, &Regs, &Regs );
    return (BYTE) ( Regs.h.al == 0x1a ); /* Is the function available? */
}

/*****
*   PrintChar : Writes a character to the screen while in graphic mode.*
**-----**
*   Input    :   THECHAR = Character to be written
*                x, y     = X- and Y-coordinates of upper-left corner
*                FG       = Foreground color
*                BK       = Background color
*   Info     : Character is created in an 8x8 matrix based on the
*                8x8 ROM font.
*****/

void PrintChar( char thechar, int x, int y, BYTE fg, BYTE bk )
{
    typedef BYTE FDEF[256][8];          /* Font definition */
    typedef FDEF far *TPTR;              /* Pointer to font */

    BYTE        i, k,                  /* Loop counter */
               BMask;                  /* Bit mask for character design */

    static TPTR fptr = (TPTR) 0;        /* Pointer to font in ROM */

```

```

if ( fptr == (TPTR) 0 )          /* Pointer to font already set? */
    fptr = getfontptr();        /*No --> Use the assembler function to load */

/*- Generate character pixel by pixel -----*/

if ( bk == 255 )                /* Drawing transparent characters? */
    for ( i = 0; i < 8; ++i )    /* Yes --> Set foreground pixels only */
    {
        BMask = (*fptr)[thechar][i];    /* Get bit pattern for one line */
        for ( k = 0; k < 8; ++k, BMask <= 1 )    /* Execute columns */
            if ( BMask & 128 )            /* Pixel set? */
                setpix( x+k, y+i, fg );    /* Yes */
    }
else                             /* No --> Consider background as well */
    for ( i = 0; i < 8; ++i )      /* Execute lines */
    {
        BMask = (*fptr)[thechar][i];    /* Get bit pattern for one line */
        for ( k = 0; k < 8; ++k, BMask <= 1 )    /* Execute columns */
            setpix( x+k, y+i, ( BMask & 128 ) ? fg : bk );
    }
}

/*****
*   Line: Draws a line based on the Bresenham algorithm.
*   ****
*   Input   : X1, Y1 = Starting coordinates (0 - ...)
*             X2, Y2 = Ending coordinates
*             LPCOL = Color of line pixels
*****/

/*-- Function for swapping two integer variables -----*/

```

```

void SwapInt( int *i1, int *i2 )
{
    int dummy;

    dummy = *i2;
    *i2   = *i1;
    *i1   = dummy;
}

/*-- Main part of function -----*/

void Line( int x1, int y1, int x2, int y2, BYTE lpcol )
{
    int d, dx, dy,
        aincr, bincr,
        xincr, yincr,
        x, y;

    if ( abs(x2-x1) < abs(y2-y1) )           /* X- or Y-axis overflow? */
    {                                         /* Check Y-axis */
        if ( y1 > y2 )                       /* y1 > y2? */
        {
            SwapInt( &x1, &x2 );             /* Yes --> Swap X1 with X2 */
            SwapInt( &y1, &y2 );             /* and Y1 with Y2 */
        }

        xincr = ( x2 > x1 ) ? 1 : -1;        /* Set X-axis increment */

        dy = y2 - y1;
        dx = abs( x2-x1 );
        d  = 2 * dx - dy;

```

```

aincr = 2 * (dx - dy);
bincr = 2 * dx;
x = x1;
y = y1;

setpix( x, y, lpcol );
for (y=y1+1; y<= y2; ++y )
{
    if ( d >= 0 )
    {
        x += xincr;
        d += aincr;
    }
    else
        d += bincr;
    setpix(x, y, lpcol);
}
}
else
{
    if ( x1 > x2 )
    {
        SwapInt( &x1, &x2 );
        SwapInt( &y1, &y2 );
    }

    yincr = ( y2 > y1 ) ? 1 : -1;

    dx = x2 - x1;
    dy = abs( y2-y1 );
    d = 2 * dy - dx;
    aincr = 2 * (dy - dx);

    /* Set first pixel */
    /* Execute line on Y-axes */

    /* Check X-axes */
    /* x1 > x2? */

    /* Yes --> Swap X1 with X2 */
    /* and Y1 with Y2 */

    /* Set Y-axis increment */

```

```

bincr = 2 * dy;
x = x1;
y = y1;

setpix(x, y, lpcol);
for (x=x1+1; x<=x2; ++x )
{
    if ( d >= 0 )
    {
        y += yincr;
        d += aincr;
    }
    else
        d += bincr;
    setpix(x, y, lpcol);
}
}

/*****
* GrfxPrintf: Displays a formatted string on the graphic screen.
*-----**
* Input   : X, Y   = Starting coordinates (0 - ...)
*           FG     = Foreground color
*           BK     = Background color (255 = transparent)
*           STRING = String with format information
*           ...    = Arguments similar to printf
*****/

void GrfxPrintf( int x, int y, BYTE fg, BYTE bk, char * string, ... )
{
    va_list parameter;
    /* Parameter list for VA... macros */

```

```

char stngptr[255],                /* Buffer for formatted string */
    *cp;

va_start( parameter, string );    /* Convert parameter */
vsprintf( stngptr, string, parameter ); /* Format */
for ( cp = stngptr; *cp; ++cp, x+= 8 ) /* Formatted string */
    PrintChar( *cp, x, y, fg, bk ); /* Display using PrintChar */
}

/*****
* DrawAxis: Draws axes from the left and top borders on the screen. *
**-----**
* Input : STEPX = Increment for X-axis *
*        STEPY = Increment for Y-axis *
*        FG = Foreground color *
*        BK = Background color (255 = transparent) *
*****/

void DrawAxis( int stepx, int stepy, BYTE fg, BYTE bk )
{
    int x, y;                    /* Loop coordinates */

    Line( 0, 0, MAXX, 0, fg );   /* Draw X-axis */
    Line( 0, 0, 0, MAXY, fg );   /* Draw Y-axis */

    for ( x = stepx; x < MAXX; x += stepx ) /* Scale X-axis */
    {
        Line( x, 0, x, 5, fg );
        GrfxPrintf( x < 100 ? x - 8 : x - 12, 8, fg, bk, "%d", x );
    }

    for ( y = stepy; y < MAXY; y += stepy ) /* Scale Y-axis */

```



```

{
    Line( 0, y, 5, y, fg );
    GrfxPrintf( 8, y-4, fg, bk, "%3d", y );
}
}

/*****
*   Demo: Demonstrates the functions in this module.   *
*****/

void Demo( void )
{
    int i;                                /* Loop counter */

    DrawAxis( 30, 20, 15, 255 );          /* Draw axes */

    GrfxPrintf( X1, MAXY-10, 15, 255,
               "V8060C.C - (c) by Michael Tischer" );

    Line( X1, Y1, X1, Y2, 15 );           /* Draws a frame around the line box */
    Line( X1, Y2, X2, Y2, 15 );
    Line( X2, Y2, X2, Y1, 15 );
    Line( X2, Y1, X1, Y1, 15 );

    /*-- Create random lines within the line box -----*/

    for ( i = NUMLINES; i > 0 ; --i )
        Line( XRAND, YRAND, XRAND, YRAND, (BYTE) (i % 16) );
}

/*****
**                                     M A I N   P R O G R A M                                     ***
**

```

```

/*****
void main( void )
{
    union REGS regs;

    printf( "V8060C.C - (c) 1992 by Michael Tischer\n\n" );
    if ( IsVga() ) /* VGA card installed? */
    {
        /* Yes, but can the mode also be initialized? */
        if ( init800600() )
        {
            /* Mode O.K. */
            Demo(); /* Execute demo */
            getch(); /* Wait for a key */
            regs.x.ax = 0x0003; /* Shift into text mode */
            int86( 0x10, &regs, &regs );
        }
        else
        {
            printf( "Attention: 800x600 mode could not be initialized\n" );
            exit( 1 );
        }
    }
    else
    {
        /* No, no VGA card */
        printf( "This program requires a VGA card\n" );
        exit(1); /* End program */
    }
}

```