```
//Done
/*************************************************************************/
/*                                                                       */
/*                            Display Renderer                           */
/*                          5x7 Dot Matrix Codes                         */
/*                taken from Digital Oscilloscope Project                */
/*                               EE/CS 52                                */
/*                                                                       */
/*************************************************************************/

/*
   This file contains the render_displaybuffer function. It takes as input a pointer to
   the start of the display buffer to write to, and an ASCII string to render there

   Revision History
      5/27/08  Glen George       Initial revision of ascii_patterns(from
                                 3/10/95 version of char57.asm).
      6/10/17  Will Werst        Initial code
*/

const unsigned char ascii_char_patterns[] = {

    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x00)            */
    0x04, 0x0E, 0x15, 0x04, 0x04, 0x04, 0x04,   /* up arrow (0x01)          */
    0x04, 0x04, 0x04, 0x04, 0x15, 0x0E, 0x04,   /* down arrow (0x02)        */
    0x00, 0x04, 0x08, 0x1F, 0x08, 0x04, 0x00,   /* left arrow (0x03)        */
    0x00, 0x11, 0x11, 0x11, 0x1B, 0x14, 0x10,   /* greek u (mu) (0x04)      */
    0x00, 0x04, 0x02, 0x1F, 0x02, 0x04, 0x00,   /* right arrow (0x05)       */
    0x00, 0x11, 0x0A, 0x04, 0x0A, 0x11, 0x00,   /* multiply symbol (0x06)   */
    0x00, 0x04, 0x00, 0x1F, 0x00, 0x04, 0x00,   /* divide symbol (0x07)     */
    0x04, 0x04, 0x1F, 0x04, 0x04, 0x00, 0x1F,   /* plus/minus symbol (0x08) */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x09)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x0A)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x0B)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x0C)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x0D)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x0E)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x0F)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x10)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x11)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x12)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x13)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x14)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x15)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x16)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x17)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x18)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x19)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x1A)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x1B)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x1C)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x1D)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x1E)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* UNUSED (0x1F)            */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   /* space (0x20)             */
    0x04, 0x04, 0x04, 0x04, 0x04, 0x00, 0x04,   /* !                        */
    0x0A, 0x0A, 0x0A, 0x00, 0x00, 0x00, 0x00,   /* "                        */
    0x0A, 0x0A, 0x1F, 0x0A, 0x1F, 0x0A, 0x0A,   /* #                        */
    0x04, 0x0F, 0x14, 0x0E, 0x05, 0x1E, 0x04,   /* $                        */
    0x18, 0x19, 0x02, 0x04, 0x08, 0x13, 0x03,   /* %                        */
    0x08, 0x14, 0x14, 0x08, 0x15, 0x12, 0x0D,   /* &                        */
    0x0C, 0x0C, 0x08, 0x10, 0x00, 0x00, 0x00,   /* '                        */
    0x02, 0x04, 0x08, 0x08, 0x08, 0x04, 0x02,   /* (                        */
    0x08, 0x04, 0x02, 0x02, 0x02, 0x04, 0x08,   /* )                        */
    0x04, 0x15, 0x0E, 0x1F, 0x0E, 0x15, 0x04,   /* *                        */
    0x00, 0x04, 0x04, 0x1F, 0x04, 0x04, 0x00,   /* +                        */
```

```
    0x00, 0x00, 0x00, 0x0C, 0x0C, 0x08, 0x10,   /* ,                */
    0x00, 0x00, 0x00, 0x1F, 0x00, 0x00, 0x00,   /* -                */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x0C, 0x0C,   /* .                */
    0x00, 0x01, 0x02, 0x04, 0x08, 0x10, 0x00,   /* /                */
    0x0E, 0x11, 0x13, 0x15, 0x19, 0x11, 0x0E,   /* 0                */
    0x04, 0x0C, 0x04, 0x04, 0x04, 0x04, 0x0E,   /* 1                */
    0x0E, 0x11, 0x01, 0x0E, 0x10, 0x10, 0x1F,   /* 2                */
    0x0E, 0x11, 0x01, 0x06, 0x01, 0x11, 0x0E,   /* 3                */
    0x02, 0x06, 0x0A, 0x12, 0x1F, 0x02, 0x02,   /* 4                */
    0x1F, 0x10, 0x1E, 0x01, 0x01, 0x11, 0x0E,   /* 5                */
    0x06, 0x08, 0x10, 0x1E, 0x11, 0x11, 0x0E,   /* 6                */
    0x1F, 0x01, 0x02, 0x04, 0x08, 0x10, 0x10,   /* 7                */
    0x0E, 0x11, 0x11, 0x0E, 0x11, 0x11, 0x0E,   /* 8                */
    0x0E, 0x11, 0x11, 0x0F, 0x01, 0x02, 0x0C,   /* 9                */
    0x00, 0x0C, 0x0C, 0x00, 0x0C, 0x0C, 0x00,   /* :                */
    0x0C, 0x0C, 0x00, 0x0C, 0x0C, 0x08, 0x10,   /* ;                */
    0x02, 0x04, 0x08, 0x10, 0x08, 0x04, 0x02,   /* <                */
    0x00, 0x00, 0x1F, 0x00, 0x1F, 0x00, 0x00,   /* =                */
    0x08, 0x04, 0x02, 0x01, 0x02, 0x04, 0x08,   /* >                */
    0x0E, 0x11, 0x01, 0x02, 0x04, 0x00, 0x04,   /* ?                */
    0x0E, 0x11, 0x01, 0x0D, 0x15, 0x15, 0x0E,   /* @                */
    0x04, 0x0A, 0x11, 0x11, 0x1F, 0x11, 0x11,   /* A                */
    0x1E, 0x09, 0x09, 0x0E, 0x09, 0x09, 0x1E,   /* B                */
    0x0E, 0x11, 0x10, 0x10, 0x10, 0x11, 0x0E,   /* C                */
    0x1E, 0x09, 0x09, 0x09, 0x09, 0x09, 0x1E,   /* D                */
    0x1F, 0x10, 0x10, 0x1C, 0x10, 0x10, 0x1F,   /* E                */
    0x1F, 0x10, 0x10, 0x1C, 0x10, 0x10, 0x10,   /* F                */
    0x0F, 0x10, 0x10, 0x13, 0x11, 0x11, 0x0F,   /* G                */
    0x11, 0x11, 0x11, 0x1F, 0x11, 0x11, 0x11,   /* H                */
    0x0E, 0x04, 0x04, 0x04, 0x04, 0x04, 0x0E,   /* I                */
    0x01, 0x01, 0x01, 0x01, 0x01, 0x11, 0x0E,   /* J                */
    0x11, 0x12, 0x14, 0x18, 0x14, 0x12, 0x11,   /* K                */
    0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x1F,   /* L                */
    0x11, 0x1B, 0x15, 0x15, 0x11, 0x11, 0x11,   /* M                */
    0x11, 0x19, 0x15, 0x13, 0x11, 0x11, 0x11,   /* N                */
    0x0E, 0x11, 0x11, 0x11, 0x11, 0x11, 0x0E,   /* O                */
    0x1E, 0x11, 0x11, 0x1E, 0x10, 0x10, 0x10,   /* P                */
    0x0E, 0x11, 0x11, 0x11, 0x15, 0x12, 0x0D,   /* Q                */
    0x1E, 0x11, 0x11, 0x1E, 0x14, 0x12, 0x11,   /* R                */
    0x0E, 0x11, 0x10, 0x0E, 0x01, 0x11, 0x0E,   /* S                */
    0x1F, 0x04, 0x04, 0x04, 0x04, 0x04, 0x04,   /* T                */
    0x11, 0x11, 0x11, 0x11, 0x11, 0x11, 0x0E,   /* U                */
    0x11, 0x11, 0x11, 0x0A, 0x0A, 0x04, 0x04,   /* V                */
    0x11, 0x11, 0x11, 0x11, 0x15, 0x1B, 0x11,   /* W                */
    0x11, 0x11, 0x0A, 0x04, 0x0A, 0x11, 0x11,   /* X                */
    0x11, 0x11, 0x0A, 0x04, 0x04, 0x04, 0x04,   /* Y                */
    0x1F, 0x01, 0x02, 0x04, 0x08, 0x10, 0x1F,   /* Z                */
    0x0E, 0x08, 0x08, 0x08, 0x08, 0x08, 0x0E,   /* [                */
    0x00, 0x10, 0x08, 0x04, 0x02, 0x01, 0x00,   /* \                */
    0x0E, 0x02, 0x02, 0x02, 0x02, 0x02, 0x0E,   /* ]                */
    0x04, 0x0A, 0x11, 0x00, 0x00, 0x00, 0x00,   /* ^                */
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1F,   /* _                */
    0x06, 0x06, 0x04, 0x02, 0x00, 0x00, 0x00,   /* `                */
    0x00, 0x00, 0x0E, 0x01, 0x0F, 0x11, 0x0F,   /* a                */
    0x10, 0x10, 0x16, 0x19, 0x11, 0x19, 0x16,   /* b                */
    0x00, 0x00, 0x0E, 0x11, 0x10, 0x11, 0x0E,   /* c                */
    0x01, 0x01, 0x0D, 0x13, 0x11, 0x13, 0x0D,   /* d                */
    0x00, 0x00, 0x0E, 0x11, 0x1F, 0x10, 0x0E,   /* e                */
    0x02, 0x05, 0x04, 0x0E, 0x04, 0x04, 0x04,   /* f                */
    0x0D, 0x13, 0x13, 0x0D, 0x01, 0x11, 0x0E,   /* g                */
    0x10, 0x10, 0x16, 0x19, 0x11, 0x11, 0x11,   /* h                */
    0x04, 0x00, 0x0C, 0x04, 0x04, 0x04, 0x0E,   /* i                */
    0x01, 0x00, 0x01, 0x01, 0x01, 0x11, 0x0E,   /* j                */
    0x10, 0x10, 0x12, 0x14, 0x18, 0x14, 0x12,   /* k                */
    0x0C, 0x04, 0x04, 0x04, 0x04, 0x04, 0x0E,   /* l                */
    0x00, 0x00, 0x1A, 0x15, 0x15, 0x15, 0x15,   /* m                */
```

```
    0x00, 0x00, 0x16, 0x19, 0x11, 0x11, 0x11,   /* n                    */
    0x00, 0x00, 0x0E, 0x11, 0x11, 0x11, 0x0E,   /* o                    */
    0x16, 0x19, 0x11, 0x19, 0x16, 0x10, 0x10,   /* p                    */
    0x0D, 0x13, 0x11, 0x13, 0x0D, 0x01, 0x01,   /* q                    */
    0x00, 0x00, 0x16, 0x19, 0x10, 0x10, 0x10,   /* r                    */
    0x00, 0x00, 0x0F, 0x10, 0x0E, 0x01, 0x1E,   /* s                    */
    0x04, 0x04, 0x1F, 0x04, 0x04, 0x05, 0x02,   /* t                    */
    0x00, 0x00, 0x11, 0x11, 0x11, 0x13, 0x0D,   /* u                    */
    0x00, 0x00, 0x11, 0x11, 0x11, 0x0A, 0x04,   /* v                    */
    0x00, 0x00, 0x11, 0x11, 0x15, 0x15, 0x0A,   /* w                    */
    0x00, 0x00, 0x11, 0x0A, 0x04, 0x0A, 0x11,   /* x                    */
    0x11, 0x11, 0x11, 0x0F, 0x01, 0x11, 0x0E,   /* y                    */
    0x00, 0x00, 0x1F, 0x02, 0x04, 0x08, 0x1F,   /* z                    */
    0x02, 0x04, 0x04, 0x08, 0x04, 0x04, 0x02,   /* {                    */
    0x04, 0x04, 0x04, 0x00, 0x04, 0x04, 0x04,   /* |                    */
    0x08, 0x04, 0x04, 0x02, 0x04, 0x04, 0x08,   /* }                    */
    0x08, 0x15, 0x02, 0x00, 0x00, 0x00, 0x00,   /* ~                    */
    0x0A, 0x15, 0x0A, 0x15, 0x0A, 0x15, 0x0A    /* DEL (0x7F)           */

};



/* library include files */


/* local include files */
    /* none */

/*
   render_displaybuffer

   Description: Renders the passed string to the passed buffer.

   Operation: The buffer is a sequence of bytes that specify
              how each column of pixels should be set, with each bit
              in a byte setting one pixel in a column. The code goes
              through all the columns, and picks out the appropriate
              bits from the ascii_char_patterns.

   Arguments: *string - pointer to start of null-terminated string
              *buffer - pointer to start of display buffer
              length - length of string
   Return Value: None

   Input:            None.
   Output:           None.

   Error Handling:   None.

   Algorithms:       None.
   Data Structures:  None.

   Shared Variables:

   Author:           Will Werst
   Last Modified:    June 23, 2017

*/

void render_displaybuffer(char *string, char *buffer, int length){
    int col;
    for (col = 0; col < length; col++){
        /*if (col % 6 == 0){
            buffer[col] = 0x00;
            continue;
```

```
        }*/
        int cur_str_pos = (col / 6);
        char cur_char = string[cur_str_pos];
        if (cur_char == 0x00){
            break;
        }
        int row;
        for (row = 0; row < 7; row++){
            if ((ascii_char_patterns[cur_char*7 + row] & (0x10 >> (col % 6))) != 0){         //
            If the current value is blank
                buffer[col] = buffer[col] ^ (0x1 << row);
            }
        }
    }
}

// Clears the passed buffer
void clear_displaybuffer(char *buffer, int length){
    int col;
    for (col = 0; col < 512; col++){
        buffer[col] = 0x00;
    }
}

// Does what it says it does
int divide(int num, int den){
    return num / den;
}

// Does what it says it does
int mod(int num, int den){
    return num % den;
}
```