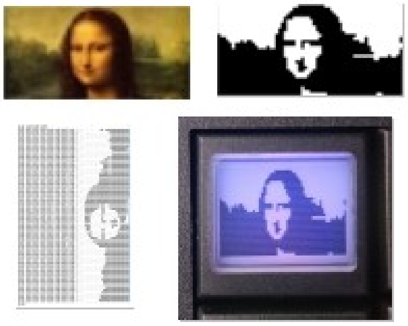
[E^3]



Bit/Pixel Mapping

Application Note

Table of Contents

Image Conversion for LCD Keyswitches....3

Image Mapping....4

Scaling to Aspect Ratio....4

Adjustment to Switch Resolution....5

Conversion to Black & White....5

Mirroring & Rotation....5

Splitting....5

Java Implementation Example....8

Notices....11

Copyright Notice....11

Technical Notice....11

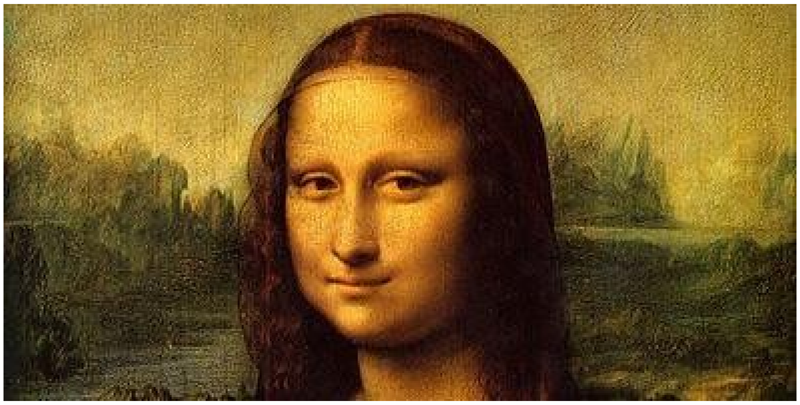
Warranty Disclaimer....11

Change History....12

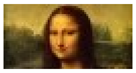
Image Conversion for LCD Keyswitches

This document shows the steps involved in sending an image to a LCD keyswitch from [E³]. The process described in this document allows a user to take any image and convert it to the appropriate data format for display on the [E³] LCD keyswitches. Also shown is the automation of these steps using a Java application.

To illustrate the example we will use an image of Leonardo DaVinci's Mona Lisa to be displayed on a SA6432 LCD keyswitch.



The conversion of a color image to the [E³] data format involves several steps:



Scaling of the image to the correct aspect ratio

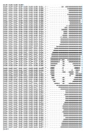
Adjustment of the resolution to the switch type

Conversion to black & white

Image mirroring

Rotation

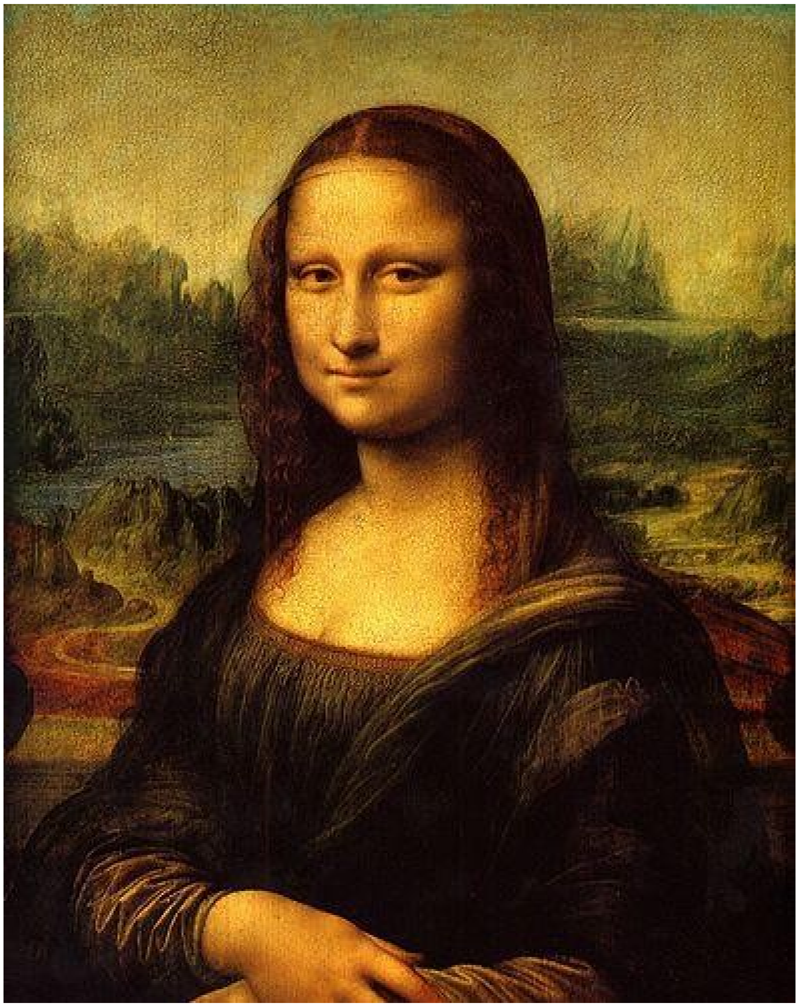
Splitting (if necessary)



Bitmapping to the data format

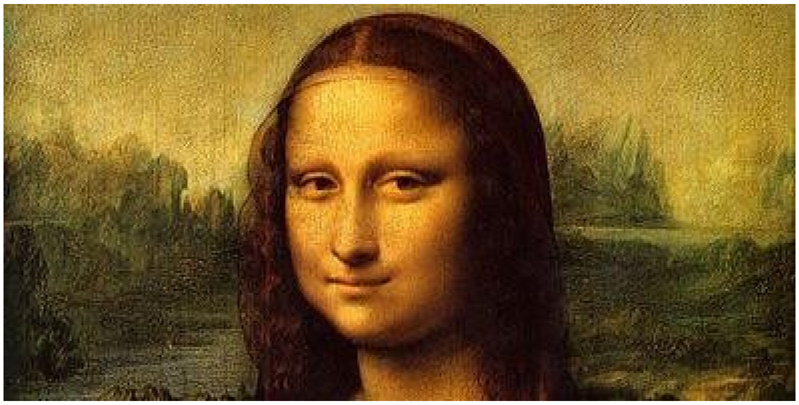
Image Mapping

Scaling to Aspect Ratio



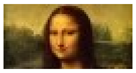
This image of the Mona Lisa has a resolution of 396 x 499 pixels corresponding to a 4-to-5 aspect ratio. The SA6432 has an aspect ratio of 2-to-1 due to its resolution of 64 x 32 pixels.

To adjust the image to the correct aspect ratio we cropped the image to 396 x 198 pixels.



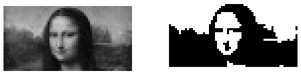
Adjustment to Switch Resolution

This image has the correct aspect ratio and needs to be converted to 64 x 32 pixels resolution.



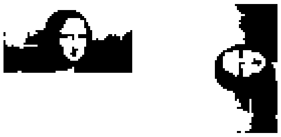
Conversion to Black & White

The LCD of the SA6432 switch is a black & white display. Therefore, we need to convert the image into a black & white bitmap. This can be done in one step or with an intermediate grayscale image as shown here.



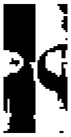
Mirroring & Rotation

In order to correspond to the bit mapping format of the LCD keyswitches the image now needs to be mirrored and rotated 90° counter-clockwise.



Splitting

The SA6432 bitmap format also requires that the upper and lower half of the image are reversed.



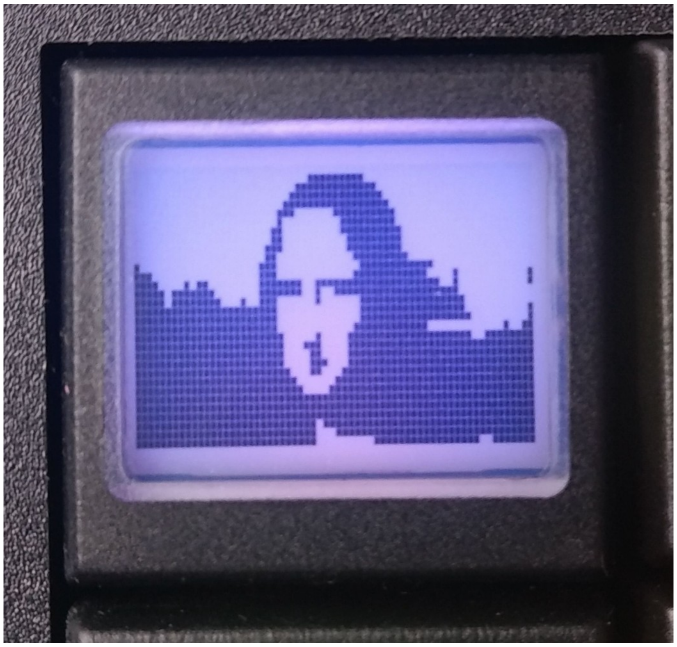
The image can now be transmitted to the SA6432 keyswitch with the 0x40 command. In order to do this four pixels, starting at the top left position, are combined into a nibble and transmitted as the least significant four bits.

The following .e3t script shows the corresponding data format for the image:

```
Transmit Bitmap: C:\MCP\Bitmap\MonaLisa.bmp
0x40                        * Set Display address and write display data
;              * Data Bytes to follow 512 each containing just one nibble
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x03 ;* -             ##  ################
0x0C 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                   ##############
0x08 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                    #############
0x08 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                    #############
0x0C 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                   ##############
0x08 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                    #############
0x08 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                    #############
0x08 0x0F 0x0F 0x07 0x00 0x00 0x00 0x00 ;* -                    ############
0x00 0x0F 0x0F 0x07 0x00 0x00 0x00 0x00 ;* -                     ###########
0x00 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                     ############
0x0A 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                  # #############
0x0B 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x08 ;* -                ### #############
0x0B 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0F ;* -             ###### #############
0x0B 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0C ;* -               #### #############
0x0B 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0C ;* -               #### #############
0x0B 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0E ;* -              ##### #############
0x0B 0x0F 0x0F 0x0F 0x00 0x00 0x08 0x0E ;* -            # ##### #############
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x08 0x0F ;* -            #####################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x08 0x0F ;* -            #####################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x08 0x0F ;* -            #####################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x0C 0x0F ;* -           ######################
0x0F 0x0F 0x0F 0x0F 0x00 0x08 0x0F 0x0F ;* -        #########################
0x0F 0x0F 0x0F 0x0F 0x00 0x0E 0x0F 0x0F ;* -      ###########################
0x0F 0x0F 0x0F 0x0F 0x00 0x0F 0x0F 0x0F ;* -     ############################
0x0F 0x0F 0x0F 0x0F 0x08 0x0F 0x0F 0x0F ;* -    #############################
0x0F 0x0F 0x0F 0x0F 0x0C 0x0F 0x0F 0x0F ;* -   ##############################
0x0F 0x0F 0x0F 0x07 0x0C 0x0F 0x0F 0x0F ;* -   #############################
0x0F 0x0F 0x0F 0x07 0x0E 0x0F 0x0F 0x0F ;* -  ##############################
0x0C 0x0F 0x0F 0x07 0x0E 0x0F 0x07 0x07 ;* -  ########## ###   #############
0x08 0x0F 0x0F 0x07 0x0E 0x0F 0x03 0x06 ;* -  #########   ##    ############
0x00 0x08 0x0F 0x07 0x0E 0x0F 0x00 0x06 ;* -  #######     ##        ########
0x00 0x00 0x0F 0x07 0x0E 0x03 0x00 0x06 ;* -  #####       ##         #######
0x00 0x00 0x0E 0x03 0x0E 0x01 0x00 0x02 ;* -  ####        #           #####
0x00 0x04 0x0C 0x03 0x0E 0x01 0x00 0x02 ;* -  ####        #        #   ####
0x08 0x0F 0x0C 0x01 0x0E 0x01 0x00 0x0E ;* -  ####        ###   #####  ###
0x00 0x0F 0x0C 0x0F 0x0E 0x01 0x00 0x00 ;* -  ####               ####  ######
0x00 0x01 0x0C 0x0F 0x0C 0x01 0x00 0x00 ;* -   ###               #     ######
0x00 0x00 0x0E 0x0F 0x0C 0x01 0x00 0x06 ;* -   ###        ##          #######
0x00 0x00 0x0F 0x0F 0x08 0x03 0x00 0x06 ;* -    ###       ##         ########
0x00 0x08 0x0F 0x0F 0x00 0x03 0x00 0x06 ;* -     ##       ##        #########
0x08 0x0F 0x0F 0x0F 0x00 0x0E 0x03 0x06 ;* -      #####   ##    #############
0x0F 0x0F 0x0F 0x0F 0x00 0x08 0x0F 0x0F ;* -        #########################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x0E 0x0F ;* -          #######################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x08 0x0F ;* -            #####################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                 ################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                 ################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x08 ;* -                #################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                 ################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                 ################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                 ################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x08 ;* -                #################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0C ;* -               ##################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0E ;* -              ###################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0E ;* -              ###################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0C ;* -               ##################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0E ;* -              ###################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0C ;* -               ##################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0C ;* -               ##################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x00 ;* -                 ################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0C ;* -               ##################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0E ;* -              ###################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0F ;* -             ####################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x00 0x0F ;* -             ####################
0x0F 0x0F 0x0F 0x0F 0x00 0x00 0x08 0x0F ;* -            #####################
* - End of Bitmap
```

Note: **The .e3t script as generated by the MCP program shows the actual image bitmap without the top and bottom half reversed.**

Once transmitted to the SA6432 keyswitch the image will appear like this:



Java Implementation Example

This chapter contains an excerpt from the [E³] PanelControl java program which takes an image and creates the image command (0x40) for transmission to a SA6432 keyswitch in a .e3t script file.

```
/**
* This method write the command for display of an image in a .e3t Script file.
*
* @param   pixelArray  Image in form of a one dimensional integer array
                       Each pixel, starting with the top left pixel, is inserted
                       into the array resulting in an array of height x width
                       resolution.
*/
private void writeImage(int pixelArray[]){

      //Pixel per column of the LCD display – in this case for resolution 64x32
      int height = 32;
      //Pixel per row of the LCD display – in this case for resolution 64x32
      int width  = 64;

      //BitSet[] Array containing a bit list of each row of the rotated image
      BitSet[] pixelMap = new BitSet[width];

      int index = 0;

      //Create PixelMap (The images is rotated 90° counter-clockwise and mirrored).
      for(int i = width - 1 ; i >= 0 ; i--){

           pixelMap[width - i - 1] = new BitSet();

           for(int j = 0 ; j < height ; j++){

                 if(pixelArray[i + width * j] == 1){
                       pixelMap[width - i - 1].set(j, true);
                   } else {
                       pixelMap[width - i - 1].set(j, false);
                   }

                   index++;
           }
        }
```

```
    //Output of the command for selecting the first pixel of the display in the
    //.e3t Script file. \the targetWriter is an output channel for that file.
    targetWriter.println("0x40 0x00 0x00 0x00");

    //this string is a buffer for assembling a nibble first in binary, then in
    //Hex format.
    String tmpBinary = "";
    //this string array contains as many strings as the rotated image has columns.
    //In the case of a 64x32 resolution there are 8 column strings.
    String[] bytesArray = new String[height / 4];

    //This loop assembles one nibble per column for each row of the rotated image.
    //Once all nibbles of a row are assembled they handed off to another method,
    //which handles the specific requirements for each resolution.
    for(int i = 0 ; i < pixelMap.length ; i++){

        index = 0;
        for(int j = 0 ; j < height ; j++){
            //Assembling one nibble. 1 means pixel ON; 0 means pixel OFF.
            if(pixelMap[i].get(j)){
                tmpBinary = tmpBinary + "1";
            } else {
                tmpBinary = tmpBinary + "0";
            }

            //Once four pixels are read the nibble representation is changed
            //from binary to HEX format. For example, a „0110“ would generate
            //the string „0x06“.
            if(tmpBinary.length() > 3){
                bytesArray[index] = "0x0" +
        Integer.toHexString(Integer.parseInt(tmpBinary, 2));
                index++;
                tmpBinary = "";
            }
        }

        //After reading the complete row the method for writing the read row
        //is executed.
            writeBytesInColumn(bytesArray);
    }

    //Finally, the END command to terminate the transmission is written into
    //the .e3t script file.
    targetWriter.println("0x43");
}
```

```
/**
* This method writes the pixels of a column of an image into the .e3t scrpt file.

*
* @param   bytesInColumn   This string array contains the column bytes in HEX format.
*/
private void writeBytesInColumn(String[] bytesInColumn){
      switch(board.getKeyType()){
            case LCDKey.SA6432:
                              //First, the last four bytes of the row are sent
                              //(since the lower part of the image must be sent
                              //first).
                              for(int i = 4 ; i < 8 ; i++){
                                    targetWriter.print(bytesInColumn[i] + " ");
                              }
                              //Next follow the first four bytes.
                              for(int i = 0 ; i < 4 ; i++){
                                          targetWriter.print(bytesInColumn[i] + " ");
                              }
                              break;
      }
}
```

NOTICES

Copyright Notice

Technical Notice

This datasheet is intended for technically qualified personnel trained in the field of electronics.

The knowledge of electronics and the technically correct implementation of the content of this datasheet are required for problem free installation, implementation and safe operation of the described product. Only qualified personnel have the required know-how to implement the specifications given in this data sheet.

For clarity, not all details regarding the product or its implementation, installation, operation, or maintenance have been included. Should you require additional information or further assistance, please contact your local [E³] distributor or [E³] Engstler Elektronik Entwicklung GmbH at **techsupport@e3-keys.com**. You may also visit our website at **www.e3-keys.com.**

Warranty Disclaimer

[E³] ENGSTLER ELEKTRONIK ENTWICKLUNG GMBH grants no warranty with respect to this data sheet, neither explicit nor implied, and it is not liable for direct or indirect damages. Some states do not grant the exclusion of incidental or consequential damages and, therefore, this statement may not be valid in such cases.

This data sheet has been produced with all due care. However, since errors cannot be excluded, [E³] Engstler Elektronik Entwicklung GmbH does not grant any warranty or accept any legal responsibility or liability in any form for erroneous statements herein.

Change History

Version	Date	Comments
0.1	10/07/09	Initial draft document (German)
0.2	11/12/09	Initial draft document (English)
0.3	06/30/20	New Formatting
0.4	03/30/22	Updated and edited content
1.0	06/15/22	Updated release version

[E³] Engstler Elektronik Entwicklung GmbH
Industriering 7 ● 63868 Grosswallstadt ● Germany
WWW.E3-KEYS.COM