Using the HT1632 for Dot Matrix LED Displays

D/N: HA0127E

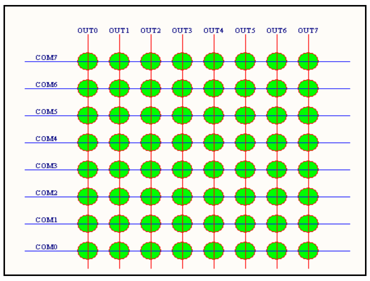
Introduction

The HT1632 is a Memory Mapping LED display driver. Its application range is wide and can be used in applications such as clocks, thermometers, humidity meters, etc. as well as industrial instrument displays. The HT1632 characteristics are shown below. The example uses the HT48F50E, however the application could also use the HT48F70E etc. The device includes 2K bits of EEPROM Data Memory.

- HT1632 Operating Voltage: 2.V~5.5V
- Multi-type display - 32 outbits & 8 commons or 24 outbits & 16 commons
- Internal display RAM - if 32 outbits & 8 commons display is selected, then the RAM is 64*4bits and if the 24 outbits & 16 commons display is selected, then the RAM is 96*4bits
- 16-level display control
- Internal 256K RC oscillator
- Serial Interface communication with MCU
- Operation and Data instruction communication with MCU
- Common lines can be selected as NMOS or PMOS open drain output drivers

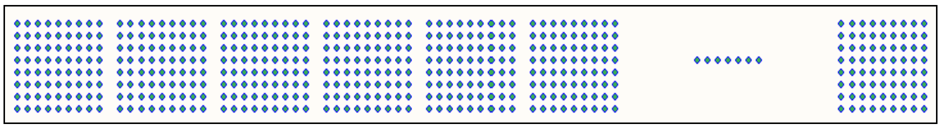
Evaluating the HT1632 LED Display

Dot Matrix LED Display Structure



Smallest LED Panel

The HT1632 is used to drive an LED display with 18 sections as shown in the above diagram to form a 144 x 8 dot matrix overall display. The effect is shown as follows:



Evaluation of the HT1632

The above has already described the HT1632 for 32x8 and 24x16 dot matrix displays, however in this application the dot matrix size to be driven is 144x8. If the HT1632 selects the 24x16 mode, then this can also drive larger dot matrix LEDs. As (144x8) / (24x16) = 3, in this way three HT1632 can drive this size of display.

The HT1632 can drive 24x16 = 384 pixel LED, each LED section matrix has 64 pixels. So one HT1632 can drive 6 minimum size LED dot matrix panels. Three HT1632 devices can therefore drive 18 LED dot matrix displays.

As the LED dot matrix display to be illuminated is especially large, therefore the demands on the HT1632 device is correspondingly large.

Selecting the MCU to Drive the HT1632

The choice of MCU will be determined by the choice of amount of displays graphic and how the display changes. A greater number of displays will require larger Program Memory capacities, while greater changes in the display will require larger Data Memory capacities. The HT1632 only requires three MCU output lines.

HT1632 LED Display Driving Method

Graphic Display Concept

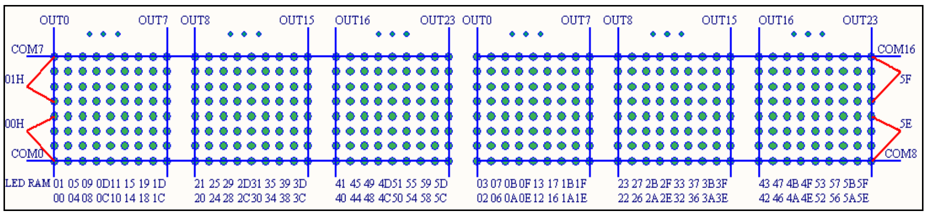
The HT1632 is a Memory Mapping LED driver type which makes programming the graphical display much more convenient. In practice using the Memory Mapping LED driver to display a graphic, becomes a logic write operation to the LED RAM.

In the LED display we always call the object to be displayed a graphic. For example, the single character ″H″ will be seen as a graphical object. For the example of ″HOLTEK SEMICONDUCTOR INC.″, this will also be seen as a graphical operation.

Every graphic in the MCU Memory will have a data and a corresponding address. The HT1632 is used to display the data written into the LED driver RAM.

LED Dot Matrix and LED RAM relationship

In the diagram below the HT1632 has to drive 6 LED graphic displays. The relationship between the LED pixels and the LED RAM, under normal display conditions, is to create a table in the LED RAM. Then according to the position pointer, extract the LED RAM from the table, and then place the related picture data into the LED RAM. In this way the required picture can be displayed.



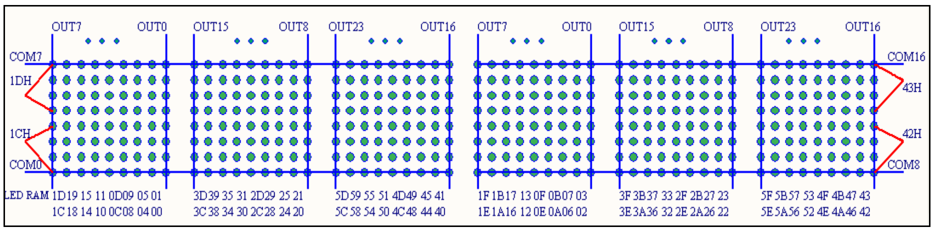
LED_RAM_TABLE:

DC 00H, 01H, 04H, 05H, 08H, 09H, 0CH, 0DH, 10H, 11H, 14H, 15H, 18H, 19H, 1CH, 1DH

DC 20H, 21H, 24H, 25H, 28H, 29H, 2CH, 2DH, 30H, 31H, 34H, 35H, 38H, 39H, 3CH, 3DH

……

If the HT1632 OUT line and COM line and LED pixel area connections experience a change, like in the diagram below, then it is only necessary to change the LED table LED RAM order. It is not necessary to change the program. This look up table method is more convenient.



LED_RAM_TABLE:

DC 1CH, 1DH, 18H, 19H, 14H, 15H, 10H, 11H, 0CH, 0DH, 08H, 09H, 04H, 05H, 00H, 01H

DC 3CH, 3DH, 38H, 39H, 34H, 35H, 30H, 31H, 2CH, 2DH, 28H, 29H, 24H, 25H, 20H, 21H

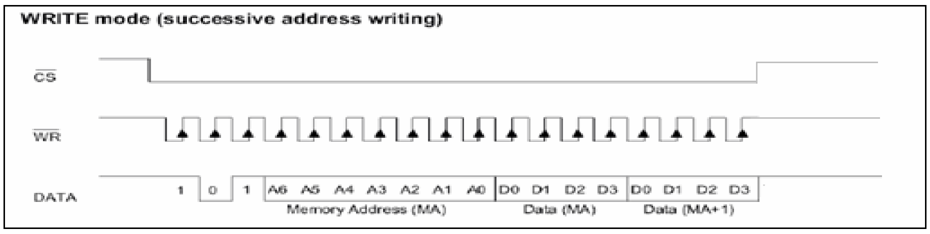
……

Establishing the Picture Data

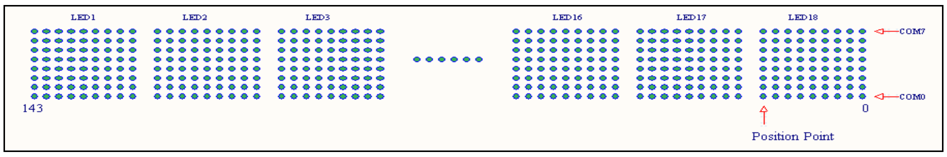
Any picture is stored with data in the MCU memory. To establish picture data, normally the LED RAM and the LED pixel is the same. For example: COM7, OUT0 crossover point corresponds to an address of 01H bit 3; COM4, OUT0 crossover point corresponds to an address of 01H bit0; COM3, OUT0 crossover point corresponds to an address of 00H bit3; COM0, OUT0 crossover point corresponds to an address of 00H bit0. 01H and 00H are both 8-bit addresses, and when setting up picture data, the highest bit7 must correspond to 01H bit3. The data lowest bit0, must correspond to 00H bit0. In this way establishing picture data by writing data to the LED RAM is convenient.

How to Write Data to the HT1632

The picture data is stored in 8-bit format in the MCU memory. However the LED RAM addresses are 4-bit. To place 8-bit data into two 4-bit RAM, we must use the HT1632 continuing bit address write method. To write 8-bit data it is only necessary to consecutively write two RAM addresses. As shown in the following diagram: note that writing bit addresses, first send the high one then the low one.



144x8 Large Screen Picture Display Method



The LED display is, as shown in the diagram, a 144x 8 pixel type. So how do we display a picture on the display? For example the word, ″HOLTEK″, one way is to first determine the position of the picture to be displayed. For the above picture, the position coordinates is from right to left numbered from 0~143, a total of 144 lines. We setup the position pointer memory with the present display position. Additionally, the data pointer must be setup. According to the position pointer we request (table or calculate) the corresponding required LED RAM address and then according to the data pointer, obtain the required picture data. Then the data is written to the corresponding LED RAM which will be displayed. To display the next line, the position and data pointers must be increased by 1. This is repeated until all the picture data has been displayed. In this way the LED display can display a picture at any position.

Moving the Graphic on the Display

- Picture moving from right to left
  The principle of moving the picture is as follows. First clear the presently displayed picture. Then change the picture position pointer to the new picture position. In this way, in a fixed time interval, the displayed picture process will generate a moving effect on the screen. The picture will move from right to left, and at each time interval the position pointer should increase by 1. Of course when the position pointer increases, the picture size will change. When the position pointer increases by one, then this is the displayed picture (line number) size. In this way, according to the time interval the position pointer will increase until line 143.

- Picture moving bottom to top
  The movement of a picture from the bottom to top is different from a picture moving from right to left. This method is that each time before data is to be written to the RAM the data must first be managed. For example, for a line if the data to be displayed is 10011001B, then the picture will move from the bottom to the top. First move the 10011001B data to the right 7-bit. When moving right the higher bits will be filled in as 0. Therefore the data becomes 00000001B. This data is written to the LED RAM. In the picture all data is managed in this way. The picture will be displayed starting from the lowest line. To display the lowest two line contents, then move the original data to the right by 6 bits with the higher bits being filled in as 0. Now the data will be 00000010B. This data is written to the LED RAM. In this way the lowest two line contents can be displayed. When the picture has been moved to the topmost position, then no more moving is required and the data can be directly written to the LED RAM.

- Other picture change display methods
  There are a many ways to display pictures on the LED display. For more methods you can consult the following examples.

HT1632 Application Example Introduction

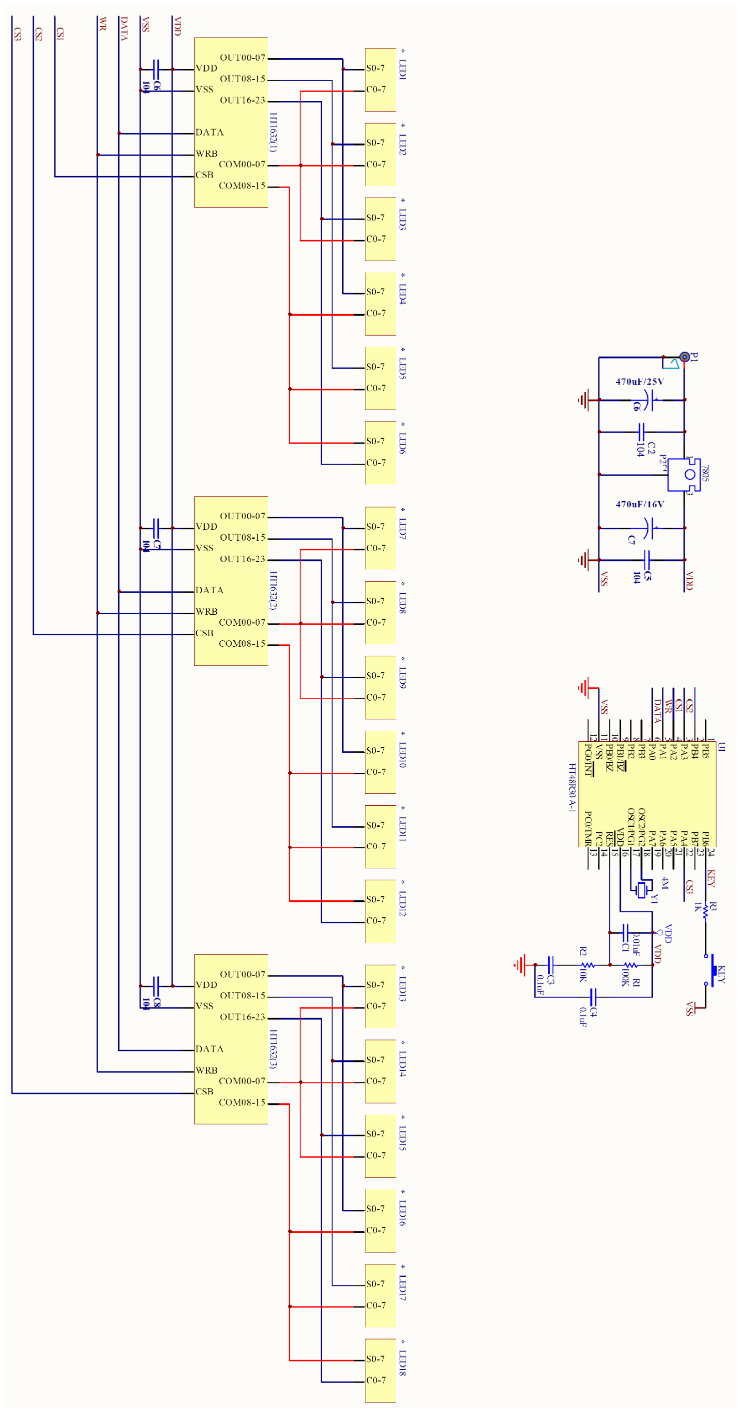
According to the above HT1632 explanation, an example is given below to provide further understanding of the HT1632 application. This example uses the Holtek C language and the MCU used is the HT48R30A-1.

DEMO Program Display Function

This DEMO program is for the above 144x8 LED display to implement the following displays and to show how the HT1632 has an LED display drive function.

- Place ″HOLTEK SEMICONDUCTOR INC.″ on the display, from right to left and gradually place in the centre of the display.
- Place ″HOLTEK SEMICONDUCTOR INC.″ on the display and flash
- Place ″HOLTEK SEMICONDUCTOR INC.″ on the display, move to the right and gradually show on the display
- Place ″HOLTEK SEMICONDUCTOR INC.″ on the display, from above and below two sides show on the screen.
- Place ″HOLTEK SEMICONDUCTOR INC.″ on the display, from the centre to the bottom and top two sides gradually extinguish
- Place ″HOLTEK SEMICONDUCTOR INC.″ on the display, from left and right two sides show on the screen centre
- Place ″HOLTEK SEMICONDUCTOR INC.″ on the display, from the centre to left and right two sides gradually extinguish
- Show HOLTEK and its LOG mark

DEMO Circuit



- Power supply section
  The Demo Board uses a 9V DC power supply. An LM7805 regulator is used to supply the 5V power for the HT48R30A-1 and the HT1632.
- HT48R30A-1 and HT1632 interface
  The HT48R30A-1 and the HT1632 interface have three lines, namely the write line (WR), the data line (DATA) and the select line (CS)
- The HT1632 power and ground lines have
  Because the display is large, the power and ground lines to the HT1632 must also be large. Therefore a 104 capacitor must be placed between VDD and VSS, like C6, C7 and C8.
- MCU oscillator and RESET circuit section
  - C1、C3、C4、R1、R2 form the Power On Reset circuit
  - Y1 is the crystal oscillator circuit

DEMO Software Description

- Document Structure
  - In the DEMO program there are four files, main.c, sub.c, 1632_driver.c and isr_tmr.c. Among these, sub.c, 1632_driver.c, isr_trm.c use INCLUDE method to include them into main.c.
  - In main.c is contained the HT48R30A-1 initialisation procedures (RAM, I/O ports, Timer setup and interrupt setup). At the same the HT1632 is also initialised.
  - The isr_tmr.c contains the timer interrupt service functions.
  - In 1632_driver.c is contained the HT48R30A-1 instructions for the HT1632.
  - In the DEMO program there are four files, main.c, sub.c, 1632_driver.c and isr_tmr.c. Among these, sub.c, 1632_driver.c, isr_trm.c use INCLUDE method to include them into main.c.
- Main variable description
  - M_TMR_MS
    The timer interrupt interval is 8ms, this variable is used to implement a 1 sec frequency divider.
  - M_TMR_SEC
    For each elapsed time of 1 sec this increases by one and is used to count seconds.
  - M_DATA_SPEED
    Increased by one every 8ms
  - M_Function_step
    Display function counter

- Main Function Description
  - Main( ) Function
  - Main( ) Function is the program start main function
  - Ini_Memory( ) Function
  - This function main purpose is to initialise the HT48R30A-1 Data Memory
  - Ini_System( ) Function
  - This function main purpose is the setup the I/O pins, timer and interrupt in the HT48R30A-1
  - Ini_1632( ) Function

    This function's main purpose is to initialise the HT1632. The initialisation process is as follows:

    1、 Write ″100″ to the HT1632 to setup the instruction mode
    2、 Write ″ 0x01″ to the HT1632 to turn on the HT1632 system clock
    3、 Write ″ 0x2c″ to the HT1632 COM lines selected as P-MOS outputs, COM lines selected as 16COM
    4、 Write ″ 0x03″ to the HT1632 - LED On
    5、 Write ″ 0x08″ to the HT1632 - set HT1632 blink off
    6、 Write ″0xaa″ to the HT1632 to set the LED illumination duty cycle as 10/16
    7、 Clear all of the HT1632 RAM to ″0″

    Here note that when initialising the HT1632, the 3 HT1632 device can be initialised together if all of the CS lines are enabled
  - SBR_DATA_DisplayCS( ) Function
  - The main purpose of this function is to drive the 6 LED displays. By calling this function, the 6 LED displays can display a picture. The main function entry point has two parameters, CSEn and Station. CSEn is used to indicate which HT1632 displayed picture, Station is used to indicate the picture position.
  - SBR_DATA_DisplayByte( ) Function
  - The main purpose of this function is for a HT1632 to drive 6 個 LED panels. By using this function call, the 6 LED panels can display a picture. This function entry point has three parameters, CSEn, Station and Data. CSEn is used to indicate which HT1632 displayed picture, Station is used to indicate the picture position and Data is used to display the data.
  - SBR_FUNCTION_STEP0( ) Function
  - The main purpose of this function is to move the picture from right to left
  - SBR_FUNCTION_STEP2( ) Function

- The main purpose of this function is to implement a flashing operation. The method of doing is to send an LED ON instruction to the HT1632 or to send an LED OFF instruction to the HT1632
- SBR_FUNCTION_STEP4( ) Function
- The main purpose of this function is to move the picture to the left
- BR_FUNCTION_STEP6( ) Function
- The main purpose of this function is to illuminate the LED panel starting from the top and bottom and moving towards the centre
- BR_FUNCTION_STEP8( ) Function
- The main purpose of this function is to gradually extinguish the panels from the centre to the top and bottom.
- BR_FUNCTION_STEP10( ) Function
- The main purpose of this function is to gradually illuminate the LED panels starting from both sides and moving towards the centre
- SBR_FUNCTION_STEP12( ) Function
- The main purpose of this function is to gradually extinguish the LED panels from the centre outwards
- BR_FUNCTION_STEP14( ) Function
- The main purpose of this function is to illuminate the 6 smallest left LED panels, illuminate the 6 smallest right LED panels, and extinguish the middle 6 LED panels
- BR_FUNCTION_STEP16( ) Function
- The main purpose of this function is to illuminate the 6 smallest left LED panels, illuminate the 6 smallest right LED panels, and on the middle 6 LED panels display the words ″HOLTEK″ and LOG
- BR_FUNCTION_STEP18( ) Function
- The main purpose of this function is to clear all of the LED RAM to ″0″

Conclusion

First the HT1632 characteristics were introduced along with how to use the HT1632 to drive a large LED display.

Secondly the HT1632 Memory Mapping LED drive method was introduced and methods for creating a moving display.

Finally a practical example was given to provide a fuller understanding of HT1632 applications.

Of course this document explanation only provides an introduction, however it is hoped that by extending the information provided users can use multiple HT1632 devices for larger LED display applications.