Dot Matrix LED Display Applications Using the HT1632C

D/N: AN0127E

Introduction

The HT1632C is a Memory Mapping LED display driver, widely applied for use in industrial instrumental displays such as digital clocks, thermometers, and hygrometers etc. The display size can be extended by cascading the HT1632C devices which is especially noticeable in the dot matrix LED display applications.

- HT1632C Operating Voltage: 2.4V~5.5V
- Multiple LED display (32 outbits & 8 commons or 24 outbits & 16 commons)
- Integrated display RAM (select 32 outbits & 8 commons for 64*4bit RAM, or select 24 outbits & 16 commons for 96*4bit RAM.)
- 16-level PWM brightness control
- Integrated 256K RC oscillator
- Serial MCU interface
- Command & Data mode instruction
- Selectable NMOS open drain output driver and PMOS open drain output driver for commons
Evaluating the HT1632C LED Display

The Dot Matrix LED Display Structure

![Diagram of LED Matrix Board](image)

The LED display to be driven by the HT1632C uses a 144x8 matrix display composed of 18 units. Each LED matrix board is shown above. The effect is shown below:

The HT1632C Device Selection Evaluation

The above content has stated that a single HT1632C includes a display size of 32*8 or 24*16. Here, when driving a 144x8 LED display, the HT1632C should choose the 24*16 display method so as to drive more LED dots. \( \frac{144 \times 8}{24 \times 16} = 3 \), thus using 3 HT1632C devices will be able to implement the above LED display drive function.

A single HT1632C can drive 24*16=384 LED dots. Each LED matrix board has 64 dots, therefore a single HT1632C can drive at least six smallest LED matrix boards. Three HT1632C devices will be capable of driving 18 LED matrix boards.

Furthermore, to drive more LED displays, more HT1632C devices will be required.
MCU Selection for the HT1632C

To choose an MCU, the number of selected figures and the variation method should be firstly decided before selecting the MCU. The more figures there are, the more ROM space it takes. The more figure variations there are, the more RAM size it takes. The HT1632C only requires three output lines from the MCU.

HT1632C LED Display Driving Method

HT1632C LED Matrix Display Driver Method

The HT1632C is a Memory Mapping LED display driver facilitating the driving of LED displays. Using a Memory Mapping LED driver for the figure display is equal to executing a logic writing operation to the LED RAM.

In an LED matrix display, all the displayed objects are called as figures. For example, the single word unit “H” is handled as one frame of a figure, so would for example the word “HOLTEK SEMICONDUCTOR INC.”

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.

Each figure in the selected MCU memory has a corresponding data unit. It is the function of the HT1632C to write this data into the RAM of the LED driver.

Corresponding Relation between LED Matrix Dots and LED RAMxx

The following figure shows the corresponding relation of the LED matrix dots and the LED RAM using a single HT1632C to drive six LED modules. In a general display condition, the LED RAM will form a table from which the position pointer will indicate the LED RAM in the table and then write the corresponding figure data to the LED RAM for display purposes.

LED_RAM_TABLE:

DC 00H, 01H, 04H, 05H, 09H, 0CH, 0DH, 10H, 11H, 14H, 15H, 18H, 19H, 1CH, 1DH
......
If the OUT lines and COM lines of the HT1632C have any connection changes with the LED matrix module, as shown in the figure below, then the order of the LED RAM in the LED table needs to be changed. The program does not require modification. From this point of view, using table read methods will be much more convenient.

**LED_RAM_TABLE:**

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


......

**Managing Figure Data**

The data for each figure is stored in the memory of the MCU. Creation of figure data should be done in such a way as make it as close to the LED RAM and LED matrix dots as possible. For example, the intersection of COM7 and OUT0 corresponds to bit 3 of the address 01H. The intersection of COM4 and OUT0 corresponds to bit 0 of the address 01H. The intersection of COM3 and OUT0 corresponds to bit 3 of the address 00H. The intersection of COM0 and OUT0 corresponds to bit 0 of the address 00H. The address bits of 1H and 00H are 8 bits in all. When creating figure data, the highest bit, bit7, should correspond to bit3 of 01H and the lowest bit, bit0, should correspond to bit 0 of 00H. In this way, it will be more convenient to write figure data into the LED RAM afterwards.
Writing Data to the HT1632C

The figure data is stored in 8-bit format in the MCU memory. However as every address bit in the LED RAM is only 4-bits, the successive address write mode in the HT1632C can be used to locate 8-bits of MCU data into two 4-bit RAM locations simply by writing to two successive RAM addresses as shown in the following figure. Note that the address high bits are written first while the data low bits are written first.

144x8 Large Screen Picture Display Method

The LED matrix display is actually a 144x8 matrix module as shown in the figure above. To show a figure on the LED matrix display, for example the word “HOLTEK”, the display location of the figure should be confirmed first as a figure composed of 0~143 columns, listed from right to the left. It is necessary to set a position pointer to memorize the current position to be displayed. Additionally, setup another Data Pointer, to follow the Position Pointer to keep track of (by table reading or calculation) the corresponding LED RAM address, and then write the figure data read from the Data Pointer into its corresponding LED RAM for display. To display the next row, simply add 1 to both the position pointer and data pointer repeatedly until all figure data is displayed completely. Any position in the LED matrix display shown above can thus display a figure.
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.
HT1632C Application Example Introduction

According to the above HT1632C explanation, an example is given below to provide further understanding of the HT1632C 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 HT1632C 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
• Power Supply
  The demo board uses DC 9V as the power supply which provides 5V voltage to the HT48R30A-1 and HT1632C after being regulated using an LM7805.

• HT48R30A-1 and HT1632C Interface
  A three line interface is used between the HT48R30A-1 and HT1632C: write line (WR), data line (DATA) and select line (CS).

• Decoupling capacitor between the HT1632C power supply and ground
  As the screen is of a larger size, the power and ground line routing to each HT1632C device will be longer, so it is necessary to add a 104 decoupling capacitor between VDD and VSS in the HT1632C as shown by C6, C7, and C8.

• MCU oscillator circuit and Reset circuit
  ▪ Y1 is the crystal oscillator circuit.
  ▪ C1, C3, C4, R1 and R2 form a Power On Reset circuit.

DEMO Software Description

• Document Structure
  – Four files are included in this demo program, main.c, sub.c, 1632_driver.c and isr_tmr.c, in which sub.c, 1632_driver.c and isr_tmr.c are enclosed within main.c using the INCLUDE function.
  – The main.c routine, executes an initialization to the HT48R30A-1 (RAM, I/O PORT, TIMER and interrupt setups etc.) and also to the HT1632C.
  – The isr_tmr.c routine mainly contains the timer interrupt service functions.
  – The 1632_driver.c routine, contains mainly the various command functions in the HT48R30A-1 to drive the HT1632C.
  – The sub.c routine, mainly contains functions regarding display figure transformation.

• Main variable description
  – M_TMR_MS
    Timer interrupt period interval is 8ms. Use this variable to execute a one second timer.
  – M_TMR_SEC
    Add “1” to the variable per second, used as a second counter.
  – M_DATA_SPEED
    Add “1” to the variable every 8ms.
  – M_Function_step
    The figure display function counter. Add 1 to the variable when a function is completed and then continue with the next function. The variable will be reset after all functions are displayed and re-started from the first function to display.
Main Function Description

- **Main() Function**
  Main() Function is the main starting function of the program.

- **Ini_Memory() Function**
  This function is used to initialize the RAM in the HT48R30A-1.

- **Ini_System() Function**
  This function is used to setup the I/O lines, timer and interrupt etc. in the HT48R30A-1.

- **Ini_1632() Function**
  The function is used to initialize the HT1632C. The initialization process is:
  1. Write 100 to the HT1632C and set as Command Mode.
  2. Write 0x01 to the HT1632C. Turn on the HT1632C System Clock.
  3. Write 0x2c to the HT1632C. Select the COM lines as P-MOS outputs. Set the number of the COM lines as 16COM.
  4. Write 0x03 to the HT1632C with LED on.
  5. Write 0x08 to the HT1632C. Set the HT1632C to Blink Off.
  6. Write 0xaa to the HT1632C. Set the Duty Circle of the LED brightness to 10/16.
  7. Clear all the RAM in the HT1632C to 0.

  Note: When initializing the HT1632C, it is acceptable to simultaneously enable the chip select signal, CS, in the three HT1632C MCUs for initialization.

- **SBR_DATA_DisplayCS() Function**
  The main function uses one HT1632C to drive six LED matrix module functions using call operations to display a figure on the six LED displays. There are two parameters for this function, CSEn and Station. CSEn is used to select the HT1632C that displays the figure. Station is used to indicate the location of the figure.

- **SBR_DATA_DisplayByte() Function**
  The main function uses one HT1632C to drive six LED matrix module functions, and display one row of a figure by calling those functions. There are three parameters for this function entry, CSEn, Station and Data. CSEn is used to select the HT1632C that displays the figure. Station indicates the location of the figure. Data indicates the data to be displayed.

- **SBR_FUNCTION_STEP0() Function**
  The main function is to shift in the figure from right to left on the screen.

- **SBR_FUNCTION_STEP2() Function**
  The main function is to enable figure blinking on the screen by transmitting an LED ON or LED OFF command to the HT1632C.

- **SBR_FUNCTION_STEP4() Function**
  The main function is to turn off the figure gradually to the left.
BR_FUNCTION_STEP6( ) Function
The main function is to illuminate the LED display starting from the top and bottom gradually to the centre.

BR_FUNCTION_STEP8( ) Function
The main function is to turn the LED display off from the centre to the top and the bottom.

BR_FUNCTION_STEP10( ) Function
The main function is to illuminate the LED display from the left and right sides to the centre.

SBR_FUNCTION_STEP12( ) Function
The main function is to turn the LED display off from the centre to the left and right sides.

BR_FUNCTION_STEP14( ) Function
The main function is to illuminate the six smallest LED matrix modules respectively on the left and right sides and turn the six LED matrix modules off in the centre.

BR_FUNCTION_STEP16( ) Function
The main function is to illuminate the six smallest LED matrix modules respectively on the left and right sides and display the words “HOLTEK” and “LOG” LED in the centre.

BR_FUNCTION_STEP18( ) Function
The main function is to clear the LED RAM to zero.

Conclusion

The above content has introduced the basic features of the HT1632C and shown a method of driving a larger LED matrix display using multiple HT1632C devices.

A method of displaying figures using the HT1632C as a Memory Mapping LED driver and shifting on the screen have also been described.

Finally, real examples were provided for users to have a better understanding of the HT1632C applications.

The descriptions above provide a good start point for users who wish to use the HT1632C to implement even larger display sizes with more animated figures.
Reference Document

Revision History

Revision: V1.10

Updated Date: October 12, 2010

Modified Contents:

- The product number has been updated to the HT1632C from the HT1632.
- The Application Circuit was modified.
- The Smallest LED Panel diagram was modified.
- In the LED Dot Matrix and LED RAM Relationship section, the diagram was modified.