

Volume

1

A-WIT TECHNOLOGIES INC.

... a passion for execution ...

7-Segment LED Display Driver Using the C Stamp

Version 1.0

A-WIT TECHNOLOGIES INC.

7-Segment LED Display Driver Using the C Stamp

© A-WIT Technologies Inc.
Phone (800) 985-AWIT • Fax (800) 985-2948

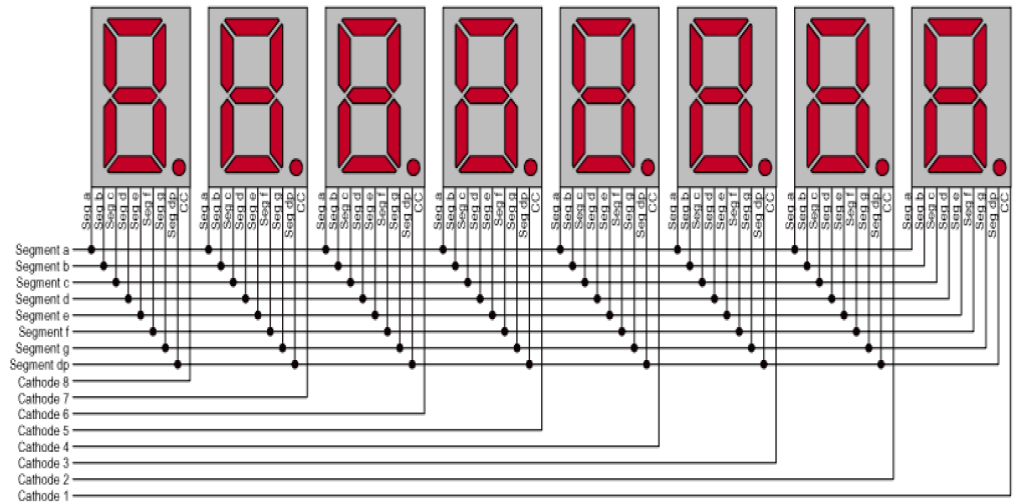
Table of Contents

Getting Started	1
Notices	4
Getting Support	4
C Stamp Program Example for the LED	
Segments Displays	4
Downloading and Running Your Program	7
Developing Your Own Programs and	
Projects	7
Terms and Conditions	8



Introduction

There are serially interfaced 7-segment LED display drivers that interface microcontrollers to 7-segment numeric LED displays. This software uses a method called “Charlieplexing” developed by Charles Allen at the Maxim Corporation. The implementation of Charlieplexing calls for the driving of LED digits using the smallest number of I/O pins on the microcontroller as possible. Additionally, the LED's must be turned on and off at a rate at which a human cannot notice any flicker, usually anywhere from 30-50Hz. Charlieplexing allows for N I/O pins to control $N*(N-1)$ LED's. At any particular time, only one of the LED segments is turned on. Immediately after the segment is lit, the segment is turned off and the next segment in the implementation is turned on. The following figure is an example of the implementation of Charlieplexing.

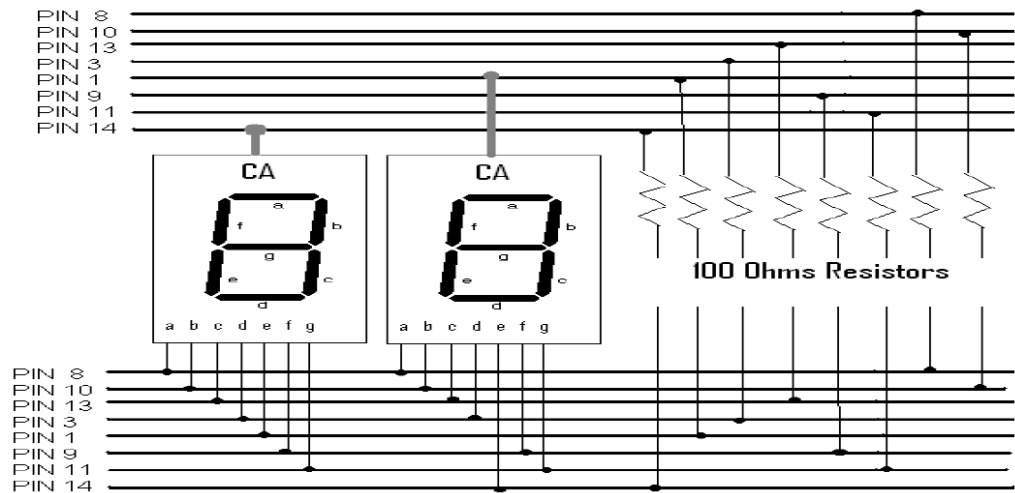


Getting Started

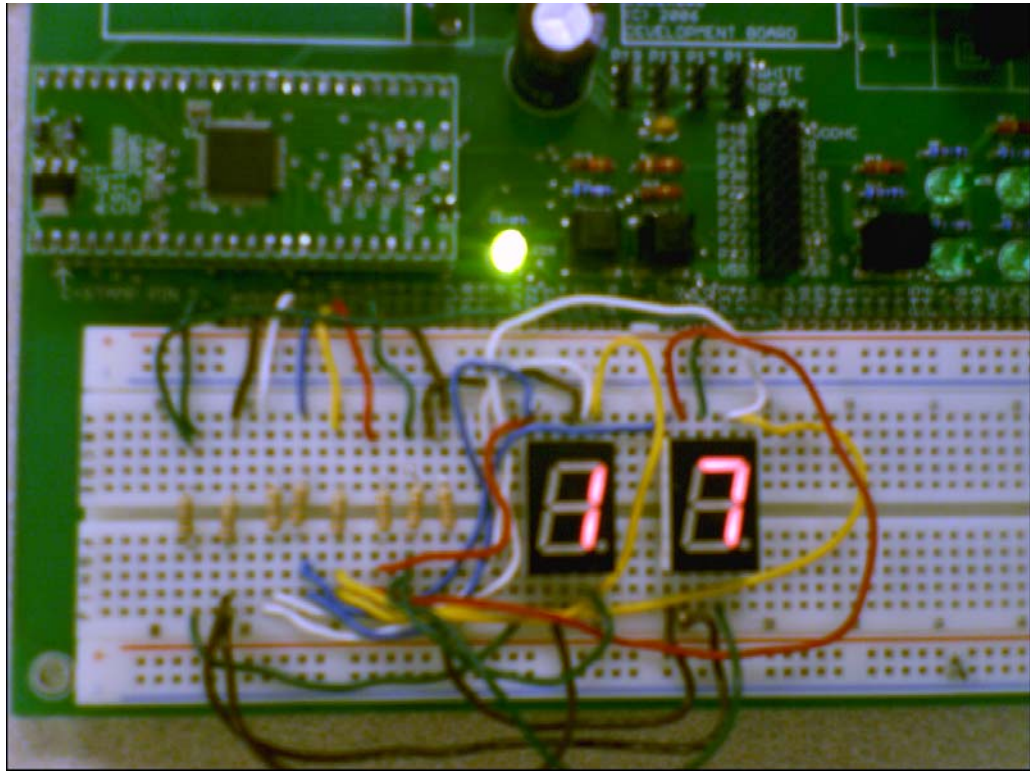
In this specific example, we show the design of a 2 Digit display driver using the Charlieplexing scheme. Using the formula $N*(N-1)$, it was determined that

mathematically, only 5 I/O pins were necessary to drive 2 LED's (16 total segments). However, a simple problem within the design prevented this from being properly implemented. It was discovered that when using only 5 I/O pins, the common anode must be switched during the process of lighting the LED. When the common anode is set to HIGH with another pin set to LOW, it results in two pins being connected to the LED, causing 2 segments to light up simultaneously, as opposed to one segment. It was then determined that at least 8 I/O pins were necessary to drive any number of seven segment LED's properly. Therefore using 5 I/O pins to drive 2 digits cannot be done.

The goal of this design was to implement the Charlieplexing scheme to drive 2 LED displays. Common anode LED displays were used in this implementation. All resistors in this design were 100 Ohms, which allowed for a peak current of 50mA with our 5V supply. The calculated refresh rate was 50Hz per segment to avoid any flicker. The figures below show the Charlieplexing schematic used as well as a picture of the implementation.



Below, one can see the LED display while the software is in operation.



The C Stamp design was able to blank the display on power up, and allowed for individual LED segment control. Individual LED's were also successfully addressed and updated without rewriting the entire display. Finally, a master/slave scheme was also successfully implemented, giving the user full LED segment control, using the RS-232 telecommunication protocol.

Notices

CSTAMP™ and CSTAMP™ Related Hardware Products, Software Products and Documentation are developed and distributed by A-WIT Technologies, Inc. All rights reserved by A-WIT Technologies, Inc. A-WIT SOFTWARE OR FIRMWARE AND LITERATURE IS PROVIDED “AS IS,” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL A-WIT BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY ARISING OUT OF OR IN CONNECTION WITH THE SOFTWARE OR FIRMWARE OR THE USE OF OTHER DEALINGS IN THE SOFTWARE OR FIRMWARE.

MPLAB C-18 and MPLAB C-18 Users Guide is reproduced and distributed by A-WIT Technologies, Inc. under license from Microchip Technology Inc. All rights reserved by Microchip Technology Inc. MICROCHIP SOFTWARE OR FIRMWARE AND LITERATURE IS PROVIDED “AS IS,” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL MICROCHIP BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY ARISING OUT OF OR IN CONNECTION WITH THE SOFTWARE OR FIRMWARE OR THE USE OF OTHER DEALINGS IN THE SOFTWARE OR FIRMWARE.

Getting Support

If possible, please check the C Stamp website www.c-stamp.com under SUPPORT for any updates to documentation, changes, or notices that may have become available since your Installation CD was produced. If you continue to have any issues for which a solution is not found in the aforementioned website, please e-mail tech_support@a-wit.com for help.

C Stamp Program Example for the LED Segments Displays

```
#include "CS110000 . h"

// Prototypes
void setupLED(BYTE sevseg, BYTE ledseg);
```

```

void writeLEDs(BYTE data[2]);

// Cathodes for each seven segment in turn
const BYTE SEVSEGS_CAS[2] = {14, 1};

// PINS that CONTROL EACH OF THE DIFFERENT SEGMENTS OF
// LED
const BYTE SEVSEGS_LDS[2][7] = {
    {8, 10, 13, 3, 1, 11, 9},
    {8, 10, 13, 3, 14, 11, 9}
};

// Segments to turn on for each number
// (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
const BIT NUMBERS[10][7] = {{1, 1, 1, 1, 1, 1, 0},
                             {0, 1, 1, 0, 0, 0, 0},
                             {1, 1, 0, 1, 1, 0, 1},
                             {1, 1, 1, 1, 0, 0, 1},
                             {0, 1, 1, 0, 0, 1, 1},
                             {1, 0, 1, 1, 0, 1, 1},
                             {1, 0, 1, 1, 1, 1, 1},
                             {1, 1, 1, 0, 0, 0, 0},
                             {1, 1, 1, 1, 1, 1, 1},
                             {1, 1, 1, 1, 0, 1, 1}};

void main(void){
// Initializing VALUES to be used
    NIBBLE result;
    RAM BYTE buffer[255];
    BYTE hello[] = "Enter 2 numbers:\n\r";
    BYTE data[2];
    BYTE i;
    BYTE j;

// SEROUT and SERIN will be used to tell the chip what
// 2 digits to display
    SEROUT(0, 0, 9.6, 0, 8, 0, 0, hello, 18);
    PAUSE 100);
    result = SERIN(0, 0, 9.6, 8, 0, 0, buffer, 2, 0);
    data[0] = buffer[0] - 48;
    data[1] = buffer[1] - 48;

// Turning the PIN to GTPIND if doesn't have to be lit
    for(j=0; j<2; j=j+1){
        for(i=0; i<7; i=i+1)

```

```

        GTPIND(SEVSEGS_LDS[j][i]);
    }
    for(i=0; i<2; i=i+1)
        GTPIND(SEVSEGS_CAS[i]);

    while(TRUE)
        writeLEDs(data);
}

// Light up segment ledseg in 7-segment LED sevseg
void setupLED(BYTE sevseg, BYTE ledseg){
// Setting Common Anode for the correct PIN
    STPIND(SEVSEGS_CAS[sevseg], HIGH);

//Turning Correct segments on
    STPIND(SEVSEGS_LDS[sevseg][ledseg], LOW);
    PAUSE(1);
    STPIND(SEVSEGS_LDS[sevseg][ledseg], HIGH);
    PAUSEUS(1);

    GTPIND(SEVSEGS_LDS[sevseg][ledseg]);
    GTPIND(SEVSEGS_CAS[sevseg]);
}

// Write 8 numbers to 8 LEDs, one segment at a time
void writeLEDs (BYTE data[2]){
// Declaring Variables Used in Function
    BYTE seg , chip;
    BYTE numtowrite;
    BIT segison;

// Looping through each of the different Segments of
// the LED
    for(seg=0; seg<7; seg=seg+1){
// Looping through each LED chip
        for(chip=0; chip<2 ; chip=chip+1){
// Extracting the Number to be written
            numtowrite = data[chip];
// Extracting whether the Segment
            segison = NUMBERS[numtowrite][seg];
// should be on or off
            if(segison)
// Turns on the LED Segment by calling setUpLED
// function
                setupLED(chip, seg);

```

```
    }  
  }  
}
```

Downloading and Running Your Program

Download and start the program by pushing and letting go of the RESET button while pushing the START button. Then you can let go of the START button.

Developing Your Own Programs and Projects

Now that you have successfully developed and run your program, it is easy to move on to more complex and elaborate projects and circuits of your own.

Terms and Conditions

Quality Assurance

A-WIT has stringent quality control procedures in place to insure the best quality products.

90-Day Limited Warranty

A-WIT Technologies, Inc warrants its products against defects in materials and workmanship for a period of 90 days. If you discover a defect, A-WIT Technologies, Inc. will, at its option, repair, replace, or refund the purchase price. After 90 days, products can still be sent in for repair or replacement, but there will be a \$10.00USD minimum inspection/labor/repair fee (not including return shipping and handling charges).

14-Day Money-Back Guarantee

If, within 14 days of having received your product, you find that it does not suit your needs, you may return it for a refund. A-WIT will refund the purchase price of the product in the form of a check, excluding shipping/handling costs, once the product is received. This refund does not apply if the product has been altered or damaged. If you decide to return the products after the 14-day evaluation period, a 20% restocking fee will be charged against a credit.

Disclaimer

Warranty does not apply if the product has been altered, modified, or damaged. A-WIT makes no other warranty of any kind, expressed or implied, including any warranty of merchantability, fitness of the product for any particular purpose even if that purpose is known to A-WIT, or any warranty relating to patents, trademarks, copyrights or other intellectual property. A-WIT shall not be liable for any injury, loss, damage, or loss of profits resulting from the handling or use of the product shipped.

How to Return a Product

When returning, you must first e-mail sales@a-wit.com for a Return Merchandise Authorization number. No packages will be accepted without the RMA number clearly marked on the outside of the package. After inspecting and testing, we will return your product, or its replacement using the same shipping method used to ship the product to A-WIT within 30 days. In your package, please include a daytime telephone number and a brief explanation of the problem.

Please contact our Sales Department at sales@a-wit.com if you have any questions regarding our warranty policy or if you are requesting an RMA number.

