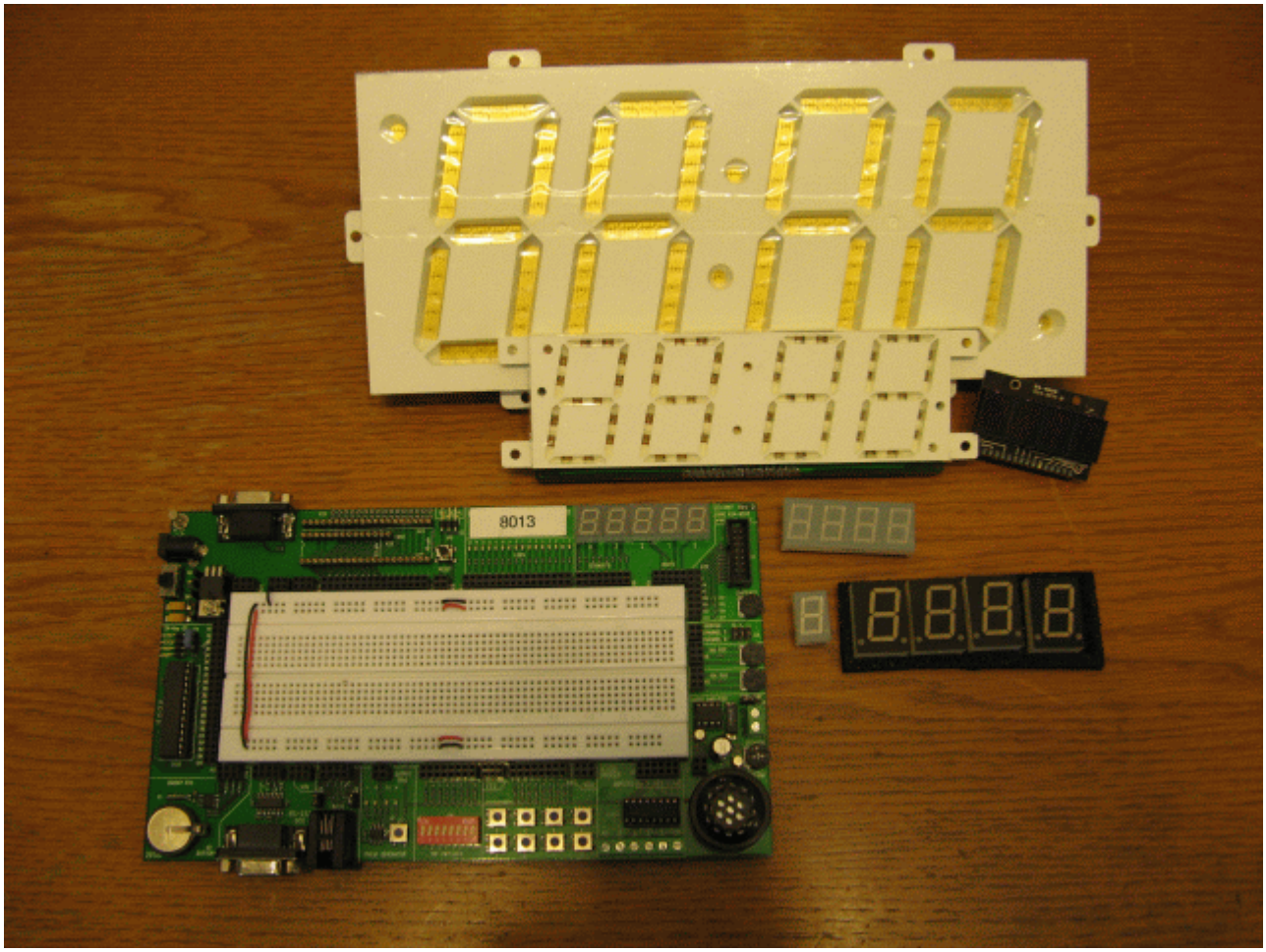
Tutorial SX28 - 7seg LED w/ MAX7219

Errata: 04/25/2020 - Added info on 10K ISET resistor on pin 18.

What is a 7 Segment Digit?

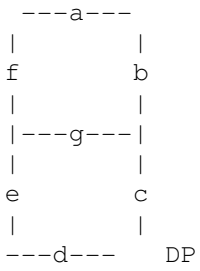
A **seven segment display** is actually made up of several LEDs arranged in a package to enable displaying digits. Some displays have an eighth LED for a decimal point and other displays are "14 segments" so that all of the letters can be displayed. Still other displays have several LEDs per segment so that they are brighter or larger. The large display in the picture below has 5 LEDs per segment. This means that to light it up you need more than 5vdc.



An assortment of 7 segment displays

Typical 7 Segment Displays

All 7 segment displays use a letter system to identify each segment. The diagram below illustrates how each segment is labeled. If you have a display that has a decimal point and four digits that means that you need to be able to individually address 32 individual LED segments (4 digits * 8 segments per digit). This requires that the microcontroller (MCU) dedicate 32 pins to controlling those 7 segment displays.



A 7 segment digit has at least 9 pins and are typically referred to as "common anode" and in the table below are shown as having a bus polarity of "CA". This means that the anode side of each LED segment is connected to a single pin and the cathode side is connected to a specific pin.

Here are a couple of examples of a Lite-On LTS-4801 display:

Link to distributor	Stock#	Bus Polarity	Color	Price
Jameco	1955722	CA	Green	$0.89
Mouser	859-LTS-4801JG	CA	Green	$1.16

A second type is the "common cathode" where the cathodes are tied together.

Finally, some 7 segment displays bring all leads out to pins so that it can be used either way. These are nice for the versatility but a pain to wire.

Most of the time you want "common anode" because an MCU can "sink" (bring the pin to zero) more current than it can "source" (supplying 5vdc to the pin).

For example, to display a "2" on the display, you need to light up segments A,B,G,E and D. You can either figure that out for each digit or you can use a display driver that does it for you. Additionally, a small microcontroller is often not able to handle the combined current drain of several digits when displaying an "8" let alone when trying to display four "8"s. To solve the problem the MCU displays each LED segment for a very brief moment and that tricks the eye to thinking that all of the segments are lit at the same time (look up "persistence of vision"). To do that takes lots of processing in the MCU because it needs to light one segment briefly, light the next one briefly, etc. All that processing power could be better spent doing more useful things like reading sensors, doing calculations, etc. The problem can be solved by using a chip called an LED Driver which relieves the MCU of the mundane task of refreshing the LED segments and communicates with the MCU using only a couple of pins.

In this tutorial we will look at the MAX7219 chip. This device can not only control EIGHT 7-segment displays but EACH of the 64 LEDs can be controlled individually and it only requires three pins on your MCU. Additional benefits are that the MAX7219 can control the intensity of the digits and convert common BCD format into which segments need to be lit up.

Hardware Connections

Ok, to follow along, download the datasheet on the Jameco part above (which is technically a Lite-on LTS-4801). This has the following pinout:

```
Pin   Function
 1    Cathode G                      ---A---
 2    Cathode F                     |       |
 3    Common Anode                  F       B
 4    Cathode E                     |       |
 5    Cathode D                     |---G---|
 6    Cathode DP (decimal point)    |       |
 7    Cathode C                     E       C
 8    Common Anode                  |       |
 9    Cathode B                      ---D---  DP
10    Cathode A
```

The MAX7219 is designed to handle up to 8 digits and up to 8 segments. That means it can handle 64 individual LED segments! Here is the MAX7219 pinout:

```
---------MAX7219----------
DIN   01|       |24 DOUT
DIG0  02|       |23 SEGD
DIG4  03|       |22 SEG DP
GND   04|       |21 SEGE
DIG6  05|       |20 SEGC
DIG2  06|       |19 V+
DIG3  07|       |18 ISET
DIG7  08|       |17 SEGG
GND   09|       |16 SEGB
DIG5  10|       |15 SEGF
DIG1  11|       |14 SEGA
LOAD  12|       |13 CLK
------------------------
```

The last diagram is the pinout of SX28.

```
 ----------SX28---------
 RTCC  01|      |28 MCLR
 VDD   02|      |27 OSC1
 n.c.  03|      |26 OSC2
 VSS   04|      |25 RC7
 n.c.  05|      |24 RC6
 RA0   06|      |23 RC5
 RA1   07|      |22 RC4
 RA2   08|      |21 RC3
 RA3   09|      |20 RC2
 RB0   10|      |19 RC1
 RB1   11|      |18 RC0
 RB2   12|      |17 RB7
 RB3   13|      |16 RB6
 RB4   14|      |15 RB5
```

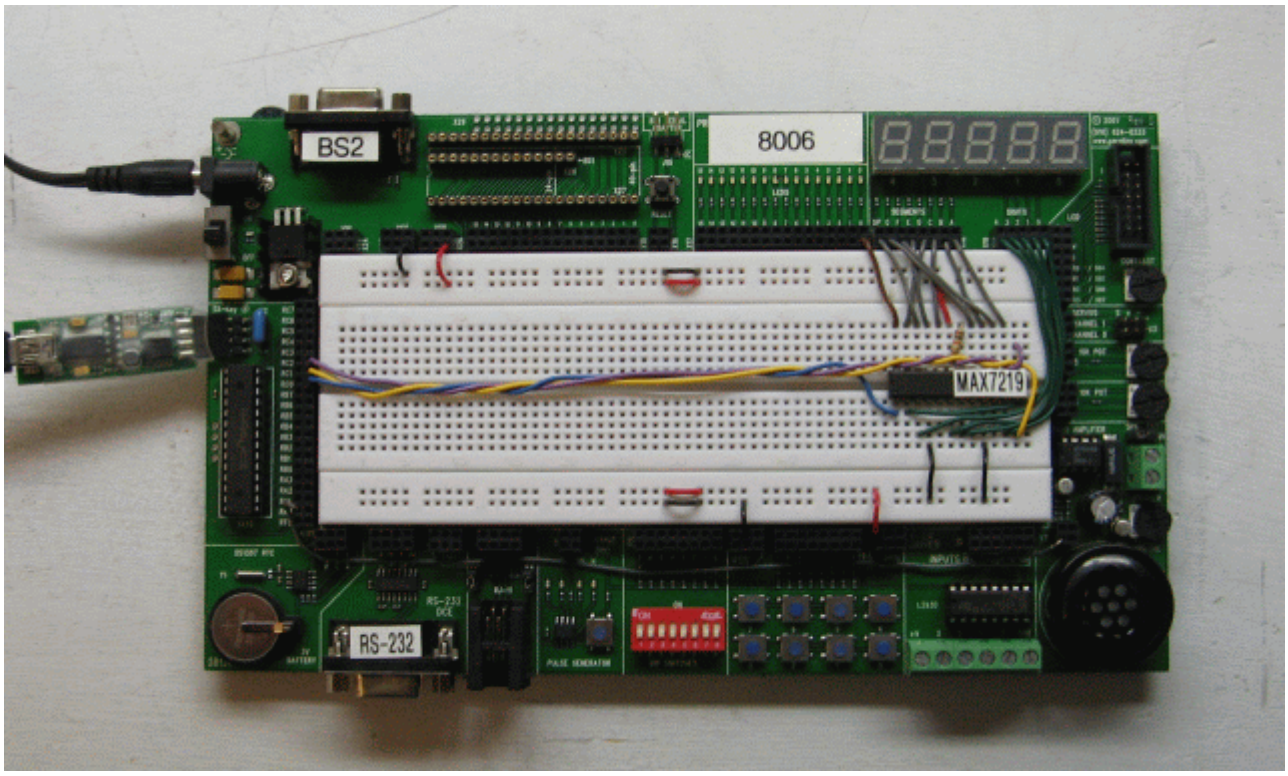

_____

I used the Professional Development Board to test my program but if you do not have one of those you can just tie all of the pins that represent "segment A" on your 7 segment displays together and all of the pins that represent "segment B" together, etc. The "segment A" pins are then connected to the MAX7219 pin 14. The "segment B" pins are then connected to the MAX7219 at pin 09, etc. The "common anode" on each of the digits is individually wired to the MAX7219. Digit one is wired to pin 11 on the MAX7218. Note that many 7 segment displays have TWO grounds. In the case of the LTE-4801 shown above, it has a ground connection on pin 3 and pin 8 and you need to connect both or the display will not operate correctly.

From	Pin	Function	To	Pin	Comments
SX28	06	RA0	your piezo		
MAX7219	01	(DIN)	SX28	18	RC0
MAX7219	12	(LOAD)	SX28	19	RC1
MAX7219	13	(CLK)	SX28	20	RC2
MAX7219	14	(SEGA)	Display	10	(segment A)
MAX7219	16	(SEGB)	Display	09	(segment B)
MAX7219	20	(SEGC)	Display	07	(segment C)
MAX7219	23	(SEGD)	Display	05	(segment D)
MAX7219	21	(SEGE)	Display	04	(segment E)
MAX7219	15	(SEGF)	Display	02	(segment F)
MAX7219	17	(SEGG)	Display	01	(segment G)
MAX7219	22	(SEG DP)	Display	06	(segment DP)
MAX7219	11	(DIG1)	Display	3,8	(digit 1 common anode)
MAX7219	06	(DIG2)	Display	3,8	(digit 2 common anode)
MAX7219	07	(DIG3)	Display	3,8	(digit 3 common anode)
MAX7219	03	(DIG4)	Display	3,8	(digit 4 common anode)
MAX7219	04,09	GND			Connect to Ground
MAX7219	19	Vdd			Connect to 5vdc
MAX7219	18	(ISET)			Connect a 10K resistor to Vdd

On the professional Development Board, here is how I wired it up:

- The PDB has five digits so I wired that one up also.
- Note that I set it up so "Digit 1" is on the far right and "Digit 5" is on the far left. Some tutorials reverse that order. It doesn't matter as long a you remember which one is which.
- I also connected an LED to pin 24 of the MAX7219. This is the "Data Output" pin and you can use it to twinkle an led to verify that you are successfully getting data into the MAX7219.
- Finally, pin 18 has a 10K resistor which is connected to 5vdc. This helps to regulate the amount of current each LED draws.



Setting up the SX28

Now we are FINALLY ready to do some programming! Let's keep it simple.

First I started out by defining which pins I was going to use:

```
TX7219_DQ VAR RC.0    ' Data In - MAX7219 pin 01 - blue wire
TX7219_Load VAR RC.1  ' Load    - MAX7219 pin 12 - yellow wire
TX7219_Clk  VAR RC.2  ' Clock   - MAX7219 pin 13 - purple wire
```

Next I declared some variables. I wanted to put the "transmit" functionality into a subroutine so that it would use up less memory. I designed the subroutine so that it would take two parameters:

- Function (or digit location)
- Data

When you first start up the MAX7219 you have to initialize it so that first parameter identifies which function I want to initialize in the chip and the second parameter is the data I want to send to that function. Later on, when I want to just transmit my numbers to the chip, the first parameter will be the position and the second digit will be the value. First, though I have to set aside some room for variables in my program.

```
TX7219_Function   VAR  Byte
TX7219_Data       VAR  Byte
```

Next I need to indicate that I want to use subroutines in my code.

```
Init_7219      SUB    ' Initialize the MAX7219
TX_7219        SUB 2  ' Transmit data to the MAX7219
```


Initializing the MAX7219

Here you will initialize the MAX7219. I choose to call a subroutine mainly because I don't like to clutter up the beginning of my programs. The initialize subroutine does the following:

- **Decode Mode** - This is telling us that the MAX7219 is going to accept numbers and the 7219 will figure out what segments need to be lit. This takes a tremendous amount of stain off of the main MCU. Thus, I will eventually send a "2" to the MAX7219 and the MAX7219 will lite up segments A,B,G,E, and D. Very kool feature!
- **Segment Luminosity Intensity** - I just set it to full on. However, you could set it so that your display can be dimmed by setting something in the MCU. An example might be where you build a clock and you want the clock to sense the room light and automatically dim in the middle of the night.
- **Scan Limit** - I have five digits on the PDB so I set my program to 5. You want to set it to the least number of digits so that you do not have to wait for the MAX7219 to update digits that do not exist.
- **Shutdown Register** - when the MAX7219 first boots up is presets itself with several features, most of which are useless. This command simply says, I'm done setting the configuration, now implement them and go into "normal" mode.
- **Test Mode** - you may or may not need this functionality but you can set up the MAX7219 to turn on all of the segments so that you can verify that the segments all work. Here I am just crudely turning it on for three seconds and then returning the display to normal mode.

Using the MAX7219

To use the MAX7219 you have to bring the LOAD pin LOW, shift the bits into the latch register on the MAX7219 and then bring the LOAD pin high to tell it you are done. This actually takes longer to talk about than to do. Keep in mind that you need to first tell the MAX7219 the function/position then you need to give it the data. Thus, you are sending 16 bits down the Data pin for each digit you want to update.


```
DEVICE SX28,OSC4MHZ,TURBO,STACKX,OPTIONX
FREQ 4_000_000
Piezo             VAR  RA.0      ' Speaker
TX7219_DQ         VAR  RC.0      ' Data In to MAX7219 pin 01 (BLUE)
TX7219_LOAD       VAR  RC.1      ' LOAD    to MAX7219 pin 12 (YELLOW)
TX7219_Clk        VAR  RC.2      ' Clock   to MAX7219 pin 13 (PURPLE)
                                 ' Note: SCL must be one higher than SDA
TX7219_Function   VAR     byte
TX7219_Data       VAR     byte
Init_7219         SUB            ' Initialize the MAX7219
TX_7219           SUB 2          ' send a digit to MAX7219
PROGRAM Start
Start:
   ra=0                  ' Initialize Port A
   rc=0                  ' Initialize Port C
   tris_a=%0000          ' Set pin direction, 0=output, 1=inputs
   tris_c=%00000000      ' Set pin direction, 0=output, 1=inputs

   Init_7219             ' Initialize the 7219
   SOUND Piezo,100,10    ' Send sound to indicate start of program
Main:
   TX_7219 $05,$0F      ' Blank Digit
   TX_7219 $04,$07      ' Display a 7
   TX_7219 $03,$02      ' Display a 2
   TX_7219 $02,$01      ' Display a 1
   TX_7219 $01,$09      ' Display a 9
   goto main
Init_7219:              ' Initialize subroutine (see datasheet for tables)
   TX_7219 $09,$FF      ' Decode Mode set to Code B for digits 7-0 (Table 4, pg 7)
   TX_7219 $0A,$0F      ' Segment luminosity intensity - max on    (Table 7, pg 9)
   TX_7219 $0B,$05      ' Set scan limit = FIVE digits on PDB      (Table 8, pg 9)
   TX_7219 $0C,$01      ' Set shutdown register to normal operation(Table 3, pg 7)
   TX_7219 $0F,%01      ' Turn on test mode (lite all segments)
   PAUSE 1000
   PAUSE 1000
   PAUSE 1000
   TX_7219 $0F,$00       ' Turn off Test mode
   ' Misc Notes: MAX7219  H=$0C, E=$0B, L=$0D, P=$0E  -=$0A  blank=$0F
   RETURN
' -----------------------------------------------------------------------
' TX_7219
' -----------------------------------------------------------------------
' Use: TX_7219 function (or position), data
' --- function = which digit or a control setting (table 2 in datasheet)
' --- data is BCD code of digit
' --- alternatively the first parameter can be the POSITION of the digit.
'     Depending on how you wire your 7 seg display (from left to right or
'     from right to left) will determine your 1 digit and your 2 digit, etc.
' -----------------------------------------------------------------------
TX_7219:
   TX7219_Function  = __param1
   TX7219_Data      = __param2
   TX7219_LOAD      = 0 ' setT LOAD LOW to allow MAX7219 to receive data
   SHIFTOUT TX7219_DQ, TX7219_Clk, MSBFIRST, TX7219_Function
   SHIFTOUT TX7219_DQ, TX7219_Clk, MSBFIRST, TX7219_Data
   TX7219_LOAD      = 1 ' set LOAD HIGH to latch data and send to display
   RETURN
```

