Experiment #29: Advanced 7-Segment Multiplexing

This experiment demonstrates the use of seven-segment displays with an external multiplexing controller. Multi-digit seven-segment displays are frequently used on vending machines to display the amount of money entered.

Building The Circuit (Note that schematic is NOT chip-centric)
Experiment #29: Advanced Seven-Segment Multiplexing

[Diagram of a seven-segment display with labels A to G and Pins 1 to 22, showing connections for Digit 1 to Digit 4.]

Page 172 • StampWorks Manual Version 1.2
Experiment #29: Advanced 7-Segment Multiplexing

File...... Ex29 - Change Counter.BS2
Purpose... Controlling 7-segment displays with MAX7219
Author.... Parallax
E-mail.... stamptech@parallaxinc.com
Started...
Updated... 01 MAY 2002

($STAMP BS2)

Program Description

This program is a coin counter -- it will count pennies, nickels, dimes and quarters using pushbutton inputs. The "bank" is displayed on four 7-segment LED displays that are controlled with a MAX7219.

Revision History

I/O Definitions

DataPin    CON    7       ' data pin (MAX7219.1)
Clock      CON    6       ' clock pin (MAX7219.13)
Load       CON    5       ' load pin (MAX7219.12)
Coins      VAR    InL     ' coin count inputs

Constants

Decode    CON    $09    ' bcd decode register
Brite     CON    $0A    ' intensity register
Scan      CON    $0B    ' scan limit register
ShutDn    CON    $0C    ' shutdown register (1 = on)
Test      CON    $0F    ' display test mode
Experiment #29: Advanced Seven-Segment Multiplexing

DecPnt  CON  %10000000
Blank    CON  %1111                   ' blank a digit
Yes     CON  1
No      CON  0

'  ________________________________________________________________
'  Variables
'  ________________________________________________________________

money   VAR     Word                    ' current money count
deposit VAR     Byte                    ' coins deposited
penny   VAR     deposit.Bit0           ' bit values of deposit
nickel  VAR     deposit.Bit1
dime    VAR     deposit.Bit2
quarter VAR     deposit.Bit3
dollar  VAR     deposit.Bit4
digit   VAR     Nib                     ' display digit
d7219   VAR     Byte                    ' data for MAX7219
index   VAR     Nib                     ' loop counter
idxOdd  VAR     index.Bit0             ' is index odd? (1 = yes)

'  ________________________________________________________________
'  EEPROM Data
'  ________________________________________________________________

'  Segments               .abcdefg
'  --------------------
Full     DATA    %01000111                       ' F
DATA    %00111110                       ' U
DATA    %00001110                       ' L
DATA    %00001110                       ' L

'  ________________________________________________________________
'  Initialization
'  ________________________________________________________________

Initialize:
  DirL = %11100000                              ' data, clock and load as outs
  coins as inputs

FOR index = 0 TO 7
  LOOKUP index, [Scan, 3, Brite, 5, Decode, $0F, ShutDn, 1], d7219
  SHIFTOUT DataPin, Clock, MSBFirst, [d7219]
Experiment #29: Advanced 7-Segment Multiplexing

IF (idxOdd = No) THEN No_Load
PULSOUT Load, 5                       ' load parameter

No_Load:
    NEXT

GOSUB Show_The_Money

' ------------------------------------------------------------------------
' Program Code
' ------------------------------------------------------------------------

Main:
    GOSUB Get_Coins
    IF (deposit = 0) THEN Main                    ' wait for coins

    money = money + (penny * 1)                   ' add coins
    money = money + (nickel * 5)
    money = money + (dime * 10)
    money = money + (quarter * 25)
    money = money + (dollar * 100)

    GOSUB Show_The_Money                          ' update the display
    PAUSE 100
    GOTO Main

' ------------------------------------------------------------------------
' Subroutines
' ------------------------------------------------------------------------

Get_Coins:
    deposit = %00011111                           ' enable all coin inputs
    FOR index = 1 TO 10
        deposit = deposit & ~Coins                  ' test inputs
        PAUSE 5                                     ' delay between tests
    NEXT
    RETURN

Show_The_Money:
    IF (money >= 9999) THEN Show_Full
    FOR index = 4 TO 1
        d7219 = Blank
        IF ((index = 4) AND (money < 1000)) THEN Put_Digit
        d7219 = money DIG (index - 1)
Experiment #29: Advanced Seven-Segment Multiplexing

```
IF (index <> 3) THEN Put_Digit
   d7219 = d7219 | DecPnt                      ' decimal point on DIGIT 3

Put_Digit:
   SHIFTOUT DataPin, Clock, MSBFirst, [index, d7219]
   PULSOUT Load, 5
   NEXT
   RETURN

Show_Full:
   ' turn BCD decoding off
   SHIFTOUT DataPin, Clock, MSBFirst, [Decode, 0]
   PULSOUT Load, 5
   FOR index = 4 TO 1
      READ (4 - index + Full), d7219              ' read and send letter
      SHIFTOUT DataPin, Clock, MSBFirst, [index, d7219]
      PULSOUT Load, 5
   NEXT
END
```

Behind The Scenes

Multiplexing multiple seven-segment displays requires a lot of effort that consumes most of the computational resources of the BASIC Stamp. Enter the MAXIM MAX7219 LED display driver. Using just three of the BASIC Stamp’s I/O lines, the MAX7219 can be used to control up to eight, seven-segment displays or 64 discrete LEDs (four times the number of I/O pins available on the BASIC Stamp).

The MAX7219 connects to the LED displays in a straightforward way; pins SEG A through SEG G and SEG DP connect to segments A through G and the decimal point of all of the common-cathode displays. Pins DIGIT 0 through DIGIT 7 connect to the individual cathodes of each of the displays. If you use less than eight digits, omit the highest digit numbers. For example, this experiment uses four digits, numbered 0 through 3, not 4 through 7.

The MAX7219 has a scan-limit feature than limits display scanning to digits 0 through $n$, where $n$ is the highest digit number. This feature ensures that the chip doesn't waste time and duty cycles (brightness) trying to scan digits that aren't there.
When the MAX7219 is used with seven-segment displays, it can be configured to automatically convert binary-coded decimal (BCD) values into appropriate patterns of segments. This makes the display of decimal numbers simple. The BCD decoding feature can be disabled to display custom patterns. This experiment does both.

From a software standpoint, driving the MAX7219 requires the controller to:

- Shift 16 data bits out to the device, MSB first.
- Pulse the Load line to transfer the data.

Each 16-bit data package consists of a register address followed by data to store to that register. For example, the 16-bit value $0407 (hex) writes a “7” to the fourth digit of the display. If BCD decoding is turned on for that digit, the numeral “7” will appear on that digit of the display. If decoding is not turned on, three LEDs will light, corresponding to segments G, F, and E.

In this experiment, the MAX7219 is initialized to:

- Scan = 3 (Display digits 0 – 3)
- Brightness = 5
- Decode = $0F (BCD decode digits 0 – 3)
- Shutdown = 1 (normal operation)

Initialization of the MAX7219 is handled by a loop. Each pass through the loop reads a register address or data value from a lookup table. After each data value is shifted out, the address and data are latched into the MAX7219 by pulsing the Load line.

Most of the work takes place in the subroutine called `Show_The_Money`. When the money count is less than 9999, the value will be displayed on the seven-segment digits, otherwise the display will read “FULL.” The routine scans through each digit of money and sends the digit position and value (from the DIG operator) to the MAX7219. Since the display shows dollars and cents, the decimal point on the third digit is enabled. When the position and digit have been shifted out, the display is updated by pulsing the Load line. To keep the display neat, the leading zero is blanked when the money value is less than 1000.

When the value of money reaches 9999, the display will change to “FULL.” This is accomplished by disabling the BCD decoding of the MAX7219 and sending custom letter patterns to the MAX7219. These patterns are stored in DATA statements.
Experiment #29: Advanced Seven-Segment Multiplexing

The main loop of the program is simple: it scans the switch inputs with `Get_Coins` and updates the money count for each switch pressed. This particular code is an excellent example of using variable aliases for readability.

**Challenge**

Modify the code in experiment 27 to display the input voltage on the seven-segment displays.