AppKit:
Using the MAX7219 LED Display Driver

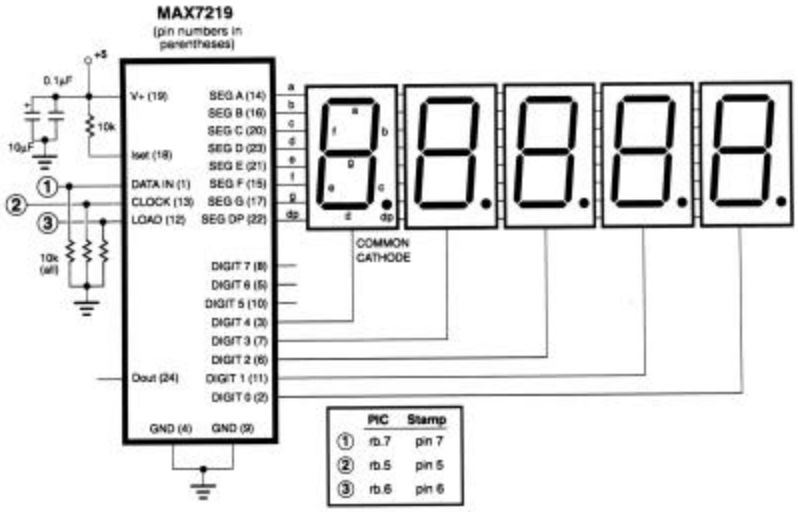
This AppKit shows how to use the Maxim MAX7219 LED display driver chip with PIC microcontrollers and Parallax BASIC Stamp® single-board computers.

Description

The MAX7219 drives common-cathode LED displays from one to eight seven-segment digits in length. It can also be used to drive up to 64 discrete LEDs configured as eight common-cathode clusters of eight LEDs each.

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.

Built-in pulse-width modulation and current-limiting circuits control the brightness of the displays with only a single external resistor. With eight display digits, the MAX7219 scans the display at approximately 1200 Hz to prevent visible flicker of the display. If a display of less than eight digits is used, the driver can be programmed to scan only the digits actually connected to it, increasing the potential brightness and scanning frequency of the display.



Hookup diagram for the PIC and Stamp demo programs

Hardware interface

The MAX7219 interfaces with controllers through three pins: data in, clock, and load. It 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, the applications demonstrated here use five digits, numbered 0 through 4, not 3 through 7. As you will see in the program examples, 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.

As shown in the figure, the manufacturer recommends a 10-µF bypass capacitor (aluminum electrolytic) in parallel with a 0.1-µF ceramic capacitor to ensure that the MAX7219's transient current requirements are met. Stamp users should not attempt to run a MAX7219-driven display from the Stamp's built-in voltage regulator. With eight segments driven the circuit shown can draw 80 to 320 mA—far beyond that regulator's maximum rating of 50 mA.

To restrict the maximum current (and therefore the maximum brightness of the display) to a lower value, increase the value of the 10k resistor from V+ to Iset in the schematic. Table 11 in the manufacturer's documentation (page 20 of this package) provides suggested resistances in 1000-ohm units for segment currents of 10 to 40 mA at various LED forward-voltages. To determine overall maximum current draw, multiply the segment current by eight. For example, assuming an LED forward drop of 2 volts (typical for red LEDs is 1.7), and a desired segment current of 10 mA, Table 11 suggests a 56k resistor. Maximum LED current draw would be 10 mA times eight = 80 mA. You may use a pot to permit adjustment of the maximum brightness, but it should be in series with a 10k fixed resistor to avoid exceeding the MAX7219's maximum current capabilities.

The figure also shows pulldown resistors on inputs to the MAX7219. When power is first applied to PIC or Stamp controllers, or when they are reset, their I/O lines float. The MAX7219 can see this as valid data and display garbage until the PIC or Stamp gains control. The most common result is that the MAX7219 enters test mode with all segments lit at full brightness. This can drag down the output of a marginal power supply preventing the PIC or Stamp from ever getting properly powered up. The pulldown resistors prevent these problems. If you want to reduce the parts count in a design, try eliminating the pulldowns on the clock and load lines, but leave one on the data line. The rationale is that although the MAX7219 may still clock in bad data, it will not enter the test mode, which consists of all 1s.

Software interface

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

(1) Shift 16 data bits out to the device, msb first.
(2) 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 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.

A chart of the register addresses and their functions appears in Table 2 of the manufacturer's documentation on page 17 of this package. Each of the example programs also contains symbols for the most commonly used register addresses. One subtlety about the test-mode register: Once you have put the MAX7219 into test mode (by sending $0F01), the only way to restore normal operation is to write a zero to the test register; $0F00.

PIC Program Listing

```
; Program: MAX7219.SRC
; This program controls the MAX7219 LED display driver. It demonstrates
; the basics of communicating with the 7219, and shows a convenient
; method for storing setup data in tables.

; Hardware interface with the 7219:
DATA        =       rb.7            ; Bits are shifted out this pin to 7219.
CLK         =       rb.5            ; Data valid on rising edge of this clock pin.
Load        =       rb.6            ; Tells 7219 to transfer data to LEDs.

; The 7219 accepts 16-bit packets of data sent most-significant bit (msb)
; first. The upper byte of the packet is the address of a register within
; the 7219. These registers hold data to be displayed on the LEDs, or
; setup bits that define the 7219's operating mode.

; High-order byte opcodes for the MAX7219:
dig_0       =       1               ; addresses digit 0.
dig_1       =       2               ; "      " 1.
dig_2       =       3               ; "      " 2.
dig_3       =       4               ; "      " 3.
dig_4       =       5               ; "      " 4.
dig_5       =       6               ; "      " 5.
dig_6       =       7               ; "      " 6.
dig_7       =       8               ; "      " 7.
dcd         =       9               ; addresses the decode register.
brite       =       10              ; "      "          " intensity register.
scan        =       11              ; "      "          " scan-limit register.
switch      =       12              ; "      "          " on/off register.
test        =       15              ; activates test mode (all digits on, 100% bright)

            org     8
shifts      ds      1               ; Shift counter used for output to MAX7219.
max_reg     ds      1               ; Register address of MAX7219
max_arg     ds      1               ; Argument (data) for MAX7219.
temp        ds      1               ; Temporary counter used to index setup table.

; Device data and reset vector
            device  pic16c54,xt_osc,wdt_off,protect_off
            reset   start
            org     0
```

PIC Program Listing (cont)

```
; This table contains pairs of instructions and data to be sent to
; the 7219. The first three pairs set the scan-limit register for
; all digits, 0-7; the brightness to 10 (out of a possible 15);
; and enable BCD decoding of all digits. The next eight pairs
; write the digits "3.1415926" to the display. Note that setting
; bit 7 of a BCD digit (by adding or ORing with 128) turns on
; the decimal point for that digit.
; == In an actual application, you would probably want to fill the
; numeric entries of this table with some appropriate starting
; value for the display, such as blanks (15 decimal) or dashes
; (10 decimal) in all digits.

LED_setup
                jmp     pc+w            ; Jump into table based on index # in w.
                retw    scan,7          ; Scan digits 0 - 7.
                retw    brite,10        ; Max brightness is 15. (Very bright!)
                retw    dcd,11111111b   ; BCD decode all digits.
                retw    dig_7,3+128     ; "3."
                retw    dig_6,1         ; "1"
                retw    dig_5,4         ; "4"
                retw    dig_4,1         ; "1"
                retw    dig_3,5         ; "5"
                retw    dig_2,9         ; "9"
                retw    dig_1,2         ; "2"
                retw    dig_0,6         ; "6"
                retw    switch,1        ; Turn display on.

; This subroutine reads the parameters from LED_setup and uses them to
; initialize the LED driver when the hardware first wakes up.
LED_init
                clr     temp            ; Start with temp (the index #) = 0.
:loop           mov     w,temp          ; Move temp into w.
                call    LED_setup       ; And use w to retrieve value from table.
                mov     max_reg,w       ; Put table value into max_reg.
                inc     temp            ; Let temp=temp+1 to get next table entry.
                mov     w,temp          ; Move temp into w.
                call    LED_setup       ; Retrieve value from table.
                mov     max_arg,w       ; Put table value into max_arg.
                call    MAX_out         ; Send max_reg and max_arg to MAX7219.
                inc     temp            ; Point to next location in table.
                cjb     temp,#23,:loop  ; Have we reached the end of table?
                ret                     ; If so, return; else loop.
```

PIC Program Listing (cont)

```
; Transmits the 16-bit data packet stored in max_reg and max_arg
; to the MAX7219. After the routine executes, max_reg and max_arg
; are unchanged.

MAX_out
                mov     shifts,#16      ; Set up to send 16 bits.
:loop           clrb    CLK             ; End previous clock cycle.
                rl      max_arg         ; Rotate lower byte's msb into carry.
                rl      max_reg         ; Rotate carry into upper byte; msb to carry.
                movb    DATA,c          ; Move carry to data output.
                setb    CLK             ; Raise the clock.
                djnz    shifts,:loop    ; All 16 bits sent? If not, loop.
                setb    load            ; Bits sent. Pulse load pin on MAX7219.
                rl      max_arg         ; Rotate bytes one more time to return..
                rl      max_reg         ; ..them to their original states.
                clrb    load            ; Finish load pulse.
                ret                     ; Return.

start           clr     rb              ; Start with rb bits low.
                mov     !ra,#0          ; Set data direction to output.
                mov     !rb,#0          ; "             "
                call    LED_init        ; Send the data from the table.
                jmp     $               ; Done: endless loop.
```

BASIC Stamp I (BS1-IC) and BASIC Stamp Ver. D Program Listing

```
' Program: MAX7219.BAS
' This program controls the MAX7219 LED display driver. It demonstrates
' the basics of communicating with the 7219, and shows a convenient
' method for storing setup data in tables. To demonstrate practical
' application of the 7219, the program drives a 5-digit display to
' show the current value of a 16-bit counter (0-65535). The subroutines
' are not specialized for counting; they can display _any_ 16-bit
' value on the LCDs. (A specialized counting routine would be faster,
' since it would only update the digits necessary to maintain the
' count; however, it wouldn't be usable for displaying arbitrary
' 16-bit values, like the results of Pot, Pulsin, or an A-to-D
' conversion).

' Hardware interface with the 7219:
SYMBOL DATA_n           = 7             ' Bits are shifted out this pin # to 7219.
SYMBOL DATA_p           = pin7          ' "    "   "       "       ".
SYMBOL CLK = 5          ' Data valid on rising edge of this clock pin.
SYMBOL Load             = 6             ' Tells 7219 to transfer data to LEDs.

' Register addresses for the MAX7219. To control a given attribute
' of the display, for instance its brightness or the number shown
' in a particular digit, you write the register address followed
' by the data. For example, to set the brightness, you'd send
' 'brite' followed by a number from 0 (off) to 15 (100% bright).
SYMBOL dcd   = 9         ' Decode register; a 1 turns on BCD decoding.
SYMBOL brite  = 10       ' "       "   " intensity register.
SYMBOL scan = 11         ' "       "   " scan-limit register.
SYMBOL switch           = 12            ' "       "   " on/off register.
SYMBOL test   = 15       ' Activates test mode (all digits on, 100% bright)

' Variables used in the program.
SYMBOL max_dat          = b11           ' Byte to be sent to MAX7219.
SYMBOL index            = b2            ' Index into setup table.
SYMBOL nonZ             = bit1          ' Flag used in blanking leading zeros.
SYMBOL clocks           = b3            ' Bit counter used in Max_out.
SYMBOL dispVal          = w2            ' Value to be displayed on the LEDs.
SYMBOL decade           = w3            ' Power-of-10 divisor used to get decimal digits.
SYMBOL counter          = w4            ' The value to be displayed by the demo.

' The program begins by setting up all pins to output low, matching
' the state established by the pulldown resistors.
let port = $FF00            ' Dirs = $FF (all outputs) and Pins = 0 (low).
```

BASIC Stamp I and Ver. D Program Listing (cont.)

```
' Next, it initializes the MAX7219. A lookup table is convenient way
' to organize the setup data; each register address is paired with
' its setting data. The table sets the scan limit to 4 (5 digits,
' numbered 0-4); brightness to 3; BCD decoding to the lower 5 digits
' (the only ones we're displaying), and switches the display on. The
' MAX7219 expects data in 16-bit packets, but our lookup table holds
' a series of 8-bit values. That's why the loop below is designed to
' pulse the Load line _every_other_ byte transmitted.
for index = 0 to 7                          ' Retrieve 8 items from table.
  lookup index,(scan,4,brite,3,dcd,$1F,switch,1),max_dat
  gosub Max_out
  let bit0 = index & 1                      ' Look at lowest bit of index.
  if bit0 = 0  then noLoad
  pulsout Load,1                            ' If it's 1, pulse Load line.
NoLoad:                                           ' Else, don't pulse.
next                                        ' Get next item from table.

' ====================== MAIN PROGRAM LOOP ==========================
' Now that the MAX7219 is properly initialized, we're ready to send it
' data. The loop below increments a 16-bit counter and displays it on
' the LEDs connected to the MAX. Subroutines below handle the details
' of converting binary values to binary-coded decimal (BCD) digits and
' sending them to the MAX.
Loop:
  let dispVal = counter
  gosub MaxDisplay
  let counter = counter+1
goto loop

' ========================= SUBROUTINES ============================
' The MAX7219 won't accept a number like "2742" and display it on
' the LEDs. Instead, it expects the program to send it individual
' digits preceded by their position on the display. For example,
' "2742" on a five-digit display would be expressed as:
' "digit 5: blank; digit 4: 2; digit 3: 7; digit 2: 4; digit 1: 2"
' The routine MaxDisplay below does just that, separating a value
' into individual digits and sending them to the MAX7219. If the
' lefthand digits are zero (as in a number like "102") the
' routine sends blanks, not zeros until it encounters the first
' non-zero digit. This is called "leading-zero blanking."
MaxDisplay:
let decade = 10000                          ' Start with highest digit first.
let nonZ = 0                                ' Reset non-zero digit flag.
for index = 5 to 1 step -1       ' Work from digit 5 to digit 1.
  let max_dat = index                       ' Send the digit address.
  gosub Max_out
  let max_dat = dispVal/decade              ' Get the digit value (0-9).
```

BASIC Stamp I and Ver. D Program Listing (cont.)

```
  if max_dat = 0 then skip       ' If digit <> 0 then nonZ = 1.
  let nonZ = 1                                  ' If a non-zero digit has already
skip:                                           ' ..come, or the current digit is not
  if nonZ = 1 OR max_dat <> 0 OR index = 1 then skip2         '..0, or the
  let max_dat = 15                              '.._only_ digit is 0, send the digit,
skip2:                                          '..else send a blank.
  gosub Max_out                                       ' Send the data in max_dat and
  pulsout Load,1                                ' ..pulse the Load line.
  let dispVal = dispVal//decade                 ' Get the remainder of value/decade.
  let decade = decade/10       ' And go to the next smaller digit.
 next                                           ' Continue for all 5 digits.
return                                                ' Done? Return.

' Here's the code responsible for sending data to the MAX7219. It
' sends one byte at a time of the 16 bits that the MAX expects. The
' program that uses this routine is responsible for pulsing the
' Load line when all 16 bits have been sent. To talk to the MAX7219,
' Max_out places the high bit (msb) of max_dat on DATA_p, the data pin,
' then pulses the clock line. It shifts the next bit into position by
' multiplying max_dat by 2. It repeats this process eight times.
' In order to avoid hogging the bit-addressable space of w0, the
' routine uses a roundabout way to read the high bit of max_dat: if
' max_dat < $80 (%10000000) then the high bit must be 0, so a 0
' appears on DATA_p. If max_dat >= to $80, then a 1 appears on DATA_p.
Max_out:
for clocks = 1 to 8                             ' Send eight bits.
  let DATA_p = 0                                ' If msb of max_dat = 1, then let
  IF max_dat < $80 then skip3                   '..DATA_p = 1, else DATA_p = 0.
  let DATA_p = 1
skip3:
  pulsout CLK,1                                       ' Pulse the clock line.
  let max_dat = max_dat * 2    ' Shift max_dat one bit to the left.
next                                            ' Continue for eight bits.
return                                                ' Done? Return.
```

BASIC Stamp II (BS2-IC) Program Listing

```
' Program: MAX7219.BS2
' This program controls the MAX7219 LED display driver. It demonstrates
' the basics of communicating with the 7219, and shows a convenient
' method for storing setup data in tables. To demonstrate practical
' application of the 7219, the program drives a 5-digit display to
' show the current value of a 16-bit counter (0-65535). The subroutines
' are not specialized for counting; they can display _any_ 16-bit
' value on the LCDs. (A specialized counting routine would be faster,
' since it would only update the digits necessary to maintain the
' count; however, it wouldn't be usable for displaying arbitrary
' 16-bit values, like the results of Pot, Pulsin, or an A-to-D
' conversion).

' Hardware interface with the 7219:
DATA_n      con     7               ' Bits are shifted out this pin # to 7219.
CLK   con   5       ' Data valid on rising edge of this clock pin.
Load  con   6       ' Tells 7219 to transfer data to LEDs.

' Register addresses for the MAX7219. To control a given attribute
' of the display, for instance its brightness or the number shown
' in a particular digit, you write the register address followed
' by the data. For example, to set the brightness, you'd send
' 'brite' followed by a number from 0 (off) to 15 (100% bright).
decode      con     9               ' Decode register; a 1 turns on BCD decoding.
brite   con 10      ' "     "   " intensity register.
scan  con   11      ' "     "   " scan-limit register.
switch      con     12              ' "     "   " on/off register.
test    con 15      ' Activates test mode (all digits on, 100% bright)

' Variables used in the program.
max_dat     var     word            ' Word to be sent to MAX7219.
index       var     nib             ' Index into setup table.
temp  var   nib     ' Temporary variable used in outputting digits.
nonZ  var   bit     ' Flag used in blanking leading zeros.
dispVal     var     word            ' Value to be displayed on the LEDs.
odd   var   index.bit0  ' Lsb of index.

' The program begins by setting up all pins to output low, matching
' the state established by the pulldown resistors.
OUTS = 0                ' All output latches low.
DIRS = $FFFF            ' All pins to output.
```

BASIC STAMP II Program Listing (cont)

```
' Next, it initializes the MAX7219. A lookup table is convenient way
' to organize the setup data; each register address is paired with
' its setting data. The table sets the scan limit to 4 (5 digits,
' numbered 0-4); brightness to 3; BCD decoding to the lower 5 digits
' (the only ones we're displaying), and switches the display on. The
' MAX7219 expects data in 16-bit packets, but our lookup table holds
' a series of 8-bit values. That's why the loop below is designed to
' pulse the Load line _every_other_ byte transmitted.
for index = 0 to 7                              ' Retrieve 8 items from table.
  lookup index,[scan,4,brite,3,decode,$1F,switch,1],max_dat
  shiftout DATA_n,CLK,msbfirst,[max_dat]
  if odd = 0  then noLoad       ' If odd is 1, pulse Load line.
  pulsout Load,1
NoLoad:                                             ' Else, don't pulse.
next                                            ' Get next item from table.

' ====================== MAIN PROGRAM LOOP ==========================
' Now that the MAX7219 is properly initialized, we're ready to send it
' data. The loop below increments a 16-bit counter and displays it on
' the LEDs connected to the MAX. Subroutines below handle the details
' of converting binary values to binary-coded decimal (BCD) digits and
' sending them to the MAX.
Loop:
   gosub MaxDisplay
   dispVal = dispVal+1
goto loop

' ========================= SUBROUTINES ============================
' The MAX7219 won't accept a number like "2742" and display it on
' the LEDs. Instead, it expects the program to send it individual
' digits preceded by their position on the display. For example,
' "2742" on a five-digit display would be expressed as:
' "digit 5: blank; digit 4: 2; digit 3: 7; digit 2: 4; digit 1: 2"
' The routine MaxDisplay below does just that, separating a value
' into individual digits and sending them to the MAX7219. If the
' lefthand digits are zero (as in a number like "102") the
' routine sends blanks, not zeros until it encounters the first
' non-zero digit. This is called "leading-zero blanking."
MaxDisplay:
nonZ = 0                        ' Clear flag for 1st non-zero digit.
for index = 5 to 1              ' Work from digit 5 down to digit 1.
  shiftout DATA_n,CLK,msbfirst,[index]          ' Send digit position.
  temp = dispVal dig (index-1)                  ' Get decimal digit (0-4) of dispVal.
  if temp = 0 then skip1        ' If digit = 0, don't set nonZ flag.
  nonZ = 1
```

BASIC STAMP II Program Listing (cont)

```
skip1:
  if nonZ = 1 OR temp <> 0 OR index = 1 then skip2          ' If leading 0..
  temp = 15                                                 '..write a blank to the display.
skip2:
  shiftout DATA_n,CLK,msbfirst,[temp]                       ' Send the digit.
  pulsout Load,5                                            ' And load the display.
next                                                        ' Repeat for all 5 digits.
return                                                      ' Done? Return.
```

19-4452; Rev 1; 5/92

MAXIM

Serially Interfaced, 8-Digit LED Display Driver

General Description

The MAX7219 is a compact, serial input/output common-cathode display driver that interfaces microprocessors (µPs) to 7-segment numeric LED displays of up to 8 digits, bar-graph displays, or 64 individual LEDs. Included on-chip are a BCD code-B decoder, multiplex scan circuitry, segment and digit drivers, and an 8x8 static RAM that stores each digit. Only one external resistor is required to set the segment current for all LEDs.

A convenient 3-wire serial interface connects to all common µPs. Individual digits may be addressed and updated without rewriting the entire display. The MAX7219 also allows the user to select code-B decoding or no-decode for each digit.

The MAX7219 includes a 150µA low-power shutdown mode, analog and digital brightness control, a scan-limit register which allows the user to display from 1 to 8 digits, and a test mode which forces all LEDs on.

Applications

Bar-Graph Displays
7-Segment Displays
Industrial Controllers
Panel Meters
LED Matrix Displays

Features

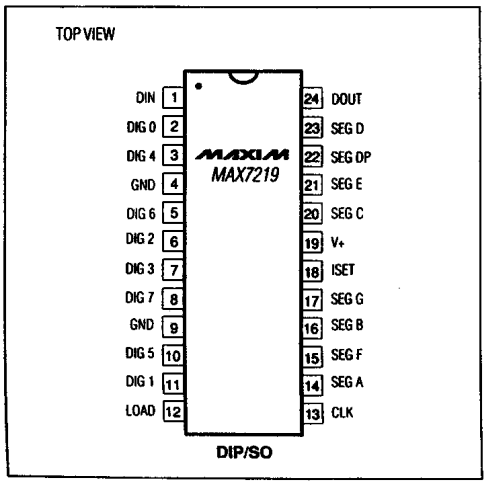
- ♦ **10MHz Serial Interface**
- ♦ **Individual LED Segment Control**
- ♦ **Decode/No-Decode Digit Selection**
- ♦ **150µA Low-Power Shutdown (Data Retained)**
- ♦ **Digital and Analog Brightness Control**
- ♦ **Display Blanked on Power-Up**
- ♦ **24-Pin DIP and SO Packages**
- ♦ **Drives Common-Cathode LED Display**

Ordering Information

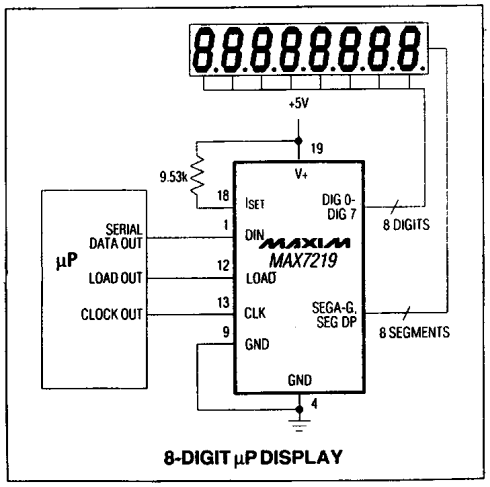
PART	TEMP. RANGE	PIN-PACKAGE
MAX7219CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX7219CWG	0°C to +70°C	24 Wide SO
MAX7219C/D	0°C to +70°C	Dice*
MAX7219ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX7219EWG	-40°C to +85°C	24 Wide SO
MAX7219ERG	-40°C to +85°C	24 Narrow CERDIP

* *Contact factory for dice specifications.*

Pin Configuration



Typical Application Circuit



Serially Interfaced, 8-Digit LED Display Driver

ABSOLUTE MAXIMUM RATINGS

V+ Voltage 7V
DIG0-DIG7 Sink Current 500mA
SEG A-G, DP Source Current 100mA
Input Voltage (any pin) ..... V+ + 0.3V to -0.3V
Continuous Power Dissipation (T_A = +85°C)
Narrow Plastic DIP 0.87W
Wide SO 0.76W
CERDIP 1.1W

Operating Temperature Ranges:
MAX7219C _ G ..... 0°C to +70°C
MAX7219E _ G ..... -40°C to +85°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10 sec) ..... +300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability

ELECTRICAL CHARACTERISTICS

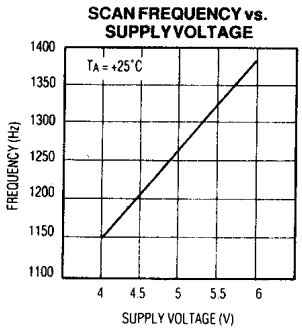
(V+ = 5V±10%, R_{SET} = 9.53kΩ ±1%, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

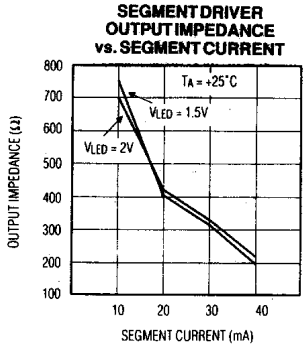
PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Operating Supply Voltage	V+		4.0		6.0	V
Shutdown Supply Current	I_Q	DIN,CLK and LOAD = GND or V+, shutdown register set to 0, T_A = +25°C			150	µA
Operating Supply Current	I_{OP}	R_{SET} = infinity			8	mA
		All segments and decimal points on, I_{SEGO} = -40mA		330		mA
Display Scan Rate	f_{OSC}	V+ = 5V, 8 digits scanned	500	1300	2000	Hz
Digit Drive Sink Current	I_{DIGI}	T_A = +25°C, V+ = 5V, V_{OUT} = 0.65V	320			mA
Segment Drive Current Source	I_{SEGO}	T_A = +25°C, V+ = 5V, V_{OUT} =V+ - 1V	-30	-37	-40	mA
Segment Drive Current Matching				3.0		%
Digit Drive Source Current	I_{DIGO}	Digit off, V_{OUT} = V+ - 0.3V	-2			mA
Segment Drive Current Sink	I_{SEGI}	Segment off, V_{OUT} = 0.3V	5			mA
LOGIC INPUTS						
Input Current	I	DIN, CLK and LOAD V_{IN} = 0V V_{IN} = V+			-1 1	µA
Logic 1 Input Voltage	V_{IH}		3.5			V
Logic 0 Input Voltage	V_{IL}				0.8	V
Hysteresis Voltage		DIN, CLK, and LOAD		1.0		V
Output High Voltage	V_{OH}	DOUT I_{OUT} = -1mA, I_{OUT} -1µA	V+-1.0	V+		V
Output Low Voltage	V_{OL}	DOUT, I_{OUT} = 1.6mA			0.4	V
Data-Hold Time DATAIN to Clock	t_{IDH}		0	-5		ns
Data-Setup Time DATAIN To Clock	t_{IDS}		25			ns
Clock-to-Serial Output Prop Delay	t_{OPD}	C_{LOAD} = 50pF			25	ns
Clock Low Time	t_{CKL}		50			ns
Clock High Time	t_{CKH}		50			ns
Data-to-Segment Prop Delay (Note 1)	t_{DSPD}	C_{LOAD} = 50pF	0		2.25	ms
Load-Rising Edge to Next Clock Rising Edge	t_{LDCK}		50			ns
Clock-to-Load Rising Edge Setup Time	t_{CKLD}		0			ns
Load Low Time	t_{LDL}		50			ns
Load High Time	t_{LDH}		50			ns

Note 1: Guaranteed by design.

Serially Interfaced, 8-Digit LED Display Driver

Typical Operating Characteristics





Pin Description

PIN	NAME	FUNCTION
1	DIN	Serial Data Input. Data is loaded into an internal 16-bit shift register on the rising edge of CLK.
2, 3, 5-8, 10, 11	DIG0-7	8 digit drive lines that sink current from the display.
4, 9	GND	Ground (both GND pins must be connected)
12	LOAD	Load Data Input. On LOAD's rising edge, the last 16 bits of serial input data are latched.
13	CLK	Clock Input. 10MHz maximum rate. On CLK's rising edge, data is shifted into the internal shift register. On CLK's falling edge, data is clocked out of DOUT.
14-17, 20-23	SEG A-G, DP	7-segment drive and decimal point lines that source current to the display.
18	ISET	Connect to V+ through a resistor (R_{SET}) to set the peak segment current (Refer to "*Selecting R_{SET} Resistor*" section).
19	V+	Supply Voltage
24	DOUT	Serial Data Output. The data into DIN is valid at DOUT 16.5 clock cycles later.

Serially Interfaced, 8-Digit LED Display Driver

Block Diagram

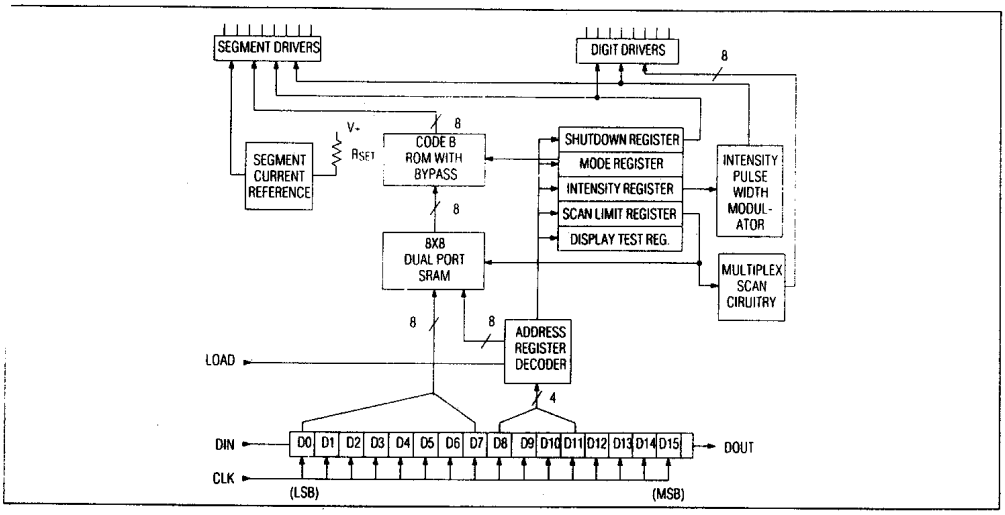


TABLE 1. SERIAL DATA FORMAT (16 BITS)

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	ADDRESS				MSB	DATA						LSB

X = "don't care" bit

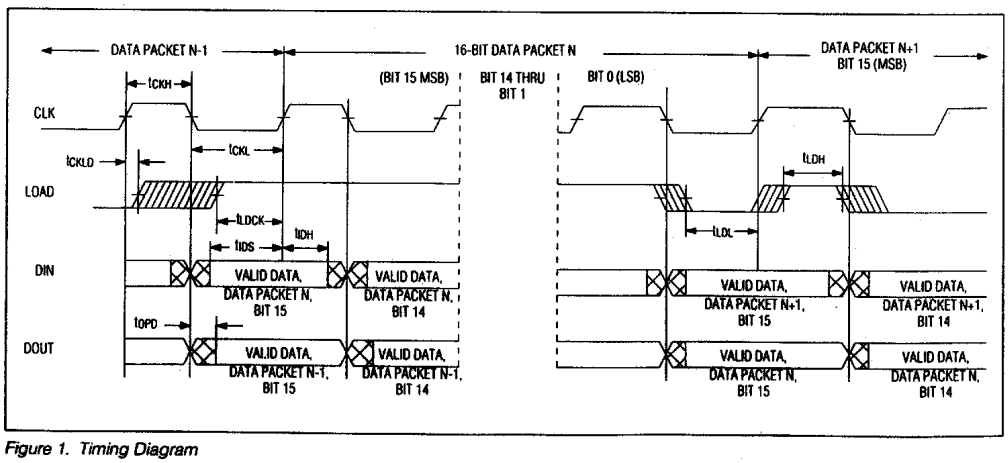


Figure 1. Timing Diagram

Serially Interfaced, 8-Digit LED Display Driver

Detailed Description

Serial Addressing Modes

Serial data at DIN, sent in 16-bit packets, is shifted into the internal 16-bit shift register with each rising edge of CLK. The data is then latched into either the digit or control registers on the rising edge of LOAD. LOAD must go high concurrently with or after the 16th rising clock edge, but before the next rising clock edge or data will be lost. Data at DIN is propagated through the shift register and appears at DOUT 16.5 clock cycles later. Data is clocked out on the falling edge of CLK. Data bits are labeled D0-D15 (Table 1). D8-D11 contain the register address, D0-D7 contain the data, and D12-D15 are "don't care" bits. The first bit received is D15, the most significant bit (MSB).

Digit and Control Registers

Table 2 lists the 14 addressable digit and control registers. The digit registers are realized with an on-chip, 8x8 dual-port SRAM. They are addressed directly so that individual digits can be updated and retain data as long as V+ typically exceeds 2V. The control registers consist of: decode mode, display intensity, scan limit (number of scanned digits), shutdown, and display test (all LEDs on). A no-operation (no-op) register is also included, which allows data to be passed from DIN to DOUT when devices are cascaded without changing the display or affecting any control registers.

Shutdown Mode

When the MAX7219 is in shutdown mode, the scan oscillator is halted, all segment current sources are pulled to ground, and all digit drivers are pulled to V+, thereby blanking the display. Data in the digit and control registers remains unaltered. Shutdown can be used to save power or as an alarm to flash the display by successively entering and leaving the shutdown mode. For minimum supply current in shutdown mode, logic inputs should be at ground or V+ (CMOS logic levels).

Typically, it takes less than 250µs for the MAX7219 to leave shutdown mode. Note that the display driver can still be programmed while in shutdown mode, and that shutdown mode can be overridden by the display-test function.

Table 2. Register Address Map

REGISTER	ADDRESS D15-D12	D11	D10	D9	D8	HEX CODE
NO-OP	X	0	0	0	0	X0
DIGIT 0	X	0	0	0	1	X1
DIGIT 1	X	0	0	1	0	X2
DIGIT 2	X	0	0	1	1	X3
DIGIT 3	X	0	1	0	0	X4
DIGIT 4	X	0	1	0	1	X5
DIGIT 5	X	0	1	1	0	X6
DIGIT 6	X	0	1	1	1	X7
DIGIT 7	X	1	0	0	0	X8
DECODE MODE	X	1	0	0	1	X9
INTENSITY	X	1	0	1	0	XA
SCAN LIMIT	X	1	0	1	1	XB
SHUTDOWN	X	1	1	0	0	XC
DISPLAY TEST	X	1	1	1	1	XF

Table 3. Shutdown Register Format (Address (Hex) = XC)

	ADDR CODE (HEX)	REGISTER DATA D7	D6	D5	D4	D3	D2	D1	D0
SHUTDOWN MODE	XC	X	X	X	X	X	X	X	0
NORMAL OPERATION	XC	X	X	X	X	X	X	X	1

Initial Power-Up

On initial power-up, all control registers are reset, the display is blanked, and the MAX7219 enters shutdown mode. Therefore the user must program the display driver prior to display use since it will initially be set to scan one digit, it will not decode data in the data registers, and the intensity register will be set to its minimum value.

Decode-Mode Register

The decode-mode register sets BCD code B (0-9, E, H, L, P, and –) or no-decode operation for each digit. Each bit in the register corresponds to one digit. A logic high selects code B decoding while a logic low bypasses the decoder. Examples of the decode mode control-register format are shown in Table 4.

Table 4. Decode-Mode Register Examples (Address (Hex) = X9)

	REGISTER DATA D7	D6	D5	D4	D3	D2	D1	D0	(HEX CODE)
NO DECODE FOR DIGITS 7-0	0	0	0	0	0	0	0	0	00
CODE B DECODE FOR DIGIT 0 NO DECODE FOR DIGITS 7-1	0	0	0	0	0	0	0	1	01
CODE B DECODE FOR DIGITS 3-0 NO DECODE FOR DIGITS 7-4	0	0	0	0	1	1	1	1	0F
CODE B DECODE FOR DIGITS 7-0	1	1	1	1	1	1	1	1	FF

Serially Interfaced, 8-Digit LED Display Driver

able 5. Code B Font

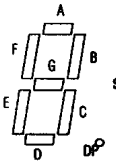
7-SEGMENT CHARACTER	REGISTER DATA						ON SEGMENTS = 1							
	D7*	D6-D4	D3	D2	D1	D0	DP*	A	B	C	D	E	F	G
0		X	0	0	0	0		1	1	1	1	1	1	0
1		X	0	0	0	1		0	1	1	0	0	0	0
2		X	0	0	1	0		1	1	0	1	1	0	1
3		X	0	0	1	1		1	1	1	1	0	0	1
4		X	0	1	0	0		0	1	1	0	0	1	1
5		X	0	1	0	1		1	0	1	1	0	1	1
6		X	0	1	1	0		1	0	1	1	1	1	1
7		X	0	1	1	1		1	1	1	0	0	0	0
8		X	1	0	0	0		1	1	1	1	1	1	1
9		X	1	0	0	1		1	1	1	1	0	1	1
-		X	1	0	1	0		0	0	0	0	0	0	1
E		X	1	0	1	1		1	0	0	1	1	1	1
H		X	1	1	0	0		0	1	1	0	1	1	1
L		X	1	1	0	1		0	0	0	1	1	1	0
P		X	1	1	1	0		1	1	0	0	1	1	1
blank		X	1	1	1	1		0	0	0	0	0	0	0

The decimal point is set by bit D7 = 1

When the code B decode mode is used, the decoder looks only at the lower nibble of the data in the digit registers (D3-D0), disregarding bits D4-D6. D7, which sets the decimal point (SEG DP), is independent of the decoder and is positive logic (D7=1 turns the decimal point on). The code-B font is listed in Table 5.

When no-decode is selected, data bits D7-D0 correspond to the segment lines of the MAX7219. Table 6 shows the one-to-one pairing of each data bit to the appropriate segment line.

Table 6. No-decode Mode Data Bits and Corresponding Segment Lines



STANDARD 7-SEGMENT LED

	REGISTER DATA							
	D7	D6	D5	D4	D3	D2	D1	D0
CORRESPONDING SEGMENT LINE	DP	A	B	C	D	E	F	G

Intensity Control and Interdigit Blanking

The MAX7219 allows the display brightness to be controlled with an external resistor (R_{SET}) connected between V+ and ISET, and digitally using the intensity register. The peak current sourced from the segment drivers will nominally be 100 times the current entering ISET. This resistor can either be fixed, or variable to allow brightness adjustment from the front panel. Its minimum value should be 9.53kΩ, which typically sets the segment current at 37mA.

Digital control of segment current is provided by an internal pulse-width modulated DAC, which is loaded from the lower nibble of the intensity register. The DAC scales the average segment current in 16 steps from a maximum of 31/32, down to 1/32 of the peak current set by R_{SET}. The intensity register format is listed in Table 7. Maximum brightness occurs with a duty cycle of 31/32 because the interdigit blanking time is set to 1/32 of a cycle. Interdigit blanking time can be increased by decreasing the duty cycle.

Serially Interfaced, 8-Digit LED Display Driver

Table 7. Intensity Register Format (Address (Hex) = XA)

DUTY CYCLE	REGISTER DATA D7	D6	D5	D4	D3	D2	D1	D0	(HEX CODE)
1/32 (min on)	X	X	X	X	0	0	0	0	X0
3/32	X	X	X	X	0	0	0	1	X1
5/32	X	X	X	X	0	0	1	0	X2
7/32	X	X	X	X	0	0	1	1	X3
9/32	X	X	X	X	0	1	0	0	X4
11/32	X	X	X	X	0	1	0	1	X5
13/32	X	X	X	X	0	1	1	0	X6
15/32	X	X	X	X	0	1	1	1	X7
17/32	X	X	X	X	1	0	0	0	X8
19/32	X	X	X	X	1	0	0	1	X9
21/32	X	X	X	X	1	0	1	0	XA
23/32	X	X	X	X	1	0	1	1	XB
25/32	X	X	X	X	1	1	0	0	XC
27/32	X	X	X	X	1	1	0	1	XD
29/32	X	X	X	X	1	1	1	0	XE
31/32 (max on)	X	X	X	X	1	1	1	1	XF

Scan-Limit Register

The scan-limit register sets how many digits are displayed, from 1 to 8. They are displayed in a multiplexed manner with a typical display scan rate of 1300Hz with 8 digits displayed. If fewer digits are displayed, the scan rate is $8f_{OSC}/N$, where N is the number of digits scanned. Since the number of scanned digits affects the display brightness, the scan-limit register should not be used to blank portions of the display (such as leading zero suppression). The scan-limit register format is listed in Table 8.

If the scan-limit register is set for three digits or less, individual digit drivers will dissipate excessive amounts of power. Consequently, the value of the R_{SET} resistor must be adjusted according to the number of digits displayed, to limit individual digit driver power dissipation. Table 9 lists the number of digits displayed and the corresponding maximum recommended segment current when the internal digit drivers are used.

Display-Test Register

The display-test register operates in two modes: normal and display test. Display-test mode turns all LEDs on by overriding – but not altering – all controls and digit registers (including the shutdown register). In display-test mode, 8 digits are scanned and the duty cycle is 31/32. Table 9 lists the display-test register format.

Table 8. Scan-Limit Register Format (Address (Hex) = XB)

	REGISTER DATA D7	D6	D5	D4	D3	D2	D1	D0	(HEX CODE)
*DISPLAY DIGIT 0 ONLY	X	X	X	X	X	0	0	0	X0
*DISPLAY DIGITS 0 & 1	X	X	X	X	X	0	0	1	X1
*DISPLAY DIGITS 0 1 2	X	X	X	X	X	0	1	0	X2
DISPLAY DIGITS 0 1 2 3	X	X	X	X	X	0	1	1	X3
DISPLAY DIGITS 0 1 2 3 4	X	X	X	X	X	1	0	0	X4
DISPLAY DIGITS 0 1 2 3 4 5	X	X	X	X	X	1	0	1	X5
DISPLAY DIGITS 0 1 2 3 4 5 6	X	X	X	X	X	1	1	0	X6
DISPLAY DIGITS 0 1 2 3 4 5 6 7	X	X	X	X	X	1	1	1	X7

* See "*Scan-Limit Register*" text for application.

Table 9. Maximum Segment Current for 1, 2 or 3 Digit Displays

NUMBER OF DIGITS DISPLAYED	MAXIMUM SEGMENT CURRENT
1	10mA
2	20mA
3	30mA

Table 10. Display-Test Register Format (Address (Hex) = XF)

	REGISTER DATA D7	D6	D5	D4	D3	D2	D1	D0
NORMAL OPERATION	X	X	X	X	X	X	X	0
DISPLAY TEST MODE	X	X	X	X	X	X	X	1

Note: The MAX7219 remains in display-test mode (all LEDs on) until the display-test register is reconfigured for normal operation.

No-Op Register

The no-op register is used when cascading MAX7219s. Connect all devices' LOAD inputs together and connect DOUT to DIN on adjacent MAX7219s. DOUT is a CMOS logic level output that easily drives DIN of a successively cascaded MAX7219. Refer to the "*Serial Addressing Modes*" section for detailed information on serial input/output timing. For example, if four MAX7219s are cascaded, then to write to the fourth chip, send the desired 16-bit word, followed by three no-op codes (hex X0XX, see Table 2). When load goes high, data is latched in all devices. The first three chips receive no-op commands, and the fourth receives the intended data.

Serially Interfaced, 8-Digit LED Display Driver

Applications Information

Supply Bypassing and Wiring

To minimize power-supply ripple due to the peak digit driver currents, connect a 10µF electrolytic and a 0.1µF ceramic capacitor between V+ and GND as close to the device as possible. The MAX7219 should be placed in close proximity to the LED display, and connections should be kept as short as possible to minimize the effects of wiring inductance and electromagnetic interference. Also, both GND pins must be connected to ground.

Selecting R_{SET} Resistor and Using External Drivers

The current per segment is approximately 100 times the current in ISET. To select RSET, see Table 11. The MAX7219's maximum allowable segment current is 40mA. For an LED forward voltage drop of 2.5V, RSET must be greater than 9.53k. For segment current levels above the MAX7219 limits, external drivers will be needed. In this application, the MAX7219 serves as only a controller for other high-current drivers or transistors. Therefore, to conserve power in the MAX7219, use RSET = 47k when using external current sources as segment drivers.

The example in Figure 2 uses the MAX7219's segment drivers, a MAX333 single-pole double-throw analog switch, and external transistors to drive 4.0" AND4107SCL common-cathode displays. The 5.6V zener diode has been added in series with the decimal point LED because the decimal point LED forward voltage is typically 4.2V, while for all other segments the LED forward voltage is typically 8V. Note that since external transistors are used to sink current (DIG 0 and DIG 1 are used as logic switches), peak segment currents of 40mA are allowed even though only two digits are displayed. In applications where the MAX7219's digit drivers are used to sink current and fewer than four digits are displayed, see Table 9 which specifies the maximum allowable segment current. RSET must be selected accordingly (see Table 11).

Refer to the "Power Dissipation" section to calculate acceptable limits for ambient temperature, segment current, and the LED forward-voltage drop.

Table 11. RSET vs. Segment Current and LED Forward Voltage

	V_{LED} (V)				
I_{SEG} (mA)	1.5	2	2.5	3	3.5
40	11.3	10.4	9.8	8.9	7.8
30	16.3	15	14	12.9	11.4
20	26.2	24.6	22.8	20.9	18.6
10	60.1	56	51.7	47	41.9

Table 12. Package Thermal Resistance Data

PACKAGE	THERMAL RESISTANCE (θ_{JA})
24 Narrow DIP	+75°C/W
24 Wide SO	+85°C/W
24 CERDIP	+60°C/W
Maximum Junction Temperature (T_J) = +150°C	
Maximum Ambient Temperature (T_A) = +85°C	

Computing Power Dissipation

The upper limit for power dissipation (PD) for the MAX7219 is determined from the following equation:

$$PD = (V+ \times 8mA) + (V+ - V_{LED})(DUTY \times I_{SEG} \times N)$$

where:

V+ = Supply Voltage

DUTY = Duty Cycle set by intensity register

N = number of segments driven (worst case is 8)

V_{LED} = LED forward voltage

I_{SEG} = Segment Current set by RSET

Dissipation Example:

I_{SEG} = 40mA, N = 8, DUTY = 31/32, V_{LED} = 1.8V at 40mA, V+ = 5.25V

$$PD = 5.25V(8mA) + (5.25V - 1.8V)(31/32 \times 40mA \times 8)$$
$$= 1.11W$$

Thus, for a CERDIP package (θ_{JA} = +60C°/W from table 12), the maximum allowed ambient temperature T_A is given by:

$$T_J max = T_A + PD \times \theta_{JA}$$
$$+150°C = TA + 1.11W \times +60°C/W$$
$$T_A = +83.4°C$$

Cascading Drivers

The example in Figure 3 drives 16 digits using a 3-wire µP interface. If the number of digits is not a multiple of 8, set both drivers' scan-limit registers to the same number so one display will not appear brighter than the other. For example, if 12 digits are needed, use 6 digits per display

Serially Interfaced, 8-Digit LED Display Driver

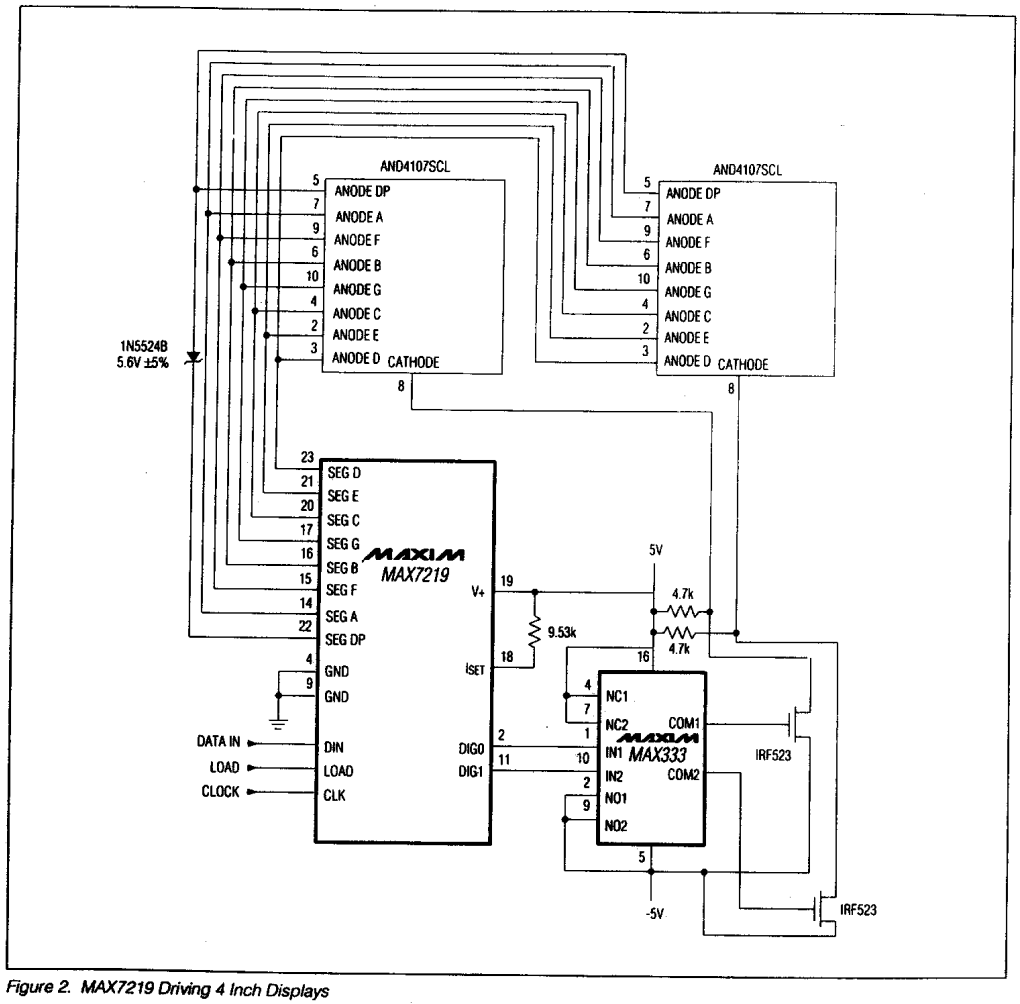


Figure 2. MAX7219 Driving 4 Inch Displays

Serially Interfaced, 8-Digit LED Display Driver

with both scan-limit registers set for 6 digits so that both displays have a 1/6 duty cycle per digit. If 11 digits are needed, set both scan-limit registers for 6 digits and leave one digit driver unconnected. If one display is set for 6 digits and the other for 5 digits, the second display will appear brighter because its duty cycle per digit will be 1/5 while the first display's will be 1/6. Refer to the "No Op Register" section for additional information.

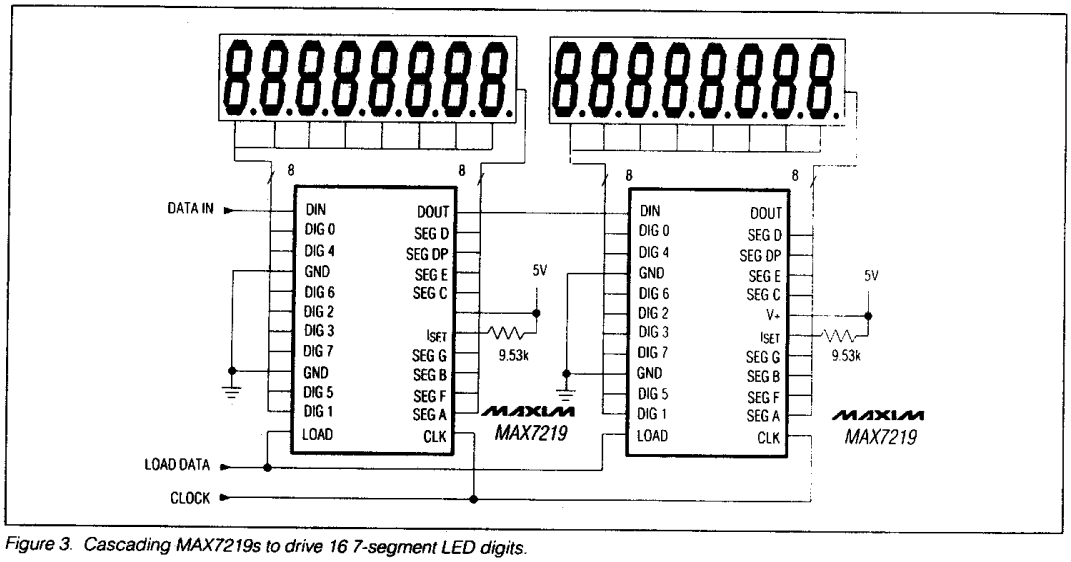


Figure 3. Cascading MAX7219s to drive 16 7-segment LED digits.

Chip Topography