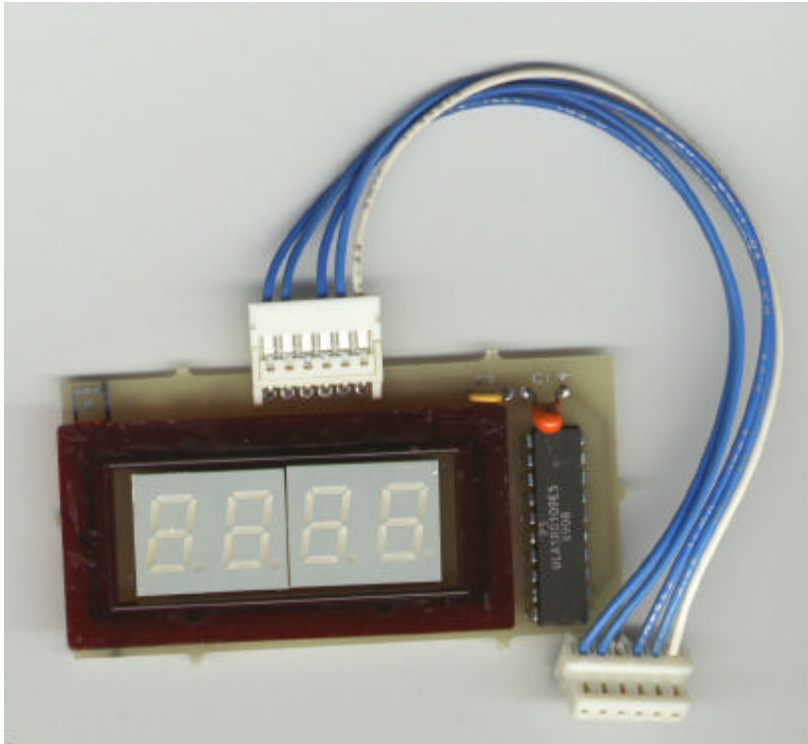




HC4LED Display Module (USED)

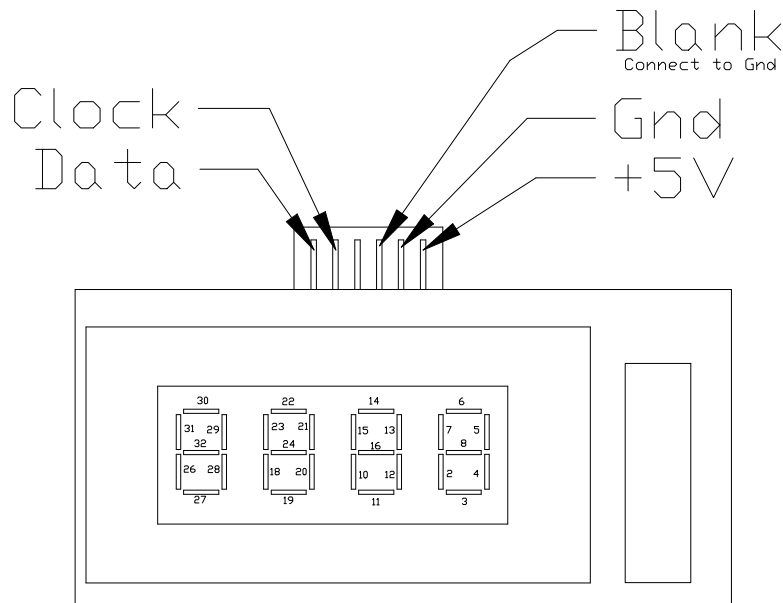
4-Digit 7-Segment Display

- * SPI (SHIFTIN) driver IC
- * 60mA with all segments on
- * Segment Addressable, create some letters & symbols
- * 6" Cable Included; 6 Pin, standard 0.1" spacing
- * 0.39" High red digits
- * Demo programs for BS1 / Prop1; BS2; SX; and PICs
- * Used / Pulled from working systems (may have light surface scratches)
- * NO DECIMAL POINTS (sorry)

Hitt Consulting

105 S Locust Street
Shiremanstown Pa 17011
info@hittconsulting.com
www.hittconsulting.com

HC4LED: 4-Digit 7-Segment Display



OVERVIEW

The HC4LED Display Module contains 4 digits of 7-segment displays. Each digit is segment addressable. In other words any segments can be turned on or off. This allows great flexibility to create some letters and symbols, as well as moving segment displays.

The module contains a display driver that controls each segment so the microcontroller does not have to continually refresh the display. The interface is simple SPI serial (2-wire), consisting of a clock and data line. There is also a "Blank" signal input that **MUST** be connect to ground for the display to show any segments. The display can be blanked by connecting this line to +5 or allowing it to "float".

The module requires about 60mA when all segments are on.

SPI SERIAL INTERFACE

The module uses a simple SPI interface with clock and data lines. The data state is shifted in when the clock signal goes from low to high (rising edge). Segment values (0 or 1) are shifted in starting on the left most digit and travel through each segment until they reaches the rightmost digit. Therefore when sending new segment values for the whole display they should be sent starting with the rightmost digit. Note that 8 bits are required even though there are only 7 segments (the first bit is ignored). The decimal points are NOT connected and cannot be made to function.

IMPORTANT: The display will not be updated with the new segment states until the clock line is held high for at least 1millisecond.

Example Programs

Parallax(R) Basic Stamp 2

```
' HC4LED.BS2
' This program demonstrates the use of the HC4LED display module
' HC4LED Pinout:
'   Pin1 = +5VDC (White wire)
'   Pin2 = Gnd
'   Pin3 = Blank (Must connect to Gnd to enable display)
'   Pin4 = No Connection
'   Pin5 = Clock (Connect to pin 1 for this demo program)
'   Pin6 = Data (Connect to pin 0 for this demo program)
'
'
' To display values:
'   Set the "zeros" variable to 0=No leading zeros; or 1=Show leading zeros
'   Set the variable "value" to 0 to 9999
'   Then use "GOSUB DisplayValue" to show the value on the display
'
' Each segment of the display is addressable, so you can create letters
' and symbols.
' To display custom symbols:
'   Set the variables "segments(0)" thru "segments(3)" (segments(0) is on the left)
'   simply add the segment values that you want on
'   Then use "GOSUB DisplaySegments" to show the segments on the display
'
'   ---4---
'   |       |
'   2         8
'   |       |
'   |---1---|
'   |       |
' 64         16
'   |       |
'   --32---
'
' For example "F" would be 4+2+1+64 = 71
'
' {$STAMP BS2}
' {$PBASIC 2.5}

Clock PIN 1      ' HC4LED module's Clock pin
Dat  PIN 0      ' HC4LED module's Data pin

value  VAR Word      ' Required; Holds value to display
zeros  VAR Bit       ' Required; Determines if leading zeros are displayed
cnt    VAR Byte      ' Required; Used by display routines
segments VAR Byte (4) ' Required; Used by display routines & customer chars
'
DO      ' Repeat forever
  FOR value = 0 TO 1000 ' Count from 0 to 1000
    GOSUB DisplayValue  ' Display count on HC4LED module
    PAUSE 100           ' Wait 0.1 seconds
  NEXT                 ' Next count
  PAUSE 1000           ' Wait 1 second
  segments(0) = 28 ' 7  ' Setup to display "72°F"
  segments(1) = 109 ' 2
  segments(2) = 15 ' °
  segments(3) = 71 ' F
  GOSUB DisplaySegments ' Display custom characters
  PAUSE 2000           ' Wait 2 seconds
LOOP
STOP

DisplayValue:
' Call with:
'   value = (value to display)
'   zeros = (0=No leading zeros; 1=Show leading zeros)
FOR cnt = 0 TO 4
  LOOKUP value DIG cnt, [126,24,109,61,27,55,115,28,127,31], segments(3-cnt)
NEXT
```

```

IF zeros = 0 THEN
  IF segments(0) = 126 THEN
    segments(0) = 0
  IF segments(1) = 126 THEN
    segments(1) = 0
  IF segments(2) = 126 THEN
    segments(2) = 0
  ENDIF
ENDIF
ENDIF
ENDIF
ENDIF

DisplaySegments:
' Call with:
' segments(0) thru segments(3) set to custom character values
' segments(0) is on the left; segments(3) is on the right
SHIFTOUT Dat, Clock, MSBFIRST,[segments(3), segments(2), segments(1), (segments(0) >> 1)\7]
Dat = segments(0) & 1
HIGH Clock
' Clock MUST remain high for at LEAST 1 millisecond for the
' new data to be latched onto the display.
PAUSE 1
RETURN

```

End of program listing

Parallax(R) Basic Stamp 1 / Prop 1

```

' HC4LED.BS1
' This program demonstrates the use of the HC4LED display module
' HC4LED Pinout:
'   Pin1 = +5VDC (White wire)
'   Pin2 = Gnd
'   Pin3 = Blank (Must connect to Gnd to enable display)
'   Pin4 = No Connection
'   Pin5 = Clock
'   Pin6 = Data
'
'
' To display values:
'   Set the "zeros" variable to 0=No leading zeros; or 1=Show leading zeros
'   Set the variable "value" to 0 to 9999
'   Then use "GOSUB DisplayValue" to show the value on the display
'
' Each segment of the display is addressable, so you can create letters
' and symbols.
' To display custom symbols:
'   Set the variables "segments(0)" thru "segments(3)" (segments(0) is on the right)
'   simply add the segment values that you want on
'   Then use "GOSUB DisplaySegments" to show the segments on the display
'
'   ---4---
'   |       |
'   2       8
'   |       |
'   |---1---|
'   |       |
'  64       16
'   |       |
'   --32---
'
' For example "F" would be 4+2+1+64 = 71
'
' {$STAMP BS1}
' {$PBASIC 1.0}

SYMBOL Clk      = 0      'Display Clock is pin 0
SYMBOL Dat      = PIN1   'Display Data is pin 1
SYMBOL Cnt      = B2     'Counter variable for DisplayDigit subroutine
SYMBOL digit    = B3     'Current value digit
SYMBOL value    = W2     'Value to be displayed

```

```

' Next available variables are: BIT8,B6,W3

Start:
  DIRS = %00000011      'Set Clock and Data pins as outputs rest as inputs
  HIGH Clk

Main:
  ' Display HELP
  LOW Clk
  B0 = 79
  GOSUB DisplaySegments
  PULSOUT Clk,1          ' Pulsout last bit except for last digit
  B0 = 98
  GOSUB DisplaySegments
  PULSOUT Clk,1
  B0 = 103
  GOSUB DisplaySegments
  PULSOUT Clk,1
  B0 = 91
  GOSUB DisplaySegments
  HIGH Clk                ' Keep Clk high to latch in new display data
  ' Show HELP for 5 seconds
  PAUSE 5000

  ' Display values from 1000 to 0
  FOR value = 1000 TO 0 STEP -1
    GOSUB DisplayValue
  NEXT
  GOTO Main

DisplayValue:
  ' Call with "value" set to the value to be displayed
  LOW Clk
  digit = value // 10
  GOSUB DisplayDigit
  PULSOUT Clk,1
  digit= value / 10 // 10
  GOSUB DisplayDigit
  PULSOUT Clk,1
  digit= value / 100 // 10
  GOSUB DisplayDigit
  PULSOUT Clk,1
  digit= value / 1000 // 10
  GOSUB DisplayDigit
  HIGH Clk
  RETURN

DisplayDigit:
  ' Called by DisplayValue subroutine
  LOOKUP digit, (126,24,109,61,27,55,115,28,127,31),B0

DisplaySegments:
  ' Call with B0 containing the segments for the next digit
  FOR cnt = 1 TO 7
    Dat = BIT7
    PULSOUT Clk,1
    B0 = B0 * 2
  NEXT
  Dat = BIT7
  RETURN

```

END OF LISTING

Parallax(R) SX SX/B

```
=====
'
' File..... HC4LED.SXB
' Compiler.. SXB Version 1.42.01
' Purpose... Demonstrate use of HC4LED display module
' Author.... Terry Hitt
' E-mail.... terry@hittconsulting.com
' Started...
' Updated...
'
=====

' -----
' Program Description
' -----

' HC4LED Pinout:
' Pin1 = +5VDC (White wire)
' Pin2 = Gnd
' Pin3 = Blank (Must connect to Gnd to enable display)
' Pin4 = No Connection
' Pin5 = Clock (Connect to RA.0 for this demo program)
' Pin6 = Data (Connect to RA.1 for this demo program)
'
'
' Each segment of the display is addressable, so you can create letters
' and symbols.
' To display custom symbols:
' Set the variables "segments(0)" thru "segments(3)" (segments(0) is on the left)
' simply add the segment values that you want on
' Then use "DisplaySegments" to show the segments on the display
'
'   ---4---
'   |       |
'   2       8
'   |       |
'   |---1---|
'   |       |
' 64       16
'   |       |
'   --32---
'
' For example "F" would be 4+2+1+64 = 71
'
' -----
' Device Settings
' -----

DEVICE SX28,OSC4MHZ,TURBO,STACKX,OPTIONX
FREQ 4_000_000

' -----
' IO Pins
' -----

Clk      VAR RA.0
Dat      VAR RA.1

' -----
' Variables
' -----

value    VAR Byte (4)
segments VAR Byte (4)
cnt      VAR Byte
temp1    VAR Byte

' =====
' PROGRAM Start
' =====

' -----
' Subroutine Declarations
' -----

DisplayValue SUB 4
UpdateValue  SUB
DisplaySegments SUB 4
LEDOut      SUB 1
```

```

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

Start:
    HIGH Clk

Main:
    DisplayValue 1,2,3,4
    PAUSE 1000
    DisplaySegments 91,103,98,79 ' HELP
    PAUSE 1000
    DisplayValue 0,0,0,0
    DO
        INC value(3)
        IF value(3) > 9 THEN
            value(3) = 0
            INC value(2)
        ENDIF
        IF value(2) > 9 THEN
            value(2) = 0
            INC value(1)
        ENDIF
        IF value(1) > 9 THEN
            value(1) = 0
            INC value(0)
        ENDIF
        IF value(0) > 9 THEN
            value(0) = 0
        ENDIF
        UpdateValue
        PAUSE 1
    LOOP

    GOTO Main

' -----
' Subroutine Code
' -----

DisplayValue:
    PUT value,__PARAM1,__PARAM2,__PARAM3,__PARAM4

UpdateValue:
    FOR cnt = 0 to 3
        READ DigitSegs+value(cnt),segments(cnt)
    NEXT
    GOTO DisplayArray

DisplaySegments:
    PUT segments,__PARAM1,__PARAM2,__PARAM3,__PARAM4

DisplayArray:
    HIGH Clk
    LEDOut segments(3)
    LEDOut segments(2)
    LEDOut segments(1)
    LEDOut segments(0)
    RETURN

LEDOut:
    temp1 = __PARAM1
    SHIFTOUT Dat, Clk, MSBFIRST, temp1
    RETURN

' -----
' Data
' -----

DigitSegs:
    DATA 126,24,109,61,27,55,115,28,127,31

```

END OF LISTING

Microchip(R) PIC Assembly

```

; *** COMPILED WITH BAS2PIC VERSION 0.30 ***
; ' Test Program for BAS2PIC compiler 0.30
; ' Use with Microchip PICKit1 and a PIC12f675
; ' Shows how to control the HC4LED display module

list p=12f675                ;DEVICE 12f675
#include <p12f675.inc>
errorlevel -302              ;Ignore BANK warnings
__INTW EQU 0x20
__INTSTATUS EQU 0x21
__TEMP EQU 0x22

; ' FOLLOWING LINE IS FOR DEVICE CONFIGURATION SEE DEVICE DOCUMENTATION
__CONFIG __CP_OFF & __WDT_OFF & __BODEN_ON & __PWRTE_ON & __INTRC_OSC_NOCLKOUT & __MCLRE_OFF &
__CPD_OFF

ORG 0x00                    ;PROGRAM START
NOP                          ;REQUIRED FOR DEBUGGER
GOTO START

ORG 0x04                    ;INTERRUPT VECTOR
GOTO __INTERRUPT

; ' Define Pins
#define Clk GPIO,5           ;DIM Clk AT GPIO.5
#define Dat GPIO,4           ;DIM Dat AT GPIO.4

; ' Define Variables
cnt EQU D'35'                ;DIM cnt AS BYTE
segments EQU D'36'           ;DIM segments AS BYTE
#define segmentsMSB segments,7 ;DIM segmentsMSB AT segments.7

START:

BCF STATUS,RP0              ; BANK 0
CLRF GPIO                   ; GPIO=%00000000 'All output off
MOVLW B'00000111'           ; CMCON=%00000111 'No comparitors
MOVWF CMCON
BSF STATUS,RP0              ; BANK 1 ' SET REGS IN BANK 1
MOVLW B'11111111'           ; TRISIO=%11111111 'All pins inputs
MOVWF TRISIO
CLRF VRCON                  ; VRCON=0 ' VRef Off
MOVLW B'00010001'           ; ANSEL=%00010001 ' Analog Select
MOVWF ANSEL
BCF STATUS,RP0              ; BANK 0
MOVLW B'00000001'           ; ADCON0=%00000001 ' Analog Configure
MOVWF ADCON0

MOVLW B'11001111'           ; TRISIO = %11001111 ' Make Clk & Dat outputs
BSF STATUS,RP0
MOVWF TRISIO
BCF STATUS,RP0
BSF Clk                     ; HIGH Clk

AGAIN:
MOVLW D'71'                 ; segments = 71
MOVWF segments
CALL SendSegments           ; GOSUB SendSegments
MOVLW D'15'                 ; segments = 15
MOVWF segments
CALL SendSegments           ; GOSUB SendSegments
MOVLW D'109'                ; segments = 109
MOVWF segments
CALL SendSegments           ; GOSUB SendSegments
MOVLW D'28'                 ; segments = 28
MOVWF segments
CALL SendSegments           ; GOSUB SendSegments
CLRF __TEMP ; 255+1=0        ; DELAY 255,255
MOVLW D'0' ; 255+1=0
ADDLW 0xFF
BTFSS STATUS,Z
GOTO $-2
DECFSZ __TEMP,F
GOTO $-4

```

```

; ' Clear display
CALL SendSegments          ; GOSUB SendSegments
CALL SendSegments          ; GOSUB SendSegments
CALL SendSegments          ; GOSUB SendSegments
CALL SendSegments          ; GOSUB SendSegments
CLRF __TEMP ; 255+1=0      ; DELAY 255,255
MOVLW D'0' ; 255+1=0
ADDLW 0xFF
BTFSS STATUS,Z
GOTO $-2
DECFSZ __TEMP,F
GOTO $-4

GOTO AGAIN                ; GOTO AGAIN

SendSegments:
    MOVLW D'1'            ; FOR cnt=1 TO 8
    MOVWF cnt
__FOR_cnt_1:
    BCF Dat                ; LOW Dat
    BTFSS segmentsMSB      ; IF segmentsMSB = 0 THEN DatSet
    GOTO DatSet
    BSF Dat                ; HIGH Dat
DatSet:
    BCF Clk                ; LOW Clk
    MOVLW D'10'+D'1'       ; DELAY 10
    ADDLW 0xFF
    BTFSS STATUS,Z
    GOTO $-2
    BSF Clk                ; HIGH Clk
    MOVLW D'10'+D'1'       ; DELAY 10
    ADDLW 0xFF
    BTFSS STATUS,Z
    GOTO $-2
    BCF STATUS,C           ; segments = segments << 1
    RLF segments,F
    MOVLW D'8'             ; NEXT cnt
    SUBWF cnt,W
    BTFSC STATUS,Z
    GOTO $+3
    INCF cnt,F
    GOTO __FOR_cnt_1
    RETURN                ; RETURN

SLEEP                      ;END
GOTO $

__INTERRUPT:
RETURN                    ;DEFAULT INTERRUPT CODE
END                      ;RETURN WITHOUT ENABLING FURTHER INTERRUPTS

```

END OF LISTING