Micro-Controllers

Module 4: Digital Display

PREPARED BY

IAT Curriculum Unit

August 2008
Module 4: Digital Display

Module Objectives

1. Realize digital display concept
2. Realize 7-Segment display LED
3. Learn how to program 7-Segment display

Module Contents

1. Digital Display
2. Programming 7-Segment Display
Digital Display

In this module you will learn about digital display of information, and how to program the Micro-Controller to display numbers and letters on the digital display.

When you need to know the time you simply look at your watch. Some times you need to follow the minutes and hours pointers to figure the time like the girly watch shown in figure 4.1 to the left, and other times the watch displays the time in digits simple to read like the Micky mouse watch shown to the right. Both watches will display time information, but we call the type of display in the girly pink watch as analog display. The time display in the boy’s watch is referred to as digital display. Which display is easier to read? Which one is more accurate? Where else can you find digital display?

In digital display, each digit is called 7-Segment display. A 7-Segment display is rectangular block of 7 lines of equal length that can be lit selectively to display digits and some letters. Every line corresponds to one LED that can be controlled individually. Figure 4.2 shows a part drawing of the 7-Segment LED display you will use in this module’s activities. It has one additional LED, a dot that can be used as a decimal point. The letters shown corresponds to lines. The numbers refer to the pins numbers. Table 4.1 lists the pin mapping. Pins’ maps help you connect the 7-Segment display in the electric circuit. It informs the pin number, pin name and pin reference.
<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin 1</td>
<td>Controls Segment E</td>
</tr>
<tr>
<td>Pin 2</td>
<td>Controls Segment D</td>
</tr>
<tr>
<td>Pin 3, 8</td>
<td>Common Cathode</td>
</tr>
<tr>
<td>Pin 4</td>
<td>Controls Segment C</td>
</tr>
<tr>
<td>Pin 5</td>
<td>Controls decimal point</td>
</tr>
<tr>
<td>Pin 6</td>
<td>Controls Segment B</td>
</tr>
<tr>
<td>Pin 7</td>
<td>Controls Segment A</td>
</tr>
<tr>
<td>Pin 9</td>
<td>Controls Segment F</td>
</tr>
<tr>
<td>Pin 10</td>
<td>Controls Segment G</td>
</tr>
</tbody>
</table>

Table 4.1: 7 Segment Display pin mapping

Figure 4.3 illustrates the 7-segment display schematic diagram. Each LED anode is connected to a separate pin (pins 1-2, 4, 6-7, 9-10). The cathodes are internally connected and can be connected to the circuit ground Vss by using either pin 3 or 8.

**Connect and Test the 7-Segment display Experiment**

In this experiment you will connect the 7-Segment display and manually test each segment.
You will need 1x 7-Segment display, 5x 1kΩ resistors, 5x jumper wires.

- Set the 3-position switch on the BOE to position-0 to turn off the Power.

- Build the circuit shown in figures 4.5 and 4.6. “nc” means wire is not connected.

- Reconnect power and see what happens? Segment A is supposed to emit light.

If Segment A did not emit light, revise your connections and note what was wrong.
- Disconnect power, modify the circuit by connecting the resistor to the B Segment LED. Which pin number is that? ______. If you need connection help check figures 4.7 and 4.8.

- Reconnect power; verify that Segment B emits light.

If Segment B did not emit light, revise your connections and note what was wrong.
- Disconnect the power and repeat the same steps for the remaining segments. Fill in table 4.2

<table>
<thead>
<tr>
<th>Segment</th>
<th>Pin No.</th>
<th>Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2: Segments’ test results

- Disconnect the power, and build the circuit shown in figures 4.9 and 4.10 to display number 3.

- Reconnect power, verify that it is number 3

If number 3 was not on, revise your connections and note what was wrong.

---

Fig 4.9: 7-Segment display test circuit (Number 3)

Fig 4.10: 7-Segment display test circuit schematic diagram (Number 3)
- Can you display Number 2?

- Draw the schematic diagram in the box below

- Connect the circuit and test your connection.

- Once you are done, disconnect the power.
Programming 7-Segment Display

If you include the decimal point there you need to connect eight different Basic Stamp I/O pins that send high/low signals to the 7-Segment LED display. That’s eight different **HIGH** or **LOW** commands just to display one number. If you want to count from zero to nine, that would be a huge amount of programming. It is simple but lengthy.

There are other shorter ways to program the 7-Segment display and also as simple such as: **DIRH** and **OUTH**. **DIRH** and **OUTH** are not commands, they are system defined byte variables that you can read and modify, and you don’t need to declare them.

**DIRH**

**DIRH** is a variable that controls the direction (input or output) of the pins P8-P15.

Example:

**DIRH= % 1110000**

**DIR** is short for Direction,

- **H** is short for high P8-P15,
  - it could be **L** short for low P0-P7

**%** means following number is a binary number

**11100000**: each bit corresponds to a pin in the following order:

<table>
<thead>
<tr>
<th>Pin number:</th>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

1 means Output
0 means input
OUTH

OUTH is a variable that controls the status of each output pin (high or low) for the pins P8-P15.

Example:
**OUTH= % 11110000**
  
  H is short for high (P8-P15), it could be L short for Low (P0-P7)
  
  11110000: each bit corresponds to a pin in the following order:
  
  %                  1   1   1   1   0   0   0   0
  Pin number:  15 14  13 12 11 10  9   8
  1 means high
  0 means low

**Basic Stamp Controlled 7-Segment Display via OUTH command Experiment**

In this experiment, you will connect the 7-Segment LED display to the BASIC Stamp, and then run a simple program to test and make sure each LED is properly connected.

You will need 1x 7-Segment display, 8x 1kΩ resistors and jumper wires

- Disconnect the power

- Build the circuit shown in figures 4.11 and 4.12

- Write the following program, download and test:
- Verify that every segment in the 7-Segment display lights briefly on then off. _____________________

- Fill in the Micro-Controller pin to segment connection table

<table>
<thead>
<tr>
<th>Micro-Controller Pin</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>P8</td>
<td></td>
</tr>
<tr>
<td>P9</td>
<td></td>
</tr>
<tr>
<td>P10</td>
<td></td>
</tr>
<tr>
<td>P11</td>
<td></td>
</tr>
<tr>
<td>P12</td>
<td></td>
</tr>
<tr>
<td>P13</td>
<td></td>
</tr>
<tr>
<td>P14</td>
<td></td>
</tr>
<tr>
<td>P15</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.3: Micro-Controller’s pins connection segments

- Remove the command "LOW pincounter", and rerun the program. What is the difference? _____________________

________________________

________________________
- Write the following program, download, and analyze what every command will do.

Program

```
DEBUG "Program Running!"
OUTH = %00000000
DIRH = %11111111
  ' BAFG.CDE
OUTH = %11100111
PAUSE 1000
OUTH = %10000100
PAUSE 1000
OUGH = %11010011
PAUSE 1000
OUTH = %11010110
PAUSE 1000
OUTH = %10110100
PAUSE 1000
OUTH = %01110110
PAUSE 1000
OUTH = %01110111
PAUSE 1000
OUTH = %11000100
PAUSE 1000
OUGH = %11110111
PAUSE 1000
OUGH = %11110110
PAUSE 1000
DIRH = %00000000
END
```

Analysis

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

- Modify the program to display your name and test it

- Modify the program to display the letters A, b, C, d, E, and F and fill in table 4.4.

<table>
<thead>
<tr>
<th>Letter</th>
<th>OUTH value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
</tr>
<tr>
<td>d</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.4: A-F letter value.
LOOKUP and LOOKDOWN Commands

LOOKUP and LOOKDOWN commands also can be used to Program the 7-Segment display. In LOOK commands you can select an item from a list, or compare an item to list of items.

LOOKUP
Using this command you can retrieve an item from a pre-defined list based on that item’s position (index) in the list.

Command Syntax:
```
LOOKUP
```

Index: it is a variable of maximum type byte to indicate the position of the required item in the list.
[item1, item2 ...] is the list of items that could be of any variable type. The maximum number of items in a list =256 (0-255).
Value: it is a variable to store the required item from the list.

Example:

```
Index VAR Byte
value VAR Word

DO
    DEBUG "Type index value 0 .5", CR
    DEBUG IN DEC index
    LOOKUP Index, [35, 89, 6000, 0, 15000, 777], value
    DEBUG "Value is ", DEC5 value, CR
    PAUSE 2000
    DEBUG CLS

LOOP
```

Fig 4.13: LOOKUP command example
Once you type in the index value, the related item list will be stored in Value as follows:
Index = 0, value = 35
Index = 1, value = 89
Index = 2, value = 0
...

Remarks:
- Pay attention that index starting number is **0**.
- If you give an index number exceeds the item numbers in a list, the last stored value will be displayed.

**LOOKDOWN**
LOOKDOWN command compares a value with items of a list and informs the position (index) of the first item that fits the comparison criteria.

Command Syntax:
`LOOKDOWN` value, Comparison [item1, item2 ...], Index

Value: is the variable to be compared to items in a list.

Comparison: is the comparison operation required, Let is be equal to (=) values, greater than (>), or less than (<). When comparison left empty, it will be specified equal to (=) by default.

[item1, item2 ...] is the list of items.

Index: it is the variable that will store the item’s position in the list. Every time the result will be one index value only, which is the index of the first item that matches the criteria, even if there is more than one match.
Example:

```
' ($STAMP BS2)
' ($PBASIC 2.5)

Index VAR Byte
value VAR Word
DEBUG "list is: 5,10,777,65000,1010,0", CR

DO

DEBUG "Type value", CR
DEBUGIN DEC value
LOOKDOWN value,[5,10,777,65000,1010,0],index
DEBUG "Index is: ", DEC index, CR

LOOP
```

Fig 4.14: LOOKDOWN command example

Once you type in the value, LOOKDOWN command will store the position of this value in the variable index. For Example:
value =5, Index= 0
value = 10, Index =1
value = 777, Index= 2
...
Since no comparison operator was mentioned it will be equal to (=) by default.

**LOOK commands Test Experiment**
In this part of the experiment you will simply test the functionality of LOOKUP and LOOKDOWN commands. All what you need is BOE only.

**LOOKUP command experiment:**

- Power on BOE
- Write and run the following program as is.
Fig 4.15: LOOKUP command experiment program

- **When Index is 2**, the value displayed on **Debug Terminal Window** is__________.

- Change the index number to any value between 0-5 and fill in the table.

- Do results match your expectations? ______

- Justify the result you got when index =5:

  ______________________________
  ______________________________

<table>
<thead>
<tr>
<th>Index</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.5: LOOKUP index- value table
LOOKDOWN command experiment:

- Write and run the following program as is.

```pascal
' ( $STAMP BS2 )
' ( $PBASIC 2.5 )

value VAR Byte
index VAR Nib

value = 167

DEBUG ? value

LOOKDOWN value, [7, 85, 19, 167, 28], index

DEBUG ? index, CR
DEBUG "Change the value variable to a ", CR,
   "different number in this list:\n\n\n    7, 85, 19, 167, or 28.\n\nRun the modified program and ", CR,
   "check to see what number the ", CR,
   "LOOKDOWN command places in the ", CR,
   "index variable."

END
```

Fig 4.16: LOOKDOWN command experiment program

- When value = 167, Index = ____.
- Change the value to 7, 85, 19, 28 and 168. Run the program after every change, and fill in table 4.6.
- Justify the result you got when Value was 168:

<table>
<thead>
<tr>
<th>Value</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>85</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td></td>
</tr>
<tr>
<td>168</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.6: LOOKDOWN value-index table
- Modify the LOOKDOWN command as follows:
  LOOKDOWN value, <= [7, 19, 28, 85, 167], index, where value = 35

  The index is _____.

- Justify the result:
  ____________________________________________
  ____________________________________________
  ____________________________________________
  ____________________________________________
  ____________________________________________

**Programming 7-Segment Display with LOOKUP command experiment**

In this experiment you will program 7-Segment display by using LOOKUP command instead of OUTH command, and realize the difference. You will need the same circuit you connected in the Basic Stamp Controlled 7-Segment Display via OUTH command Experiment in page 9.

- Write and run the following program:
- Verify that it displays digits 0 to 9 (with much less work).

- Take a look at the Debug Terminal while the program runs. It shows how the value of index is used by the LOOKUP command to load the correct binary value from the list into OUTH.

- Modify the program to display A, b, C, d, E and F letters.

- Once You are done, do the following:
  1- Save your program.
  2- Switch off the BOE power
  3- Disassemble your circuit, and return the components to the safe.
  4- Clean your working area and report any damaged components.