The Basys 3 board contains one four-digit common anode seven-segment LED display. Each of the four digits is composed of seven segments arranged in a "figure 8" pattern, with an LED embedded in each segment. Segment LEDs can be individually illuminated, so any one of 128 patterns can be displayed on a digit by illuminating certain LED segments and leaving the others dark, as shown in Fig. 1. Of these 128 possible patterns, the ten corresponding to the decimal digits are the most useful.

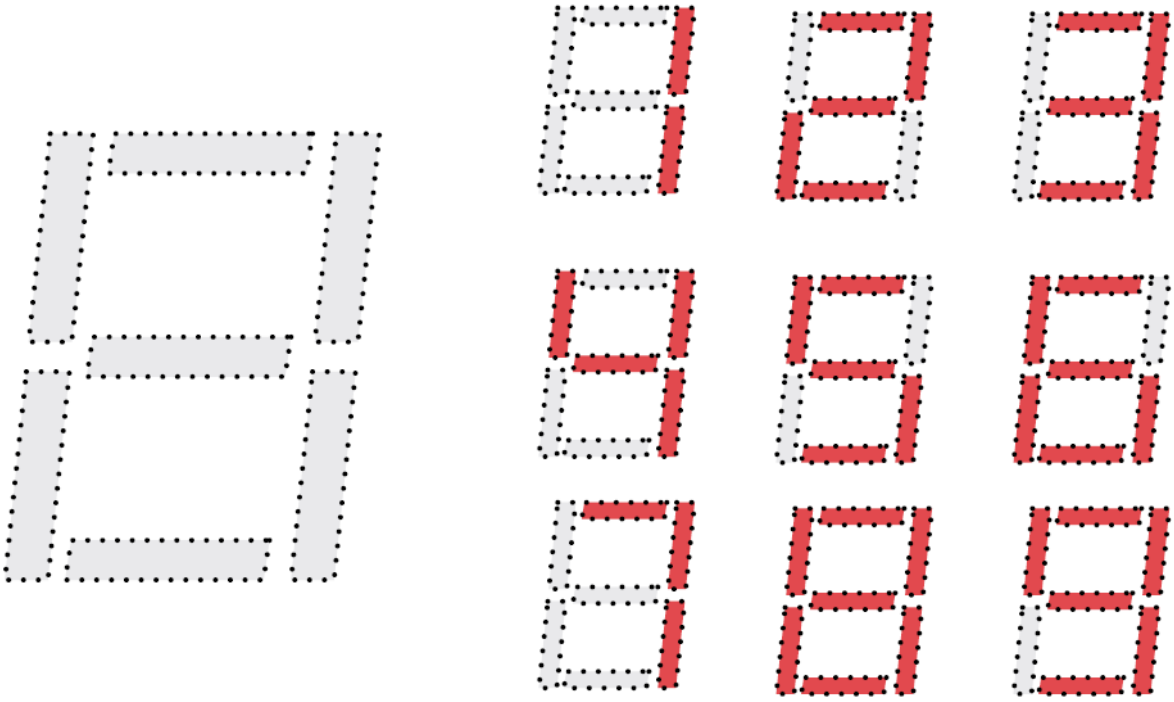


Figure 1. An un-illuminated seven-segment display and nine illumination patterns corresponding to decimal digits.

The anodes of the seven LEDs forming each digit are tied together into one "common anode" circuit node, but the LED cathodes remain separate, as shown in Fig. 2. The common anode signals are available as four "digit enable" input signals to the 4-digit display. The cathodes of similar segments on all four displays are connected into seven circuit nodes labeled CA through CG (for example, the four "D" cathodes from the four digits are grouped together into a single circuit node called "CD"). These seven cathode signals are available as inputs to the 4-digit display. This signal connection scheme creates a multiplexed display, where the cathode signals are common to all digits but they can only illuminate the segments of the digit whose corresponding anode signal is asserted.

To illuminate a segment, the anode should be driven high while the cathode is driven low. However, since the Basys 3 uses transistors to drive enough current into the common anode point, the anode enables are inverted. Therefore, both the AN0..3 and the CA..G/DP signals are driven low when active.

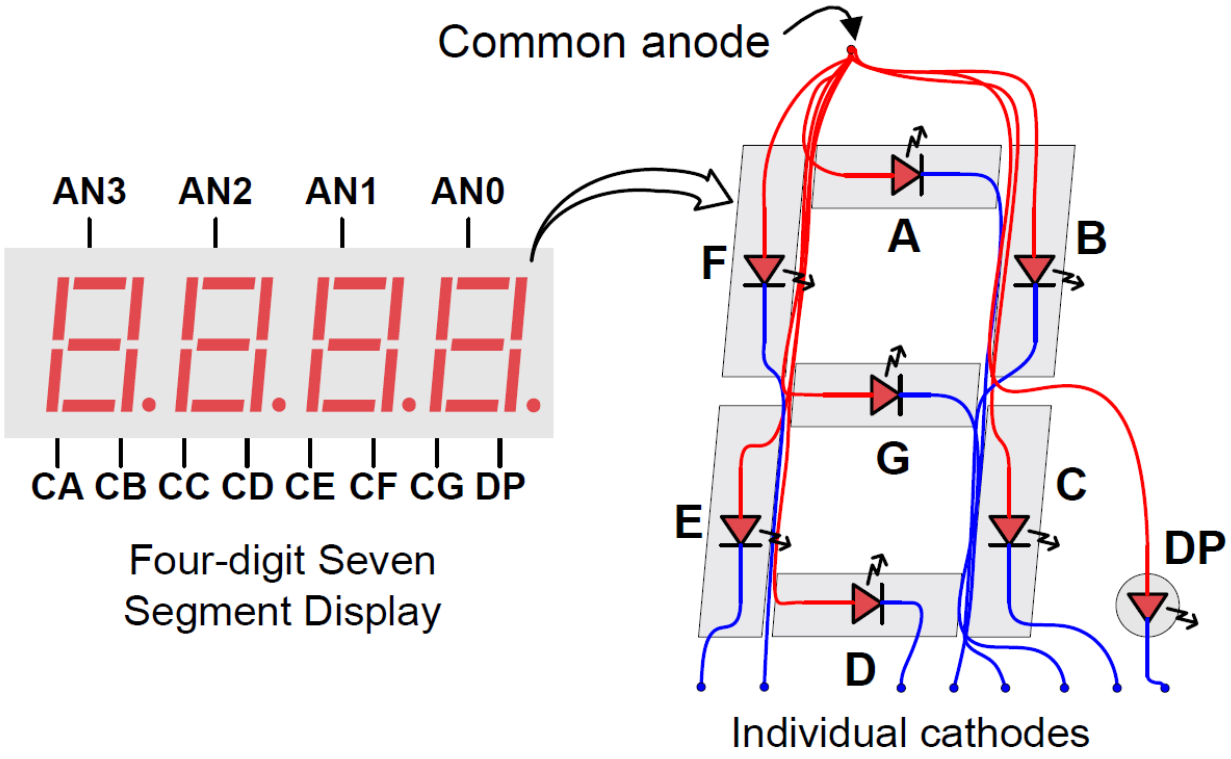


Figure 2. Common anode circuit node.

A scanning display controller circuit can be used to show a four-digit number on this display. This circuit drives the anode signals and corresponding cathode patterns of each digit in a repeating, continuous succession at an update rate that is faster than the human eye can detect. Each digit is illuminated just one-fourth of the time, but because the eye cannot perceive the darkening of a digit before it is illuminated again, the digit appears continuously illuminated. If the update, or "refresh", rate is slowed to around 45Hz, a flicker can be noticed in the display.

For each of the four digits to appear bright and continuously illuminated, all four digits should be driven once every 1 to 16ms, for a refresh frequency of about 1 KHz to 60Hz. For example, in a 62.5Hz refresh scheme, the entire display would be refreshed once every 16ms, and each digit would be illuminated for 1/4 of the refresh cycle, or 4ms. The controller must drive the cathodes low with the correct pattern when the corresponding anode signal is driven high. To illustrate the process, if AN0 is asserted while CB and CC are asserted, then a "1" will be displayed in digit position 1. Then, if AN1 is asserted while CA, CB, and CC are asserted, a "7" will be displayed in digit position 2. If AN0, CB, and CC are driven for 4ms, and then AN1, CA, CB, and CC are driven for 4ms in an endless succession, the display will show "71" in the first two digits. An example timing diagram for a four-digit controller is shown in Fig. 3.

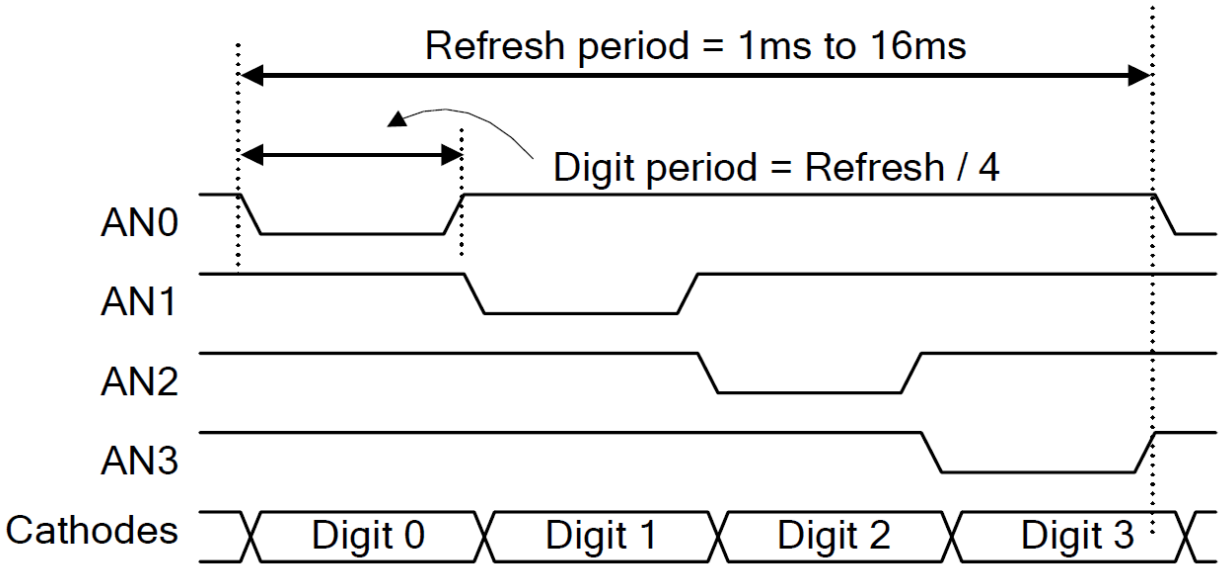
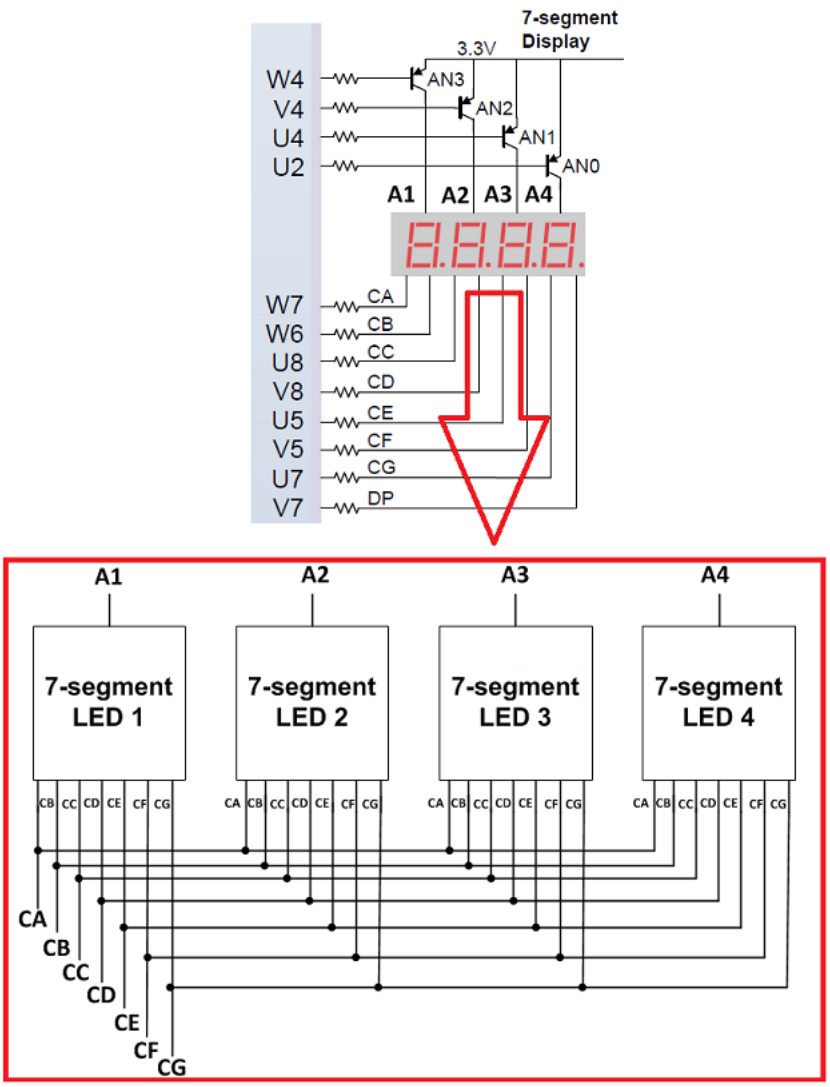


Figure 3. Four digit scanning display controller timing diagram.


7-segment
Display
3.3V
W4
V4
U4
U2
AN3
AN2
AN1
AN0
A1
A2
A3
A4
W7
W6
U8
V8
U5
V5
U7
V7
CA
CB
CC
CD
CE
CF
CG
DP
A1
A2
A3
A4
7-segment
LED 1
7-segment
LED 2
7-segment
LED 3
7-segment
LED 4
CA
CB
CC
CD
CE
CF
CG

```vhdl
-- Example VHDL code for BCD to seven-segment display on Basys 3 FPGA
process(LED_BCD)
begin
    case LED_BCD is
    when "0000" => LED_out <= "0000001"; -- "0"
    when "0001" => LED_out <= "1001111"; -- "1"
    when "0010" => LED_out <= "0010010"; -- "2"
    when "0011" => LED_out <= "0000110"; -- "3"
    when "0100" => LED_out <= "1001100"; -- "4"
    when "0101" => LED_out <= "0100100"; -- "5"
    when "0110" => LED_out <= "0100000"; -- "6"
    when "0111" => LED_out <= "0001111"; -- "7"
    when "1000" => LED_out <= "0000000"; -- "8"
    when "1001" => LED_out <= "0000100"; -- "9"
    when "1010" => LED_out <= "0000010"; -- "a"
    when "1011" => LED_out <= "1100000"; -- "b"
    when "1100" => LED_out <= "0110001"; -- "C"
    when "1101" => LED_out <= "1000010"; -- "d"
    when "1110" => LED_out <= "0110000"; -- "E"
    when "1111" => LED_out <= "0111000"; -- "F"
    end case;
end process;
```

```
-- 7-segment display controller, generate refresh period of 10.5ms
process(clock_100Mhz,reset)
begin
    if(reset='1') then
        refresh_counter <= (others => '0');
    elsif(rising_edge(clock_100Mhz)) then
        refresh_counter <= refresh_counter + 1;
    end if;
end process;
 LED_activating_counter <= refresh_counter(19 downto 18);
-- 4-to-1 MUX to generate anode activating signals for 4 LEDs
process(LED_activating_counter)
begin
    case LED_activating_counter is
    when "00" =>
        Anode_Activate <= "0111";
        -- activate LED1 and Deactivate LED2, LED3, LED4
        LED_BCD <= displayed_number(15 downto 12);
        -- the first hex digit of the 16-bit number
    when "01" =>
        Anode_Activate <= "1011";
        -- activate LED2 and Deactivate LED1, LED3, LED4
        LED_BCD <= displayed_number(11 downto 8);
        -- the second hex digit of the 16-bit number
    when "10" =>
        Anode_Activate <= "1101";
        -- activate LED3 and Deactivate LED2, LED1, LED4
        LED_BCD <= displayed_number(7 downto 4);
        -- the third hex digit of the 16-bit number
    when "11" =>
        Anode_Activate <= "1110";
        -- activate LED4 and Deactivate LED2, LED3, LED1
        LED_BCD <= displayed_number(3 downto 0);
        -- the fourth hex digit of the 16-bit number
    end case;
end process;
```