Seven-Segment-Display-Controller

Datasheet

Dominik Socher

29 3 2021

Contents

List of Figures

General Description

The “Seven-Segment-Display-Controller” IP is a component written in VHDL. Its main purpose is to show digits on a seven-segment-display. Included in the design is a BCD code decoder, an Avalon bus interface and multiplexing circuitry to allocate data to each digit. The output is in pairs of two displays. The least amount of display connected to the device has to be two. The component is capable of several hardware function which are accessible trough software. 16 bit of the 32 bit data interface is used whereof 8 bit MSB are controlling data and 8 bit LSB are display data. The complete system is shown in Figure 1.

Applications

- Seven-Segment-Displays
- Industrial-controllers
- LED-matrix

Features

- Individual Digit control
- Shutdown function
- Self test function
- Software driver

Overview

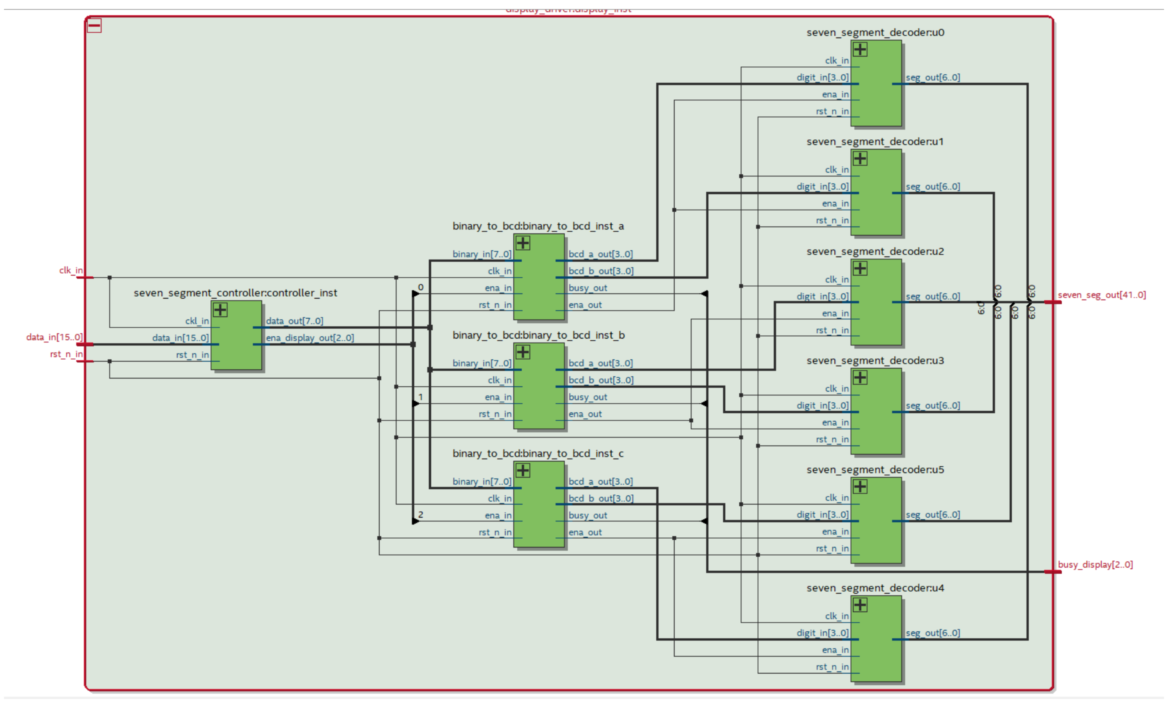

Figure 1: Block Schematic

Detailed Description

The Seven-Segment-Display-Controller uses 16 of the 32-bit of the Avalon Bus at data_in (MSB). MSB 8 bit controls the functions of the device and LSB 8 bit is used from the binary coded decimal decoder. The seven-segment data are combined in a 42-bit wide data bus. A 3 bit bus is used to read from the device. This bus returns if the device is busy. This is used for the self testing function in software. Table 1 shows the data format used of the controller

Table 1 Data format

d15	d14	d13	d12	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0
Control data								digit data							

Digit and Control Register

Table 2 list the 8 digit and control register i is possible to write direct into these registers to control the Seven-Segment-Display-Controller.

Table 2 Register Map

Register	HEX-Code	Description
Shutdown on	0x00	shuts down the device
Shutdown off	0x01	activates device
Self-test on	0xF9	setting controller to test mode
Self-test of	0x08	self test off
Digit 1-2	0x11	activates digit group 1
Digit 3-4	0x21	activates digit group 2
Digit 5-6	0x41	activates digit group 3
All digits	0xF1	activates all digit groups

Initial Power-Up

On initial power-up, all control register are reset, all display is blanked and the Seven-Segment-Display-Controller is in shutdown mode. The device must be activated by writing into the control register.

Shutdown Mode

When the Seven-Segment-Display-Controller is in shutdown mode all digits are blanked out and the device is in standby mode. By writing to the shutdown register the device gets activated.

Display Self Test

The display test register operates in tow modes: on and off. The self-test mode turns all LEDs on by overriding the digit-data-register.

Digits

By writing in the digit register it is possible to choose desired digit group or all of them.

Accesing digits

When using a standard seven-segment display like shown in Figure 2. it is possible to write integers ranging from 0-99 into data register d7-0 LSB.

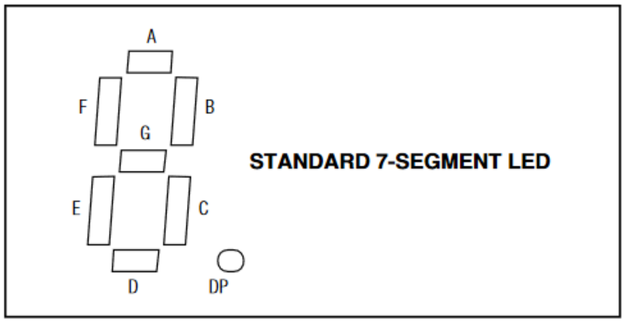

Figure 2: Seven-Segment-Display

Software

The component follows with a software driver. called *seven_seg.h*. Because of the hardware function it is necessary to call the *init* function to activate the component. Furthermore, it test the component and resets it to zero.

Writing digits

For writing digits the user need to call the function below. The first argument is the digit to choose, and the second argument the actual value.

```
void display_char(alt_u8 digit, alt_u8 character)
{
    switch (digit)
    {
    case 1:
        segment_write(display_1, character);
        break;
    case 2:
        segment_write(display_2, character);
        break;
    case 3:
        segment_write(display_3, character);
        break;
    case 4:
        segment_write(display_all, character);
        break;
    default:
        break;
    }
}
```

Writing direct to register

By calling the following function it is possible for the user to directly write to the components register. All the other functions are using this function as well.

```c
static void segment_write(alt_u8 reg_number, alt_u8 data)
{
    alt_u32 data_send = 0;
    alt_u16 fill = 0x0000;
    //combine input parameters to 32 bit long
    data_send = (reg_number << 24) | (data << 16) | fill;
    //write to register
    IOWR_32DIRECT(SEVEN_SEG_IP_0_BASE, 4, data_send);
    //printf("data %x\n", data_send);
}
```

Example

To test the functionally of the seven segment driver the user can use the following example.

```c


#include <stdio.h>
#include <system.h>
#include "seven_segment.h"

int main()
{
    printf("Hello from Nios II!\n");
  // init call for seven segment driver
  // needs to be called before using the component
    seven_segment_init();

    alt_u8 a = 20;
    alt_u8 b = 33;
    alt_u8 c = 99;
    alt_u8 d = 2;

    while (1) {
      //function to write directly to all displays
        display_char(4,a);
        for(size_t i = 0; i < 500000; i++);
        //write to display group 3
        display_char(3,b+d);
        for(size_t i = 0; i < 500000; i++);
        //write to display group 1
        display_char(1,c);
        for(size_t i = 0; i < 500000; i++);
        //write to display group 2
        display_char(2,d);
        for(size_t i = 0; i < 500000; i++);
    }
return 0;
}
```