SDC test journal

Mikkel Wohlert Jensen

2024-12-23

Contents

1 Purpose

The shutdown circuit code is designed in both RTL (VHDL) and software (C) on the Zynq, meaning that experimentation has to be done to verify logic on both platforms. This journal describes the following tests performed using the code developed on both parts of the chip:

- Testbench simulation of SDC RTL.
- Experiments using the SDC library

2 Materials used

- Zynq-Based Master Controller,
- Testbench for Zynq-Based Master Controller,
- EL302Tv Power Supply from THURLBY THANDAR INSTRUMENTS
- Vivado Design Suite 2024.1
- Vitis Unified IDE 2024.1

3 Simulation of SDC RTL

The power supply is connected to the power pins of the Master Controller PCB and a USB cable is connected between the Master Controller PCB and a PC. The SDC RTL code as well as the testbench code is shown at the bottom of this journal. The relevant parts of the IP integrator window can be seen below:

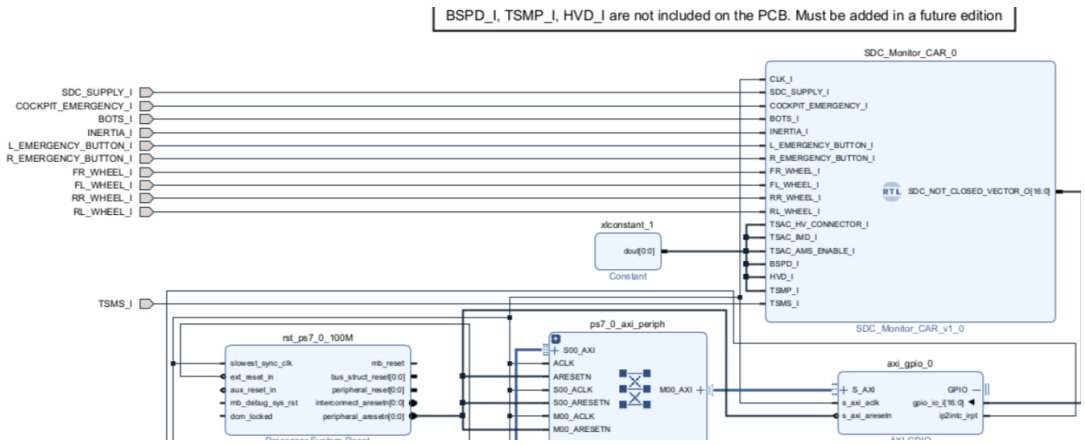

Figure 1: The relevant parts of the IP integrator window.

4 Results

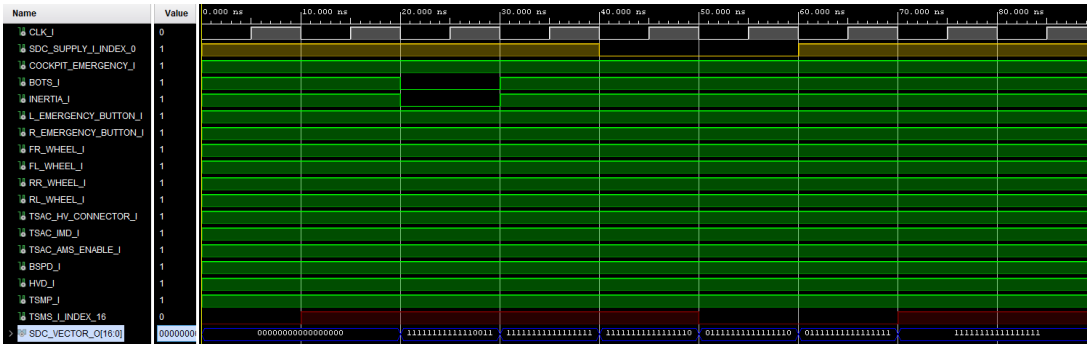

Figure 2: A snapshot of the simulation.

Do note that the logic presented in the testbench is inverted of the physical pins to the Zynq. That is, if a SDC node is closed it will input a "0" to the Zynq. This has no effect on how the algoritm works.

5 Experiments Using the SDC Library

Code for the developed SDC library in C can be found at the bottom of this journal. The API includes the functions:

```
1     int init_SDC();
2     void SDC_ISR(void *CallbackRef);
3     void SDC_open_read_as_register(uint32_t* SDC_register);
4     void SDC_open_read_as_array (uint32_t SDC_array[ NO_OF_SDC_NODES ]
          );
5     bool is_SDC_Completed();
6     const char* get_first_broken_node();
7     bool get_SDC_inertia_Open_Status();
```

Listing 1: SDC library functions.

Some of these functions will be tested here via a printout to a terminal. The MC testbench has jumpers that act as each SDC node (see report). Jumpers were removed and put back arbitrarily to simulate the physical SDC.

Interrupt test:

The first test will be of the interrupt using the code in main.c as shown below, which only initializes the SDC code: setting up the AXI GPIO instance and the GPIO interrupt used:

`init_SDC()`

SDC Monitoring Register and Array Fetching Test:

Next, a test is done of the API for fetching fetching the state of each SDC node from the PL-side to the PS via `SDC_Open_read_as_register()` and `SDC_open_read_as_array()`:

```
1     u32 SDC_open_register;
2     SDC_open_read_as_register(&SDC_open_register);
3     printf("\n Register decimal value showing status of all open SDC
          nodes: %u", SDC_open_register);
```

Listing 2: SDC_Open_read_as_register()

```
    u32 SDC_open_array[NO_OF_SDC_NODES];
    SDC_open_read_as_array(SDC_open_array);
    printf("\nArray showing status of all open SDC nodes:\n");
    for (int i = 0; i < NO_OF_SDC_NODES; i++)
    {
        printf("Node %d: %u\n", i, SDC_open_array[i]);
    }
    for(int i = 0; i < 200000000; i++) // for 2 sec delay
```

Listing 3: SDC_open_read_as_array()

SDC Check Test

Lastly, is a test of the is_SDC_Completed() function, which checks if the entire node is closed as well as a "get"-status function for an individual node:

```
    bool k = is_SDC_Completed();
    printf("\n Is the whole SDC closed: %u", k);

    k = get_SDC_inertia_Open_Status();
    printf("\n Is the inertia switch open: %u", k);
```

Listing 4: is_SDC_Completed()

6 Results of Experiments Using the SDC Library

Below are terminal printouts of the function calls in order.

Interrupt test:

```
Setting up SDC AXI GPIO\\
Setting up SDC AXI Interrupt\\
SDC broken at: TSMS

All nodes connected.\\
SDC broken at: TSMS

All nodes connected.\\
All nodes connected.\\
SDC broken at: TSMS

All nodes connected.\\
SDC broken at: Front Left Wheel

SDC broken at: Front Left Wheel

All nodes connected.\\
SDC broken at: Front Left Wheel

All nodes connected.\\
```

```
All nodes connected.\\
SDC broken at: Front Left Wheel

All nodes connected.\\
SDC broken at: Right Emergency Button

SDC broken at: Right Emergency Button

All nodes connected.\\
All nodes connected.\\
SDC broken at: BOTS

SDC broken at: BOTS

SDC broken at: Rear Left Wheel

SDC broken at: Rear Left Wheel

SDC broken at: BOTS

SDC broken at: BOTS

All nodes connected.\\
SDC broken at: BOTS

All nodes connected.\\
SDC broken at: BOTS

All nodes connected.\\
All nodes connected.\\
SDC broken at: BOTS

All nodes connected.\\
SDC broken at: BOTS

All nodes connected.\\
SDC broken at: BOTS

All nodes connected.\\
SDC broken at: BOTS

All nodes connected.
```

SDC Monitoring Register and Array Fetching Test:

```
Setting up SDC AXI GPIO\\
Setting up SDC AXI Interrupt\\
Register decimal value showing status of all open SDC nodes: 0\\
Array showing status of all open SDC nodes:\\
```

```
Node 0: 0\\
Node 1: 0\\
Node 2: 0\\
Node 3: 0\\
Node 4: 0\\
Node 5: 0\\
Node 6: 0\\
Node 7: 0\\
Node 8: 0\\
Node 9: 0\\
Node 10: 0\\
Node 11: 0\\
Node 12: 0\\
Node 13: 0\\
Node 14: 0\\
Node 15: 0\\
Node 16: 0\\
s of all open SDC nodes: 66544\\
Array showing status of all open SDC nodes:\\
Node 0: 0\\
Node 1: 0\\
Node 2: 0\\
Node 3: 0\\
Node 4: 1\\
Node 5: 1\\
Node 6: 1\\
Node 7: 1\\
Node 8: 1\\
Node 9: 1\\
Node 10: 0\\
Node 11: 0\\
Node 12: 0\\
Node 13: 0\\
Node 14: 0\\
Node 15: 0\\
Node 16: 1\\
s of all open SDC nodes: 66544\\
Array showing status of all open SDC nodes:\\
Node 0: 0\\
Node 1: 0\\
Node 2: 0\\
Node 3: 0\\
Node 4: 1\\
Node 5: 1\\
Node 6: 1\\
Node 7: 1\\
Node 8: 1\\
Node 9: 1\\
```

```
Node 10: 0\\
Node 11: 0\\
Node 12: 0\\
Node 13: 0\\
Node 14: 0\\
Node 15: 0\\
Node 16: 1

Register decimal value showing status of all open SDC nodes: 66558\\
Array showing status of all open SDC nodes:\\
Node 0: 0\\
Node 1: 1\\
Node 2: 1\\
Node 3: 1\\
Node 4: 1\\
Node 5: 1\\
Node 6: 1\\
Node 7: 1\\
Node 8: 1\\
Node 9: 1\\
Node 10: 0\\
Node 11: 0\\
Node 12: 0\\
Node 13: 0\\
Node 14: 0\\
Node 15: 0\\
Node 16: 1
```

SDC Check Test:

```
Is the inertia switch open: 0\\
Is the whole SDC closed: 1\\
Is the inertia switch open: 0\\
Is the whole SDC closed: 1\\
Is the inertia switch open: 1\\
Is the whole SDC closed: 0\\
Is the inertia switch open: 1\\
Is the whole SDC closed: 0\\
Is the inertia switch open: 1\\
Is the whole SDC closed: 0
```

7 Conclusion

- The RTL-side SDC works as intended, although the testbench may lead the reader to believe that a HIGH state for an SDC node is to be considered a "closed"-state. This is however not the case, as the physical inputs to the pins are inverted, so a closed node is a digital "0".

- The SDC software works as intended mostly, but a debouncer might be needed, as the program kept interrupting whilst a jumper was getting removed or inserted. Besides that, the code is considered to be ready for optimization for when it gets implemented into the car.