Points	Grade

Team: -

1525864 Andreas Glinserer

Digital Integrated Circuits Lab (LDIS)

384.088, Summer Term 2019

Supervisors:
Christian Krieg, David Radakovits, Axel Jantsch

Task 1: Digital Thermometer

1 Problem statement

The goal was to design a digital thermometer with the Nexys 4 DDR FPGA Board. To use was one of the two temperature sensors on the board. One sensor is especially for measuring the temperature and to read out via i2c, while the other one is part of the accelerometer sensor and is to read out with SPI. The samling rate of the sensor should be defined pre-synthesis. The data then should go to an DSP part which was to be implemented as an moving average filter. The filter width should be able to be changed during runtime. The output of the DSP should be displayed on the 8 BCD segments on the board.

Figure 1: Block diagram of the system. Source: Original Task Description

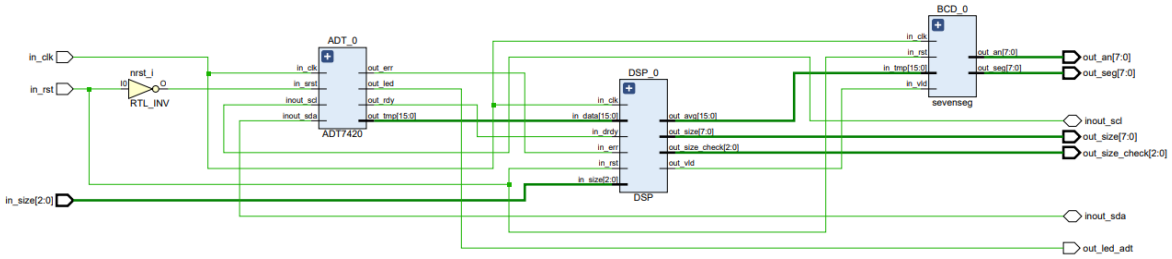


Figure 2: Schematic of the system. Source: Vivado

2 Implementation

In this section I explain how i implemented a possible solution to the proposed problem. I choose to use an active low reset for this task, which is the onboard switch with the name SW15.

2.1 Sampling

For the reading of the temperature i choose the onboard I2C-sensor. To get the values I used the IP-core from the Nexys 4 DDR Demo which includes an readout of this sensor. The sensor itself is an ADT7420 I2C temperature sensor. The output of the sampling part consists of the out_data and the out_vld ports. The out_vld ports is high for the time of once clock cycle. It also has one LED, which toggles everytime valid data was retrieved.

2.1.1 Changes of the IP

The TWI Controller file was changed only in the way to use the ieee.numeric_std.all; library. Other than that this file is original.

To the original readoutfiles were made some changes to get the wished functionality. To get 16 bit readouts I added another init vector:

```
118 constant NO_OF_INIT_VECTORS : natural := 4; -- number of init vectors in TempSensInitMap
119 constant DATA_WIDTH : integer := 1 + 8 + 8; -- RD/WR bit + 1 byte register address + 1 byte data
120 constant ADDR_WIDTH : natural := natural(ceil(log(real(NO_OF_INIT_VECTORS), 2.0)));
121
122 type TempSensInitMap_type is
123         array (0 to NO_OF_INIT_VECTORS-1) of std_logic_vector(DATA_WIDTH-1 downto 0);
124 signal TempSensInitMap: TempSensInitMap_type := (
125 IRD & x"0B" & x"CB", -- Read ID R[0x0B]=0xCB
126 IWR & x"2F" & x"00", -- Reset R[0x2F]=don't care
127 IRD & x"0B" & x"CB", -- Read ID R[0x0B]=0xCB
128 IWR & x"03" & x"80"  -- configure it for 16bit
129 );
```

To get the sampling timing right i added an extra state in which the I2C waits.

```
101 constant SAMPLECNT : NATURAL := natural(ceil(real(clockfreq*1_000_000/Samples)))*2;
102
103 -- State Machine states definition
```

```
104 type state_type is (
105         stIdle, -- Idle State
106         stInitReg,  -- Send register address from the init vector
107         stInitData, -- Send data byte from the init vector
108         stRetry,    -- Retry state reached when there is a bus error, will retry RETRY_COUNT times
109         stReadTempR,  -- Send temperature register address
110         stReadTempD1, -- Read temperature MSB
111         stReadTempD2, -- Read temperature LSB
112         stError, -- Error state when reached when there is a bus error after a successful init; stays
                here until reset
113         stWait  -- State to wait for the next sample
114 );
115 signal state, nstate : state_type;
```

The new state was also added to the ReadyFlag process:

```
252 -------------------------------------------------------------------------------
253 -- Ready Flag
254 -------------------------------------------------------------------------------
255 ReadyFlag: process (in_clk)
256 begin
257         if Rising_Edge(in_clk) then
258                 if (state = stIdle or state = stError) or state = stWait then
259                         fReady <= false;
260                 elsif (state = stReadTempD2 and twiDone = '1' and twiErr = '0') then
261                         fReady <= true;
262                 end if;
263         end if;
264 end process;
```

In the OUTPUT_DECODE process another when statement got added.

```
349 when stWait =>
350         null;
```

In the NEXT_STATE_DECODE process the stReadTempD2 state got changed and another state got added:

```
410 when stReadTempD2 =>
411         if (twiDone = '1') then
412                 if (twiErr = '1') then
413                         nstate <= stError;
414                 else
415                         nstate <= stWait;
416 --                  nstate <= stReadTempR; -- old version
417                 end if;
418         end if;
419
420 when stWait =>
421         if waitSample = SAMPLECNT then
422                 nstate <= stReadTempR;
423         end if;
```

And a whole new process got added in which the counting happens for the sampling rate:

```
265 -------------------------------------------------------------------------------
266 -- Sample counter wait
267 -------------------------------------------------------------------------------
268 SAMPLEWAIT : process(in_clk)
269 begin
270         if rising_edge(in_clk) then
271                 if state = stWait then
272                         if waitSample = SAMPLECNT then
273                                 out_led <= led_helper;
274                                 led_helper <= not led_helper;
275                         else
276                                 waitSample <= waitSample+1;
277                         end if;
278
279                 else
280                         waitSample <= 0;
281                 end if;
282         end if;
283 end process;
```

2.2 DSP

The moving average is implemented as a state machine. The size of the filter window is determined by the 3 switches SW2, SW1 and SW0. Each switch equals a number to the power of 2. SW0 = 1, SW1=2 and SW2=4. For example if SW2 is switched on and the others off then the size equals $2^4 = 16$. or if SW1 and SW2 are on then the size equals $2^{2+1} = 8$. It is done this way to make the divison for the mean value easier, since divison by a number which is a power of two equals a bitshift to the right.

The formula of the mean averge is dependent on the size which gets choosen by the switches.

$$y = \sum_{i=0}^{n} x[i] \tag{1}$$

Where i represents the last value in the buffer. This way the mean average is causal but the mean value is shifted if you print it against the original values.

I choose to implement it with a state machine. I also tried it with for loops but it took very long to synthesize. The synthesizing time shrunk from greater than 25 minutes down to less than 7 minutes.

When the size gets changed the buffer gets set to zero. It then gets slowly refilled and the temperature which gets sent to the BCD slowly increases, because it uses the fixed size for the division. It is done this way to show the working of the filter on the FPGA.

2.3 Output

The output part gets the averaged temperature. The hard part is to get from the binary value to a representation which fits on the BCD. One cannot just take the usual 4 bit approach for a segment because in the decimal system values only range from 0 to 9. So if the value would be an B in hex one needs to subtract from the current position and add to the next position. This problem is solved by using the double dabble algorithm (explanation on Wikipedia or Youtube).

Since the BCD share the an anode contact there is also the need to multiplex the data which gets to the corresponding outputs. This is done with the out_an outputs. It is also important to notice that the active select anode is active low.

This is solved in two processes. The first process takes the data in if the valid flag is high. Contrary to DSP part it calculates the double dabble with for loops instead of a state machine. It then saves its solutions to variables. The second process takes these solutions, converts it to an integer format and uses these as indices for a predefined map of outputs. Furthermore the second process does this about 800 times per second to get a refresh rate of 100Hz per BCD. This is done with a simple counter signal.

3 Reports

3.1 Timing

After the first implementation

```
report_timing
INFO: [Timing 38-91] UpdateTimingParams: Speed grade: -3, Delay Type: max.
INFO: [Timing 38-191] Multithreading enabled for timing update using a maximum of 4 CPUs
INFO: [Timing 38-35] Done setting XDC timing constraints.
INFO: [Timing 38-78] ReportTimingParams: -max_paths 1 -nworst 1 -delay_type max -sort_by slack.
Copyright 1986-2017 Xilinx, Inc. All Rights Reserved.
------------------------------------------------------------------------------------
| Tool Version : Vivado v.2017.4 (lin64) Build 2086221 Fri Dec 15 20:54:30 MST 2017
| Date         : Mon Apr  8 20:52:07 2019
| Host         : ldis running 64-bit Debian GNU/Linux 9.3 (stretch)
| Command      : report_timing
| Design       : digitherm
| Device       : 7a100t-csg324
| Speed File   : -3  PRODUCTION 1.20 2017-11-01
------------------------------------------------------------------------------------

Timing Report

Slack (MET) :             3.570ns  (required time - arrival time)
  Source:                 BCD_0/out_an_reg[1]/C
                            (rising_edge-triggered cell FDRE clocked by sys_clk_pin  {rise@0.000ns fall@10.000ns
                            period=20.000ns})
  Destination:            out_an[1]
                            (output port clocked by sys_clk_pin  {rise@0.000ns fall@10.000ns period=20.000ns})
  Path Group:             sys_clk_pin
  Path Type:              Max at Slow Process Corner
  Requirement:            20.000ns  (sys_clk_pin rise@20.000ns - sys_clk_pin rise@0.000ns)
  Data Path Delay:        7.160ns  (logic 3.481ns (48.618%)  route 3.679ns (51.382%))
  Logic Levels:           1  (OBUF=1)
  Output Delay:           5.000ns
  Clock Path Skew:        -4.235ns (DCD - SCD + CPR)
    Destination Clock Delay (DCD):    0.000ns = ( 20.000 - 20.000 )
    Source Clock Delay      (SCD):    4.235ns
    Clock Pessimism Removal (CPR):    0.000ns
  Clock Uncertainty:      0.035ns  ((TSJ^2 + TIJ^2)^1/2 + DJ) / 2 + PE
    Total System Jitter     (TSJ):    0.071ns
    Total Input Jitter      (TIJ):    0.000ns
    Discrete Jitter          (DJ):    0.000ns
    Phase Error              (PE):    0.000ns

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                         (clock sys_clk_pin rise edge)
                                                    0.000     0.000 r
    E3                                              0.000     0.000 r  in_clk (IN)
                         net (fo=0)                 0.000     0.000    in_clk
    E3                   IBUF (Prop_ibuf_I_O)       1.330     1.330 r  in_clk_IBUF_inst/O
                         net (fo=1, routed)         1.525     2.855    in_clk_IBUF
    BUFGCTRL_X0Y16       BUFG (Prop_bufg_I_O)       0.076     2.931 r  in_clk_IBUF_BUFG_inst/O
                         net (fo=2366, routed)      1.303     4.235    BCD_0/in_clk_IBUF_BUFG
    SLICE_X1Y74          FDRE                                      r  BCD_0/out_an_reg[1]/C
  -------------------------------------------------------------------    -------------------
    SLICE_X1Y74          FDRE (Prop_fdre_C_Q)       0.341     4.576 r  BCD_0/out_an_reg[1]/Q
```

```
                           net (fo=1, routed)           3.679     8.254    out_an_OBUF[1]
    K2                     OBUF (Prop_obuf_I_O)         3.140    11.394 r  out_an_OBUF[1]_inst/O
                           net (fo=0)                   0.000    11.394    out_an[1]
    K2                                                                r  out_an[1] (OUT)
  ------------------------------------------------------------------------------------------------

                           (clock sys_clk_pin rise edge)
                                                       20.000    20.000 r
                           clock pessimism              0.000    20.000
                           clock uncertainty           -0.035    19.965
                           output delay                -5.000    14.965
  ------------------------------------------------------------------------------------------------
                           required time                         14.965
                           arrival time                         -11.394
  ------------------------------------------------------------------------------------------------
                           slack                                 3.570
```

3.2 Utilization

3.2.1 Summary

I have the whole report further down. Here is the condensed information in a picture.

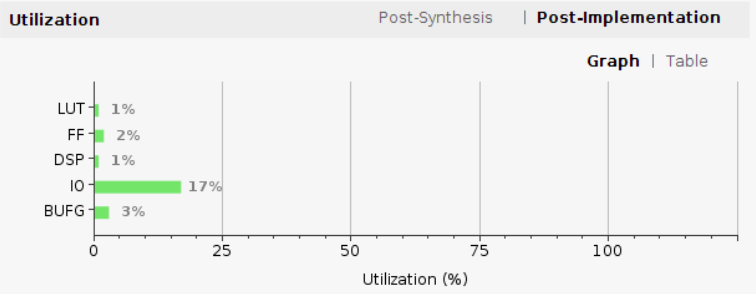
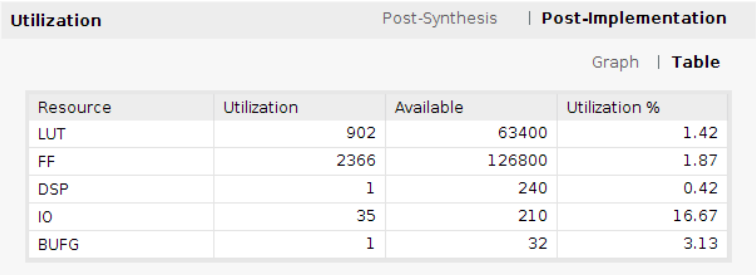


Figure 3: Utilization graph



Resource	Utilization	Available	Utilization %
LUT	902	63400	1.42
FF	2366	126800	1.87
DSP	1	240	0.42
IO	35	210	16.67
BUFG	1	32	3.13

Figure 4: Utilization table

I use more ports than needed because i output extra information with the LEDs and also use the whole BCD. The IO could be reduced this way, because who needs 4 digits after the comma. Something thats very interesting is the fact that my design uses an DSP. This is contrary to the other design i saw to compare. The DSP is the DSP48E1 and it is used in the BCD part of my design. It is needed in the part where the number gets calculated for the fractional part of the output temperature.

3.2.2 Whole Report

```
report_utilization
Copyright 1986-2017 Xilinx, Inc. All Rights Reserved.
------------------------------------------------------------------------------------
| Tool Version : Vivado v.2017.4 (lin64) Build 2086221 Fri Dec 15 20:54:30 MST 2017
| Date         : Mon Apr  8 13:28:18 2019
| Host         : ldis running 64-bit Debian GNU/Linux 9.3 (stretch)
| Command      : report_utilization
| Design       : digitherm
| Device       : 7a100tcsg324-3
| Design State : Routed
------------------------------------------------------------------------------------

Utilization Design Information

Table of Contents
-----------------
1. Slice Logic
1.1 Summary of Registers by Type
2. Slice Logic Distribution
3. Memory
4. DSP
5. IO and GT Specific
6. Clocking
7. Specific Feature
8. Primitives
```

9. Black Boxes
10. Instantiated Netlists

1. Slice Logic

Site Type	Used	Fixed	Available	Util%
Slice LUTs	902	0	63400	1.42
LUT as Logic	902	0	63400	1.42
LUT as Memory	0	0	19000	0.00
Slice Registers	2366	0	126800	1.87
Register as Flip Flop	2366	0	126800	1.87
Register as Latch	0	0	126800	0.00
F7 Muxes	259	0	31700	0.82
F8 Muxes	128	0	15850	0.81

1.1 Summary of Registers by Type

Total	Clock Enable	Synchronous	Asynchronous
0	_	-	-
0	_	-	Set
0	_	-	Reset
0	_	Set	-
0	_	Reset	-
0	Yes	-	-
0	Yes	-	Set
0	Yes	-	Reset
27	Yes	Set	-
2339	Yes	Reset	-

2. Slice Logic Distribution

Site Type	Used	Fixed	Available	Util%
Slice	644	0	15850	4.06
SLICEL	440	0		
SLICEM	204	0		
LUT as Logic	902	0	63400	1.42
using O5 output only	0			
using O6 output only	830			
using O5 and O6	72			
LUT as Memory	0	0	19000	0.00
LUT as Distributed RAM	0	0		
LUT as Shift Register	0	0		
LUT Flip Flop Pairs	198	0	63400	0.31
fully used LUT-FF pairs	24			
LUT-FF pairs with one unused LUT output	159			
LUT-FF pairs with one unused Flip Flop	159			
Unique Control Sets	25			

* Note: Review the Control Sets Report for more information regarding control sets.

3. Memory

Site Type	Used	Fixed	Available	Util%
Block RAM Tile	0	0	135	0.00
RAMB36/FIFO*	0	0	135	0.00
RAMB18	0	0	270	0.00

* Note: Each Block RAM Tile only has one FIFO logic available and therefore can accommodate only one FIFO36E1 or one FIFO18E1. However, if a FIFO18E1 occupies a Block RAM Tile, that tile can still accommodate a RAMB18E1

4. DSP

Site Type	Used	Fixed	Available	Util%
DSPs	1	0	240	0.42
DSP48E1 only	1			

5. IO and GT Specific

Site Type	Used	Fixed	Available	Util%
Bonded IOB	35	35	210	16.67
IOB Master Pads	16			
IOB Slave Pads	17			
Bonded IPADs	0	0	2	0.00
PHY_CONTROL	0	0	6	0.00
PHASER_REF	0	0	6	0.00
OUT_FIFO	0	0	24	0.00
IN_FIFO	0	0	24	0.00
IDELAYCTRL	0	0	6	0.00
IBUFDS	0	0	202	0.00
PHASER_OUT/PHASER_OUT_PHY	0	0	24	0.00
PHASER_IN/PHASER_IN_PHY	0	0	24	0.00
IDELAYE2/IDELAYE2_FINEDELAY	0	0	300	0.00
ILOGIC	0	0	210	0.00
OLOGIC	0	0	210	0.00

6. Clocking

Site Type	Used	Fixed	Available	Util%
BUFGCTRL	1	0	32	3.13

BUFIO	0	0	24	0.00
MMCME2_ADV	0	0	6	0.00
PLLE2_ADV	0	0	6	0.00
BUFMRCE	0	0	12	0.00
BUFHCE	0	0	96	0.00
BUFR	0	0	24	0.00

7. Specific Feature

Site **Type**	Used	Fixed	Available	Util%
BSCANE2	0	0	4	0.00
CAPTUREE2	0	0	1	0.00
DNA_PORT	0	0	1	0.00
EFUSE_USR	0	0	1	0.00
FRAME_ECCE2	0	0	1	0.00
ICAPE2	0	0	2	0.00
PCIE_2_1	0	0	1	0.00
STARTUPE2	0	0	1	0.00
XADC	0	0	1	0.00

8. Primitives

Ref Name	Used	Functional Category
FDRE	2339	Flop & Latch
LUT6	700	LUT
MUXF7	259	MuxFx
MUXF8	128	MuxFx
LUT2	68	LUT
LUT5	56	LUT
LUT3	52	LUT
LUT4	50	LUT
LUT1	48	LUT
CARRY4	44	CarryLogic
OBUF	28	IO
FDSE	27	Flop & Latch
IBUF	7	IO
OBUFT	2	IO
DSP48E1	1	**Block** Arithmetic
BUFG	1	Clock

9. Black Boxes

Ref Name	Used

10. Instantiated Netlists

Ref Name	Used

report_utilization: Time (s): cpu = 00:00:00.45 ; elapsed = 00:00:00.62 . Memory (MB): peak = 7024.695 ; gain = 0.000 ; free physical = 91 ; free virtual = 2276