# COL215

# HW Assignment 2 Report

# Stopwatch

- Kushagra Gupta (2021CS50592)

- Pratham Agrawal (2021CS10891)

The motive of the assignment is design and implement a circuit that that is used in a stopwatch using the 4-digit seven segment display, that we designed in our previous hardware assignment using the 16 switches in the Basys3 board. This stopwatch that we design contains 4 main functionalities-

1) Start switch : When start switch moves from 0 → 1, the stopwatch starts running.
2) Pause switch : When pause switch moves from 0 → 1, the stopwatch simply pauses.
3) Continue button : When continue switch moves from 0 → 1, the stopwatch starts running again.
4) Reset button : On reset from 0 → 1, The stopwatch resets to 0 and initialises all over again.

## Our approach-

We modelled our MODULAR circuit using 5 different parts (3 out of which are taken from the previous assignment) as follows-

1) Main – stopwatch (stopwatch.vhd)
2) Counter circuit (counter.vhd)
3) Timing Circuit (timingcircuit.vhd)
4) Main - 7 segment decoder (segment.vhd)
5) Mux (multiplexer.vhd)

## Main-stopwatch –

The stopwatch we must design, displays 3 things

1) minutes,

2) seconds, and

3)1-tenth of a second

Design:

We used the code of 4-digit seven segment display from our previous assignment in order to display digits on all the 4 7-segment displays available on the basys-3 board.

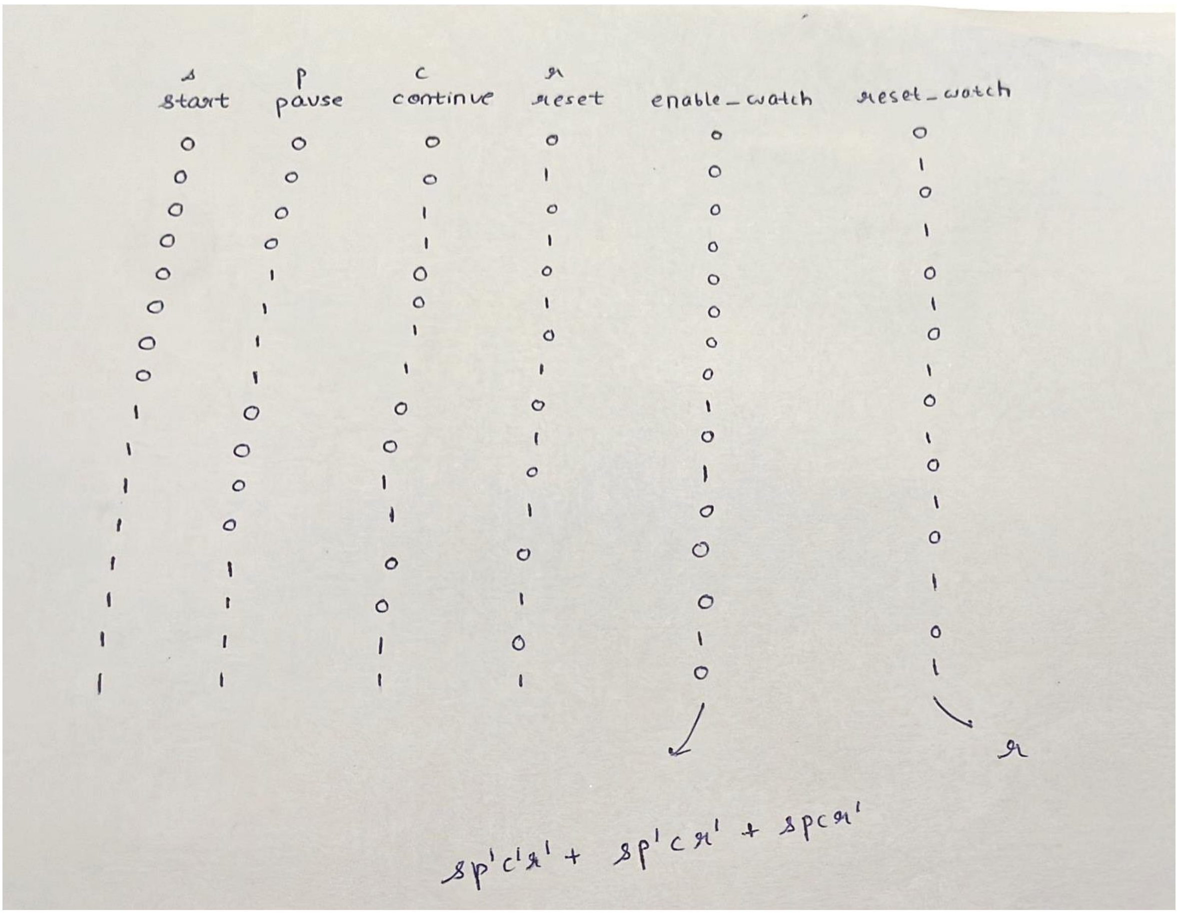
The only difference from previous assignment is that instead of displaying numbers using switches on the board, we are displaying the numbers on the 4 7-segment displays using a counter that we created in our counter.vhd file.

We are using the rising-edge functionality of the clock input in order to increase this counter. This allows us to change the numbers being displayed on the 7-segment displays continuously in intervals of 0.1s.

To perform the 4 functionalities of stopwatch using switches, we created 2 variables, called enable_watch = function of (start, pause, reset, continue) and reset_watch = reset, which represent whether the stopwatch will run or not.

## Counter.vhd file-

We designed our code of counter.vhd in such a way that after every second, the 7-segment display that is displaying the 1-tenth of a second resets to 0, similarly after every minute, the seconds display resets to 0 and after every 10 minutes, the minutes display resets to 0 (we did this by taking mod 10).



| s start | p pause | c continue | r reset | enable_watch | reset_watch |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 | 1 |

r

$sp'c'r' + sp'cr' + spcr'$

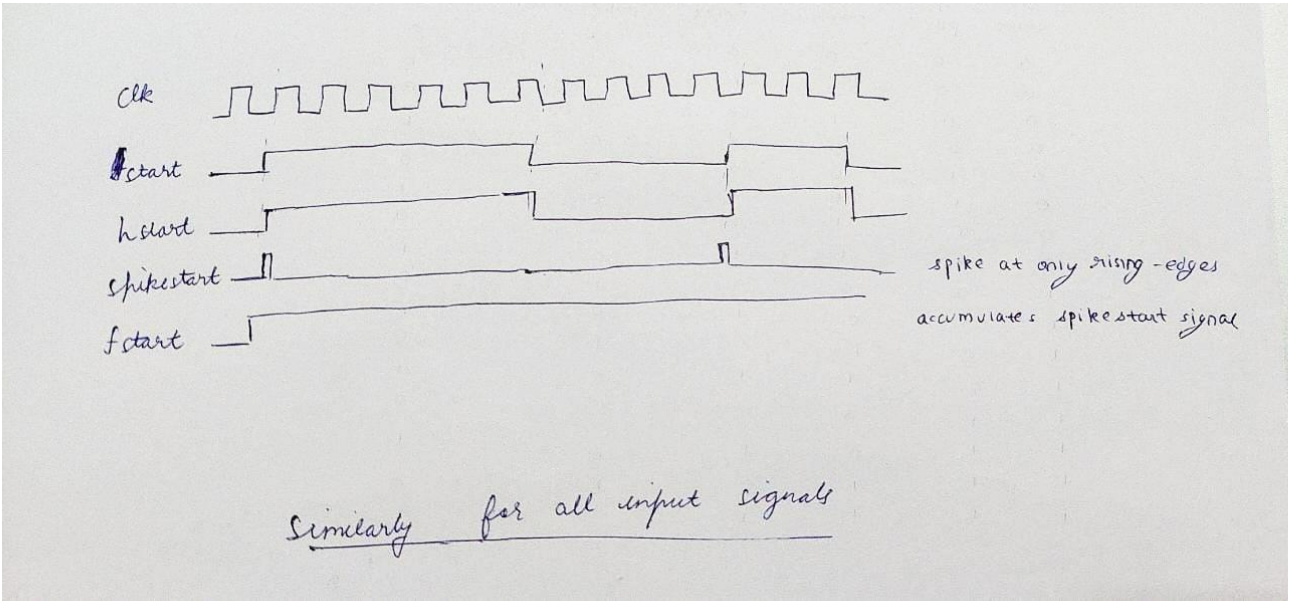
Fig.1 – Obtaining Enable_Watch and Reset_Watch from the current values of 4 switches

We defined the counter.vhd file using our clock input and the 4 switches input to Obtain Enable_Watch and Reset_Watch from the current values of 4 switches, by making a Truth Table (or Reducing via Kmap) the utility functions to start, reset, pause or continue the stopwatch

Counter2.vhd – Implementation of the Enable_Watch and Reset_Watch utilities using the previous value of the switches by the current input before the rising edge of the clock to observe the transition of a flip flop switch from 0 to 1.

Note that the spike is obtained by taking not start and hstart where hstart is defined as start after the rising edge of the clock.

The spike pulse so obtained tells us when to change the counter fstart or Enable_Watch to 1 whereas the spike of reset tells vice versa to set it to 0. (similar logic for pause and continue)


Clk
start
hstart
spikestart
fstart
spike at only rising-edges
accumulates spikestart signal
Similarly for all input signals

These variables depend on the already existing values of the 4 switches – the start, pause, continue and the reset switches; and the new values of these 4 switches.

We are able to store the events of switching these 4 switches from *low* to *high* using 4 helper signals, which stores the already existing states of these switches.

And our final signal which represents the event of switching the switch *high* is defined to be *high* when new state is *high* and the helper signal state is *low* (that is already existing state is low), where we start to increment the counter q and our signal "b" if q mod 10^7=0.

## Segment.vhd file-

Now, after we obtain outputs from the counter.vhd file in form of the numbers to be displayed on each 7-segment (in form of vectors b0,b1,b2,b3) and providing this output as input to the segment component in our stopwatch.vhd file which performs the above task (assignment-1).

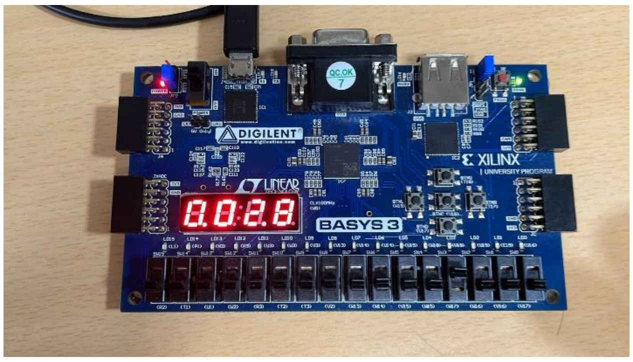
This is used to display the 4 digits on the 4 7-segment displays on the basys board at the same time. It uses 2 components, the multiplexer component and the timing circuit component as explained in the report for previous assignment.

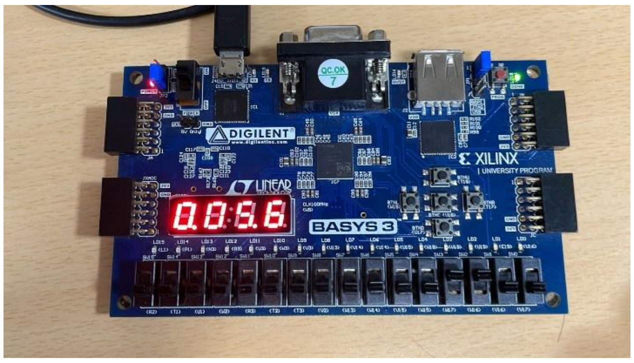
## Constraint File-

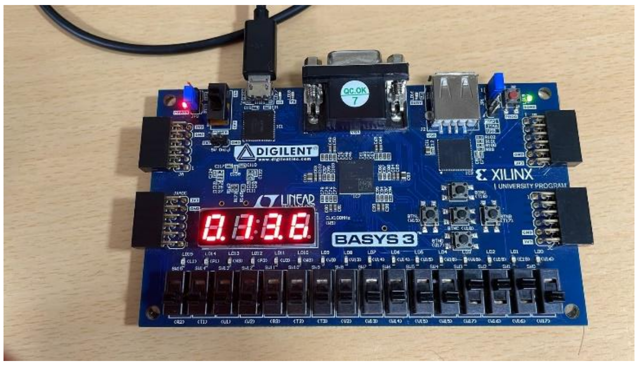
The basys3.xdc file is the constraint file for our Basys-3 FPGA board and

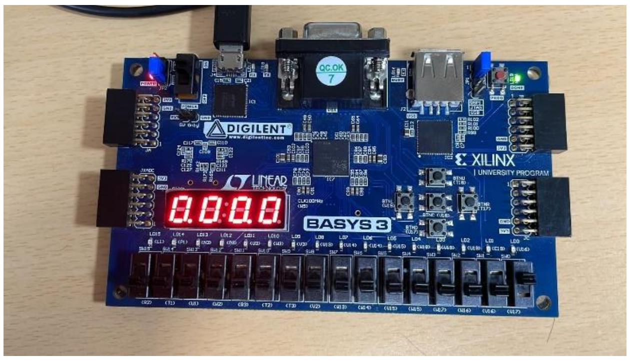
a) The input is our inbuilt clock (period 10ns - freq 100Mhz, tp =10 ns) and the 4 switches that denote the start, pause, continue, and reset switches.
b) The output is our 7 segment cathode bits and anode vector called as s0, s1 s2, s3, s4, s5, s6, s7 for the cathodes G,F,E,D,C,B,A and the vector with 0 for the digit lighted up, and dp cathode, which we use to light us the decimal points as shown in the stopwatch display.

Here are a few testing inputs we used to check our strategy:


Start the stopwatch- only start active


Pause- Start and Pause=1

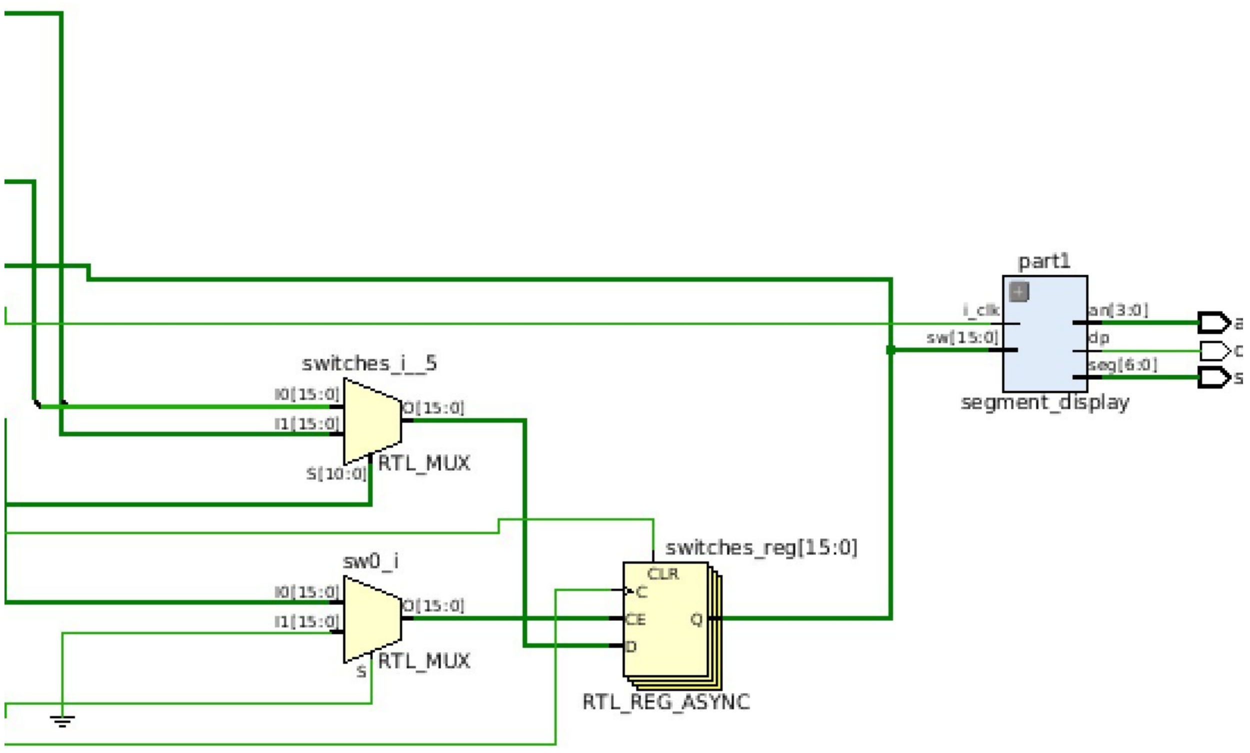

Continue after pause= s,p,c=1


Reset to 0 and stop= s,p,c,r=1

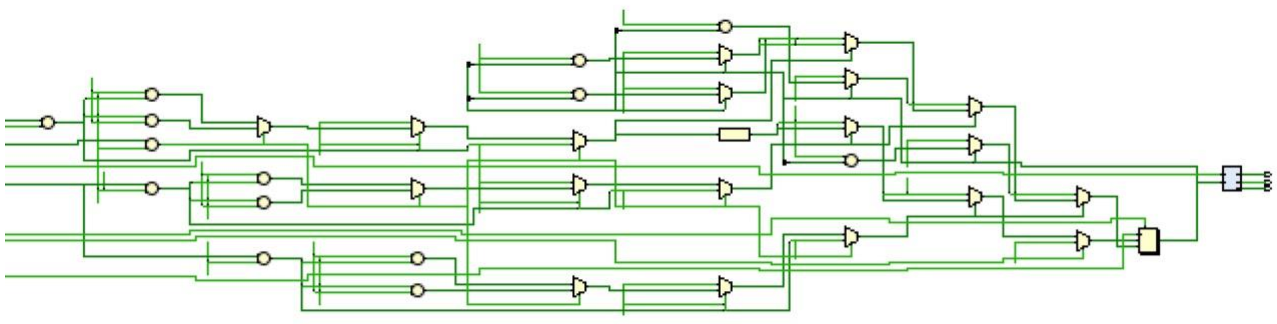
Observe the flipflops and their respective 7 segment display output

Here is a drive link containing a video of the various combinations of input switches we tested on our board and their desired outputs for our stopwatch: [Stopwatch Testing by 4 swtiches]

## Our Synthesized design (and block diagram obtained from the synthesis)



Block Diagram showing our multiplexer in segment, switch inputs for start, pause, continue, reset and respective 7 segment display output

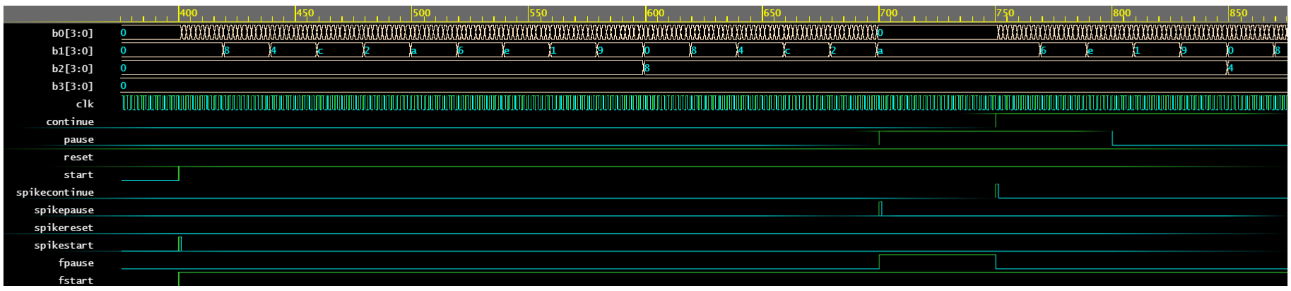


Netlist

Once, we have successfully synthesized our circuit, we run implementation on it and generate the bitstream file (.bit) uploaded used to program the device.

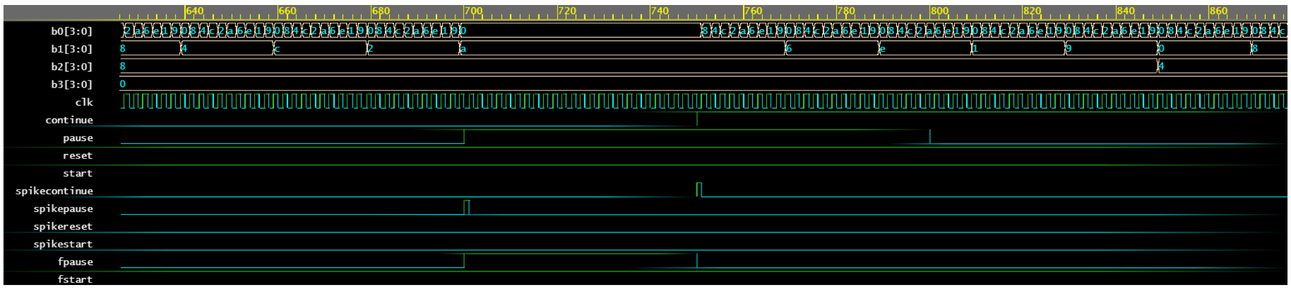
## Testbench input and output (testbench.vhd) -

We used a testbench on our main function where we use an oscillating clock with 10ns timeperiod so it changes parity for every 10 ns so it is 1 for exactly half of the time.
We use our counter2 file for the simulation and set the stopwatch change time to be 10 ns for better visualization of the simulation instead of 0.1 s



Observe the spike obtained when the input signals change value from 0 to 1. Here, b0, b1, b2, b3 are the respective 4 bit inputs to the 7 segment displays (and are in a reverse fashion as per our input so they start from 0, 8 =0001=1, 4=0010 =2 ... and increase as time progresses)

Now, in order to see our oscillating clock and cyclic output more clearly, we can change the period of output from 4 ms to 40 ns (in timing circuit,just for observing the waveform clearly) and we see the following waveforms clearly for the following input:



```
Input for the above waveform
start <= '0' , '1' after 5 ns, '0' after 300 ns, '1' after 400 ns;
pause <= '0', '1' after 700ns, '0' after 800ns;
continue <= '0', '1' after 750 ns;
reset <= '0', '1' after 200 ns;
```

So this completes our testbench analysis and design of the combinational circuit used to realize a stopwatch taking 4 bit flip flops and the Basys3-FPGA clock as input.

## Synthesis report resource counts: Flip-flops, LUTs, BRAMs, and DSPs (.rpt also uploaded)

```
Copyright 1986-2016 Xilinx, Inc. All Rights Reserved.
--------------------------------------------------------------------------------
----------------------------
| Tool Version : Vivado v.2016.4 (lin64) Build 1756540 Mon Jan 23 19:11:19 MST
2017
| Date         : Sun Oct 30 21:23:54 2022
| Host         : dhd running 64-bit Ubuntu 20.04.3 LTS
| Command      : report_utilization -file stopwatch_utilization_synth.rpt -pb
stopwatch_utilization_synth.pb
| Design       : stopwatch
| Device       : 7a35tcpg236-1
| Design State : Synthesized
--------------------------------------------------------------------------------
----------------------------

Utilization Design Information

Table of Contents
-----------------
1. Slice Logic
1.1 Summary of Registers by Type
2. Memory
```

```
3. DSP
4. IO and GT Specific
5. Clocking
6. Specific Feature
7. Primitives
8. Black Boxes
9. Instantiated Netlists

1. Slice Logic
--------------

+-------------------------+------+-------+-----------+-------+
|        Site Type        | Used | Fixed | Available | Util% |
+-------------------------+------+-------+-----------+-------+
| Slice LUTs*             | 2320 |     0 |     20800 | 11.15 |
|   LUT as Logic          | 2320 |     0 |     20800 | 11.15 |
|   LUT as Memory         |    0 |     0 |      9600 |  0.00 |
| Slice Registers         |  112 |     0 |     41600 |  0.27 |
|   Register as Flip Flop |  103 |     0 |     41600 |  0.25 |
|   Register as Latch     |    9 |     0 |     41600 |  0.02 |
| F7 Muxes                |    0 |     0 |     16300 |  0.00 |
| F8 Muxes                |    0 |     0 |      8150 |  0.00 |
+-------------------------+------+-------+-----------+-------+
* Warning! The Final LUT count, after physical optimizations and full
implementation, is typically lower. Run opt_design after synthesis, if not
already completed, for a more realistic count.


1.1 Summary of Registers by Type
--------------------------------

+-------+--------------+-------------+--------------+
| Total | Clock Enable | Synchronous | Asynchronous |
+-------+--------------+-------------+--------------+
| 0     |            _ |           - |            - |
| 0     |            _ |           - |          Set |
| 0     |            _ |           - |        Reset |
| 0     |            _ |         Set |            - |
| 0     |            _ |       Reset |            - |
| 0     |          Yes |           - |            - |
| 1     |          Yes |           - |          Set |
| 8     |          Yes |           - |        Reset |
| 0     |          Yes |         Set |            - |
| 103   |          Yes |       Reset |            - |
+-------+--------------+-------------+--------------+


2. Memory
---------

+----------------+------+-------+-----------+-------+
|    Site Type   | Used | Fixed | Available | Util% |
+----------------+------+-------+-----------+-------+
| Block RAM Tile |    0 |     0 |        50 |  0.00 |
|   RAMB36/FIFO* |    0 |     0 |        50 |  0.00 |
|   RAMB18       |    0 |     0 |       100 |  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


3. DSP
------
```

```
+-----------+------+-------+-----------+-------+
| Site Type | Used | Fixed | Available | Util% |
+-----------+------+-------+-----------+-------+
| DSPs      |    0 |     0 |        90 |  0.00 |
+-----------+------+-------+-----------+-------+


4. IO and GT Specific
---------------------

+-----------------------------+------+-------+-----------+-------+
|          Site Type          | Used | Fixed | Available | Util% |
+-----------------------------+------+-------+-----------+-------+
| Bonded IOB                  |   17 |     0 |       106 | 16.04 |
| Bonded IPADs                |    0 |     0 |        10 |  0.00 |
| Bonded OPADs                |    0 |     0 |         4 |  0.00 |
| PHY_CONTROL                 |    0 |     0 |         5 |  0.00 |
| PHASER_REF                  |    0 |     0 |         5 |  0.00 |
| OUT_FIFO                    |    0 |     0 |        20 |  0.00 |
| IN_FIFO                     |    0 |     0 |        20 |  0.00 |
| IDELAYCTRL                  |    0 |     0 |         5 |  0.00 |
| IBUFDS                      |    0 |     0 |       104 |  0.00 |
| GTPE2_CHANNEL               |    0 |     0 |         2 |  0.00 |
| PHASER_OUT/PHASER_OUT_PHY   |    0 |     0 |        20 |  0.00 |
| PHASER_IN/PHASER_IN_PHY     |    0 |     0 |        20 |  0.00 |
| IDELAYE2/IDELAYE2_FINEDELAY |    0 |     0 |       250 |  0.00 |
| IBUFDS_GTE2                 |    0 |     0 |         2 |  0.00 |
| ILOGIC                      |    0 |     0 |       106 |  0.00 |
| OLOGIC                      |    0 |     0 |       106 |  0.00 |
+-----------------------------+------+-------+-----------+-------+


5. Clocking
-----------

+------------+------+-------+-----------+-------+
|  Site Type | Used | Fixed | Available | Util% |
+------------+------+-------+-----------+-------+
| BUFGCTRL   |    1 |     0 |        32 |  3.13 |
| BUFIO      |    0 |     0 |        20 |  0.00 |
| MMCME2_ADV |    0 |     0 |         5 |  0.00 |
| PLLE2_ADV  |    0 |     0 |         5 |  0.00 |
| BUFMRCE    |    0 |     0 |        10 |  0.00 |
| BUFHCE     |    0 |     0 |        72 |  0.00 |
| BUFR       |    0 |     0 |        20 |  0.00 |
+------------+------+-------+-----------+-------+


6. 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 |
+-------------+------+-------+----------+-------+


7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| LUT6     |  999 |                 LUT |
| LUT5     |  459 |                 LUT |
| LUT3     |  435 |                 LUT |
| CARRY4   |  388 |          CarryLogic |
| LUT1     |  263 |                 LUT |
| LUT4     |  237 |                 LUT |
| LUT2     |  201 |                 LUT |
| FDRE     |  103 |        Flop & Latch |
| OBUF     |   12 |                  IO |
| LDCE     |    8 |        Flop & Latch |
| IBUF     |    5 |                  IO |
| LDPE     |    1 |        Flop & Latch |
| BUFG     |    1 |               Clock |
+----------+------+---------------------+


8. Black Boxes
--------------

+----------+------+
| Ref Name | Used |
+----------+------+


9. Instantiated Netlists
------------------------

+----------+------+
| Ref Name | Used |
+----------+------+
```