COL215 Hardware Assignment 1

Mihir Kaskhedikar, 2021CS10551
Dhruv Ahlawat, 2021CS10556

October 2022

1 Introduction

In this assignment, we design and implement a circuit that takes a 4-digit decimal/hexadecimal number (each digit of this number is 4-bit) from switches in the Basys3 board and displays it on the 4-seven segment displays on the board. The Basys3 board is so designed that at any moment it is only possible to display a fixed digit on some of the 4 displays. (i.e. we cannot write code that will display the digit 1 on the third display and digit 3 on the first display say simultaneously). So in order to display the 4-digit decimal or hexadecimal number on the display, we need to use an **in-built clock**. What the clock does is that it displays a particular digit on a particular display for some time and then displays it again after some time $4\delta t$. Now it is possible to make δt so small that the human eye cannot perceive that the display was not showing any digit for some time. (Called as persistence of vision). So if we wanted to show say the number 1729 on the display, we do the following: display 1 on the first screen in the time duration $0 - \delta t$, display 7 on the second in the duration $\delta t - 2\delta t$, display 2 on the third in the duration $2\delta t - 3\delta t$ and display 9 on the fourth in the duration $3\delta t - 4\delta t$ and keep on continuing this cycle. So at any time exactly 1 digit is shown on exactly 1 screen but the net effect that we perceive is that the number 1729 shown at all points of time. In the following report we explain how we wrote a code in VHDL that allowed us to perform the procedure described above:

2 How to show any digit on the displays?

We are given a hexadecimal(works for decimal too) digit $d = (d_3d_2d_1d_0)_2$ which we want to display on all of the 4 displays. For each digit there must be some fixed set of segments among the 7 segments that must be bright (for a segment to be bright it needs to be "OFF" in the digital logic sense as the segment is an ACTIVE LOW pin). So we can find for each digit the segments that should become bright in order to display it:

d	Bright Segments	Dark Segments
0	A,B,C,D,E,F	G
1	B,C	A,D,E,F,G
2	A,B,G,E,D	C,F
3	A,B,C,D,G	E,F
4	B,C,F,G	A,D,E
5	A,C,D,F,G	B,E
6	A,C,D,E,F,G	B
7	A,B,C	D,E,F,G
8	A,B,C,D,E,F,G	
9	A,B,C,D,F,G	E
10(a)	A,B,C,D,E,G	F
11(b)	C,D,E,F,G	A,B
12(c)	A,D,E,F	B,C,G
13(d)	B,C,D,E,G	A,F
14(e)	A,B,D,E,F,G	C
15(f)	A,E,F,G	B,C,D

Having found the bright segments for each of the digits, we now need to know for each segment all those digits for which this segment is dark, as for these digits only, the signal corresponding to this segment would be in the ON state (because it is an ACTIVE LOW pin).

Segment	Dark for digits
A	1,4,b,d
B	5,6,b,c,f
C	2,c,e,f
D	1,4,7,f
E	1,3,4,5,7,9
F	1,2,3,7,a,d
G	0,1,7,c

Now consider the segment A and let A be the pin corresponding to it. Then we know that A is on if the number to be displayed is $1, 4, 11$ or 13. These numbers in binary notations are $0001, 0100, 1011$ and 1101. Then A is the sum of minterms corresponding to these binary numbers, i.e.

$$A = s_3's_2's_1's_0 + s_3's_2s_1's_0' + s_3s_2's_1s_0 + s_3s_2s_1's_0$$

where s_3, s_2, s_1 and s_0 are the signals corresponding to bits of the digit in base 2 from left to right. Thus we have an equation for signal of each segment whose code we have saved in the file "Decoder4to7" in our submission of source code.

3 A single digit on a single display

The input number is given using 16 bits, 4 for each of its digit. So we create 4 **bit vectors** that contain the 4 bits of each digit. Call these vectors by v_0, v_1, v_2 and v_3 where v_0 is the bit vector corresponding to the first 4 bits from the right, v_1 corresponding to the next 4 bits and so on. Then we keep two selector bits s_1 and s_0 where $s = (s_1s_0)_2$ is the vector index that needs to be displayed on the sth screen from the right (considering 0 indexing). We first create a simple 4 to 1 MUX and then apply the MUX gate to each of the 4 co-ordinates of the 4 bit vectors and put the outputs for each of the MUX gate in a new vector in the corresponding position. So this new vector needs to be displayed. Finally we consider the anode signals corresponding to the displays as a_0, a_1, a_2 and a_3 and select the display by the same selctor bits s_0 and s_1 (note that anodes are also ACTIVE LOW pins, so selecting an anode is putting it in the OFF state). So now for a given s_0 and s_1 we know how to display the $s = (s_1s_0)_2$ digit of the input number on the sth display (0 indexing done from the right). Now it remains to periodically vary $s = (s_1s_0)_2$ with a small time period, which we achieve by the in-built clock.

4 The Clock

The least count of the Basys3 clock is **10ns**. As described in the introduction, we need to select a suitable δt such that the digits seem to appear simultaneously(so δt should not be too large) and they do not flicker too (so δt should not be too small). δt of the order of milliseconds is a suitable choice. Thus we want s=(s_1s_0) to cyclically assume values $00, 01, 10$ and 11 for time δt each. So we write the following code:

```
begin
process (clk)
variable cnt: integer:= 0;
variable r0: std_logic:= '0';
variable r1: std_logic:= '0';
begin

    if(clk' event and clk = '1') then
    cnt := cnt+1;
    if(cnt = 200000) then
    r0 := not r0;
    end if;
```

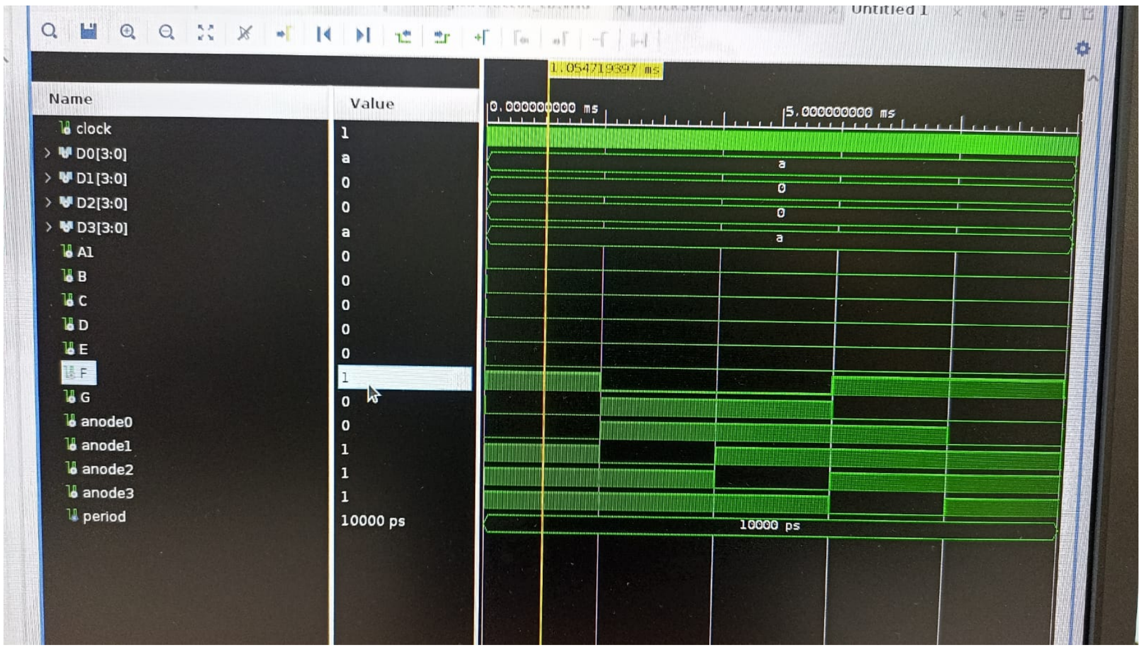
```
if(cnt = 400000) then
r0 := not r0;
r1 := not r1;
cnt := 0;
end if;

s0 <= r0;
s1 <= r1;

end process;
```

In the code the **clk** is the signal for the clock which changes itself after every 10ns. We have also kept 2 dummy variables $r0$ and $r1$ which represent the values that the signals $s0$ and $s1$ need to take after every clock period, and assign these values to $s0$ and $s1$ in the end. We have kept δt to be 2ms. So for that we declare a variable **cnt** which when assumes values that are multiples of **200000**, we change values of $s1$ and $s0$ so that $s = (s_1 s_0)$ now represents the index of the next digit that needs to be shown. So initially both $s1$ and $s0$ are 0. After 2ms we flip $s0$ but retain $s1$, so that they represent 01. After 4ms we flip both $s0$ and $s1$, so that they represent 10 and reinitialize the value of cnt to 0. Now again after 2 ms, we flip $s0$ but retain $s1$ so that they represent 11 and after 4ms, we flip both $s0$ and $s1$ so that they now represent 00 again. Thus this cycle keeps on continuing.

5 Finally...

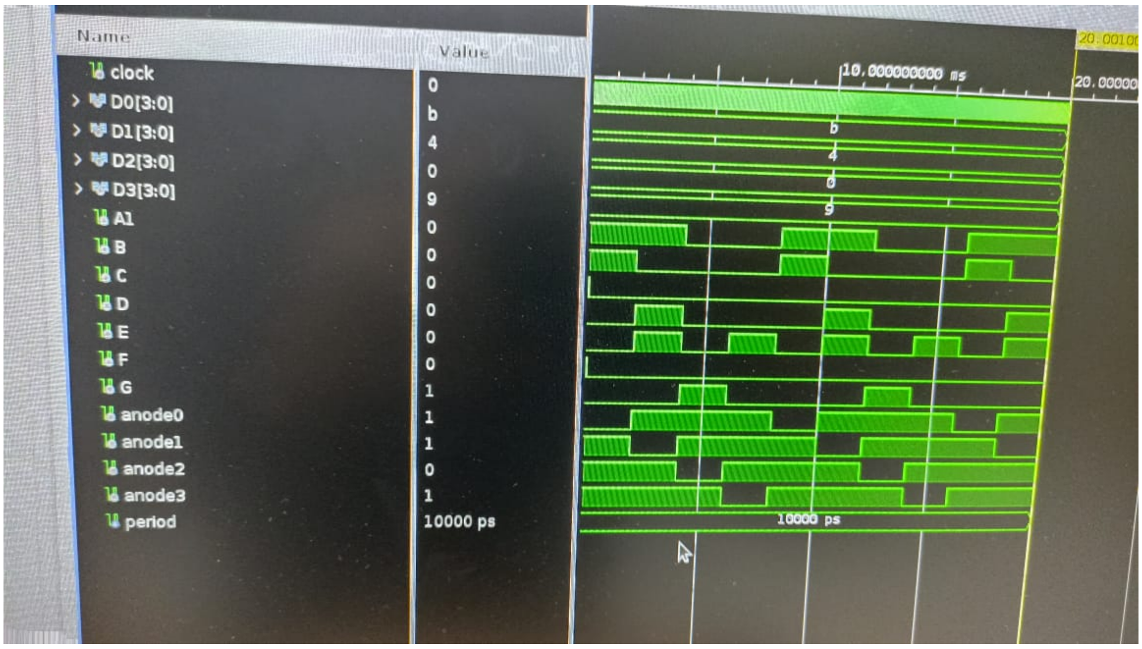
We use the signals **s0** and **s1** generated in the clock as the selector bits and display the number on the display by the procedure describe in section 3.

6 Simulation and Basys3 board snapshots



Name	Value
clock	1
D0[3:0]	a
D1[3:0]	0
D2[3:0]	0
D3[3:0]	a
A1	0
B	0
C	0
D	0
E	0
F	1
G	0
anode0	0
anode1	1
anode2	1
anode3	1
period	10000 ps

Figure 1: Simulation of the hexadecimal number A00A



Name	Value
clock	0
D0[3:0]	b
D1[3:0]	4
D2[3:0]	0
D3[3:0]	9
A1	0
B	0
C	0
D	0
E	0
F	0
G	1
anode0	1
anode1	1
anode2	0
anode3	1
period	10000 ps

Figure 2: Simulation of the hexadecimal number B409

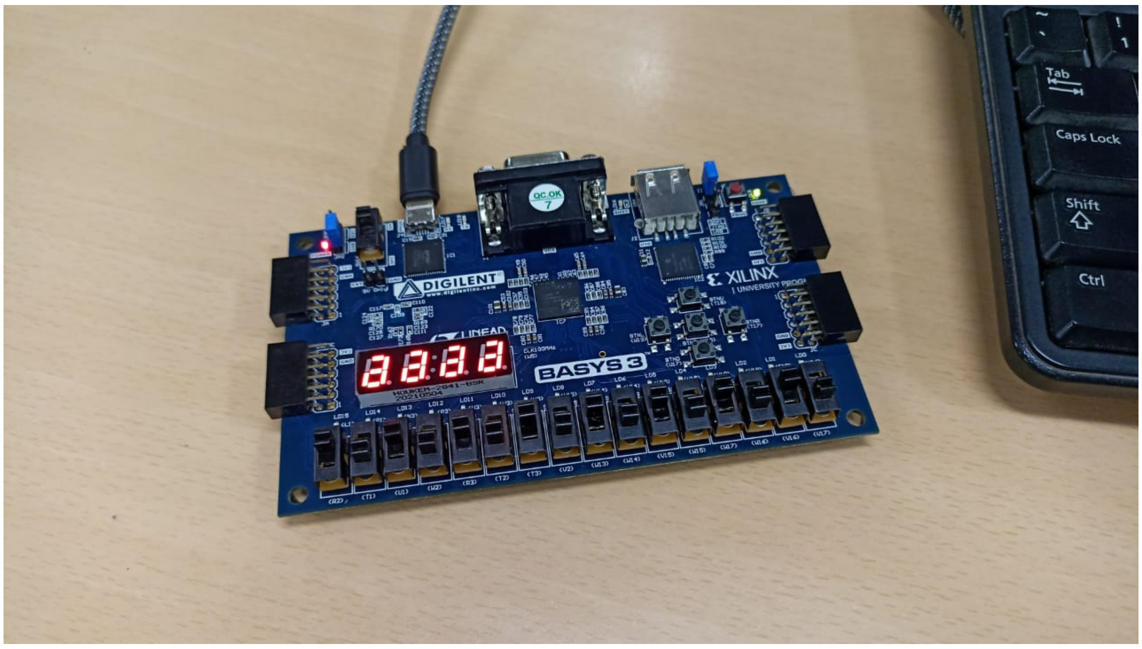


Figure 3: Displaying the hexadecimal number AAAA

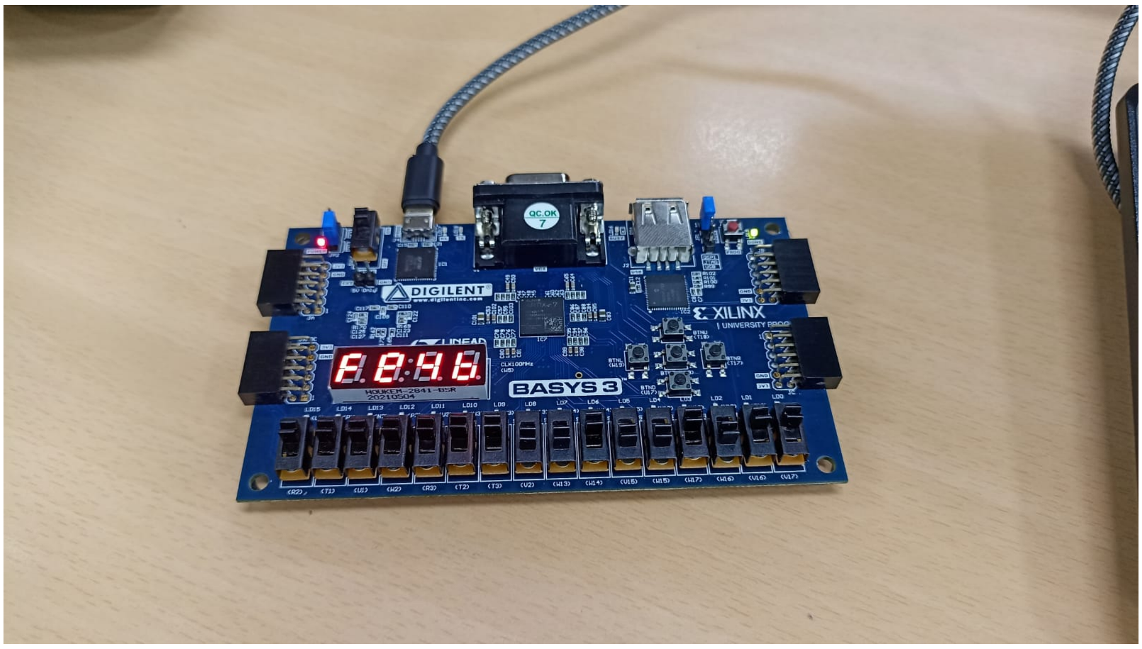


Figure 4: Displaying the hexadecimal number FE4B

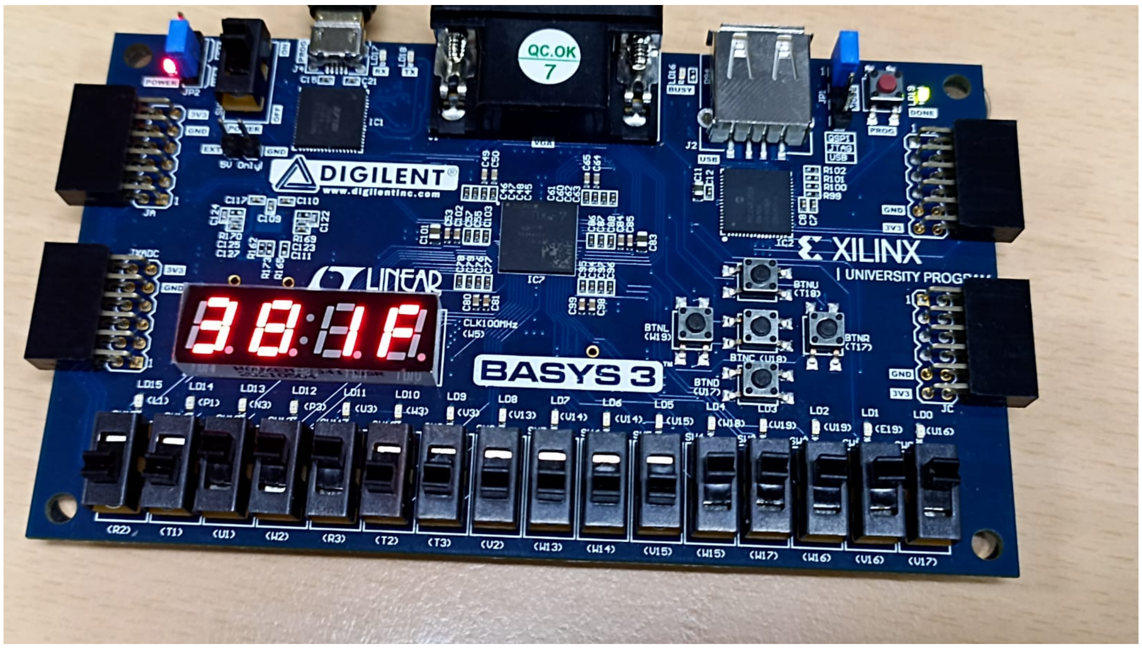


Figure 5: Displaying the hexadecimal number 381F

7 Synthesis Report

Here is the synthesis report from the .rpt file

```
    Copyright 1986-2022 Xilinx, Inc. All Rights Reserved.
-------------------------------------------------------------------------------------------------
| Tool Version : Vivado v.2022.1 (lin64) Build 3526262 Mon Apr 18 15:47:01 MDT 2022
| Date         : Fri Oct 21 03:25:55 2022
| Host         : dhd running 64-bit Ubuntu 20.04.3 LTS
| Command      : report_utilization -file MUXDigitSelector_utilization_synth.rpt -pb

MUXDigitSelector_utilization_synth.pb
| Design       : MUXDigitSelector
| Device       : xc7a35tcpg236-1
| Speed File   : -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 | Prohibited | Available | Util% |
+-------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*             |   24 |     0 |          0 |     20800 |  0.12 |
|   LUT as Logic          |   24 |     0 |          0 |     20800 |  0.12 |
|   LUT as Memory         |    0 |     0 |          0 |      9600 |  0.00 |
| Slice Registers         |   34 |     0 |          0 |     41600 |  0.08 |
|   Register as Flip Flop |   34 |     0 |          0 |     41600 |  0.08 |
|   Register as Latch     |    0 |     0 |          0 |     41600 |  0.00 |
| F7 Muxes                |    0 |     0 |          0 |     16300 |  0.00 |
| F8 Muxes                |    0 |     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 |           - |            - |
| 0     |           Yes |           - |          Set |
| 0     |           Yes |           - |        Reset |
| 0     |           Yes |         Set |            - |
| 34    |           Yes |       Reset |            - |
+-------+---------------+-------------+--------------+


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

+----------------+------+-------+------------+-----------+-------+
|    Site Type   | Used | Fixed | Prohibited | Available | Util% |
+----------------+------+-------+------------+-----------+-------+
| Block RAM Tile |    0 |     0 |          0 |        50 |  0.00 |
|   RAMB36/FIFO* |    0 |     0 |          0 |        50 |  0.00 |
|   RAMB18       |    0 |     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 | Prohibited | Available | Util% |
+-----------+------+-------+------------+-----------+-------+
| DSPs      |    0 |     0 |          0 |        90 |  0.00 |
+-----------+------+-------+------------+-----------+-------+


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

+-----------------------------+------+-------+------------+-----------+-------+
|          Site Type          | Used | Fixed | Prohibited | Available | Util% |
+-----------------------------+------+-------+------------+-----------+-------+
| Bonded IOB                  |   28 |     0 |          0 |       106 | 26.42 |
| Bonded IPADs                |    0 |     0 |          0 |        10 |  0.00 |
| Bonded OPADs                |    0 |     0 |          0 |         4 |  0.00 |
| PHY_CONTROL                 |    0 |     0 |          0 |         5 |  0.00 |
| PHASER_REF                  |    0 |     0 |          0 |         5 |  0.00 |
| OUT_FIFO                    |    0 |     0 |          0 |        20 |  0.00 |
| IN_FIFO                     |    0 |     0 |          0 |        20 |  0.00 |
| IDELAYCTRL                  |    0 |     0 |          0 |         5 |  0.00 |
| IBUFDS                      |    0 |     0 |          0 |       104 |  0.00 |
| GTPE2_CHANNEL               |    0 |     0 |          0 |         2 |  0.00 |
| PHASER_OUT/PHASER_OUT_PHY   |    0 |     0 |          0 |        20 |  0.00 |
| PHASER_IN/PHASER_IN_PHY     |    0 |     0 |          0 |        20 |  0.00 |
| IDELAYE2/IDELAYE2_FINEDELAY |    0 |     0 |          0 |       250 |  0.00 |
| IBUFDS_GTE2                 |    0 |     0 |          0 |         2 |  0.00 |
| ILOGIC                      |    0 |     0 |          0 |       106 |  0.00 |
| OLOGIC                      |    0 |     0 |          0 |       106 |  0.00 |
+-----------------------------+------+-------+------------+-----------+-------+
5. Clocking
-----------

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


6. Specific Feature
-------------------

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


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

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |   34 |        Flop & Latch |
| IBUF     |   17 |                  IO |
| CARRY4   |   16 |          CarryLogic |
| LUT6     |   13 |                 LUT |
| OBUF     |   11 |                  IO |
| LUT4     |    7 |                 LUT |
| LUT2     |    6 |                 LUT |
| LUT3     |    2 |                 LUT |
| LUT5     |    1 |                 LUT |
| LUT1     |    1 |                 LUT |
| BUFG     |    1 |               Clock |
+----------+------+---------------------+


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

+----------+------+
| Ref Name | Used |
+----------+------+
9. Instantiated Netlists
------------------------

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