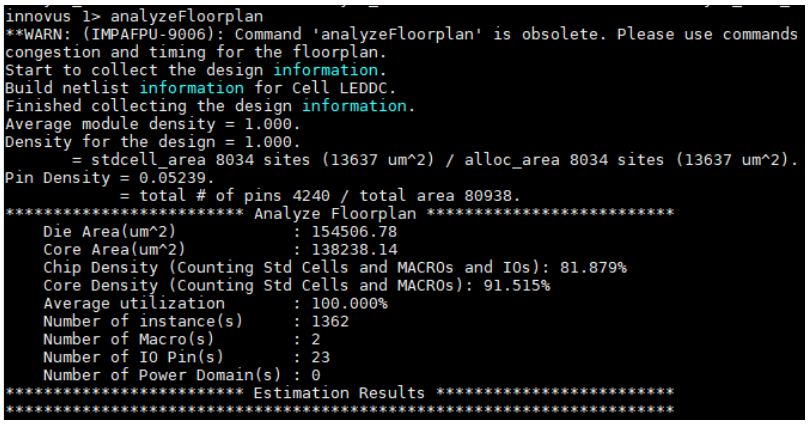
Computer-Aided VLSI System Design, Fall 2018

Final Project - LED Display Controller

Group : 06   Member : b04901015 傅子興 b04901018 游昇融

A. Requirements

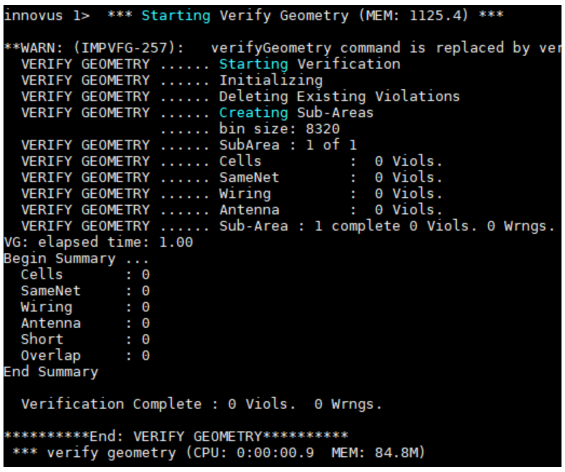
1. Layout Area


```
innovus 1> analyzeFloorplan
**WARN: (IMPAFPU-9006): Command 'analyzeFloorplan' is obsolete. Please use commands
congestion and timing for the floorplan.
Start to collect the design information.
Build netlist information for Cell LEDDC.
Finished collecting the design information.
Average module density = 1.000.
Density for the design = 1.000.
       = stdcell_area 8034 sites (13637 um^2) / alloc_area 8034 sites (13637 um^2).
Pin Density = 0.05239.
            = total # of pins 4240 / total area 80938.
************************ Analyze Floorplan *************************
    Die Area(um^2)            : 154506.78
    Core Area(um^2)           : 138238.14
    Chip Density (Counting Std Cells and MACROs and IOs): 81.879%
    Core Density (Counting Std Cells and MACROs): 91.515%
    Average utilization       : 100.000%
    Number of instance(s)     : 1362
    Number of Macro(s)        : 2
    Number of IO Pin(s)       : 23
    Number of Power Domain(s) : 0
************************ Estimation Results *************************
*********************************************************************
```

Area = 154506.78 um^2

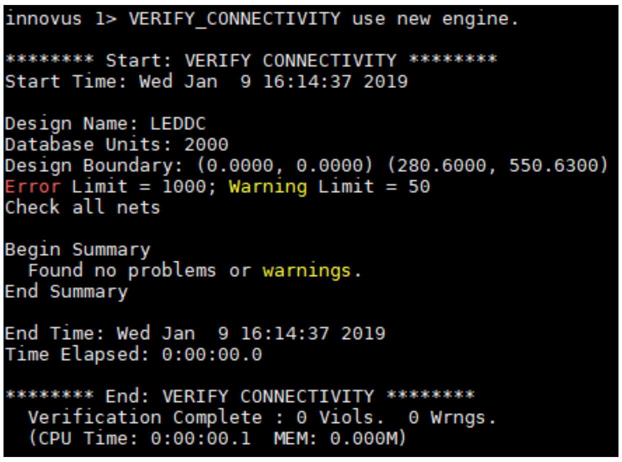
2. Verification Results

DRC Verification Result (Verify Geometry)


```
innovus 1>  *** Starting Verify Geometry (MEM: 1125.4) ***

**WARN: (IMPVFG-257):   verifyGeometry command is replaced by ver
  VERIFY GEOMETRY ...... Starting Verification
  VERIFY GEOMETRY ...... Initializing
  VERIFY GEOMETRY ...... Deleting Existing Violations
  VERIFY GEOMETRY ...... Creating Sub-Areas
                  ...... bin size: 8320
  VERIFY GEOMETRY ...... SubArea : 1 of 1
  VERIFY GEOMETRY ...... Cells          :  0 Viols.
  VERIFY GEOMETRY ...... SameNet        :  0 Viols.
  VERIFY GEOMETRY ...... Wiring         :  0 Viols.
  VERIFY GEOMETRY ...... Antenna        :  0 Viols.
  VERIFY GEOMETRY ...... Sub-Area : 1 complete 0 Viols. 0 Wrngs.
VG: elapsed time: 1.00
Begin Summary ...
  Cells       : 0
  SameNet     : 0
  Wiring      : 0
  Antenna     : 0
  Short       : 0
  Overlap     : 0
End Summary

  Verification Complete : 0 Viols.  0 Wrngs.

**********End: VERIFY GEOMETRY**********
 *** verify geometry (CPU: 0:00:00.9  MEM: 84.8M)
```

DRC Violation Count = 0

LVS Verification Result (Verify Connectivity)

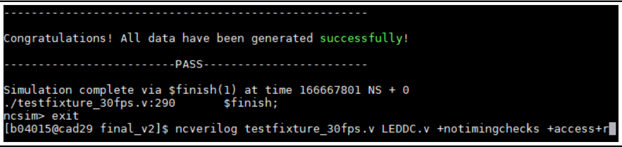

```
innovus 1> VERIFY_CONNECTIVITY use new engine.

******** Start: VERIFY CONNECTIVITY ********
Start Time: Wed Jan  9 16:14:37 2019

Design Name: LEDDC
Database Units: 2000
Design Boundary: (0.0000, 0.0000) (280.6000, 550.6300)
Error Limit = 1000; Warning Limit = 50
Check all nets

Begin Summary
  Found no problems or warnings.
End Summary

End Time: Wed Jan  9 16:14:37 2019
Time Elapsed: 0:00:00.0

******** End: VERIFY CONNECTIVITY ********
  Verification Complete : 0 Viols.  0 Wrngs.
  (CPU Time: 0:00:00.1  MEM: 0.000M)
```

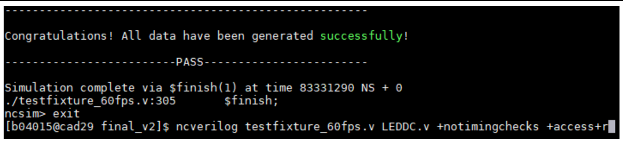
LVS Violation Count = 0

3. Simulation Results

RTL Level Simulation Result

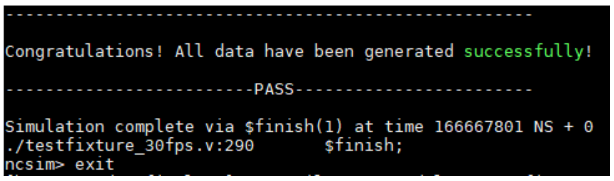
ncverilog testfixture_30|60fps.v LEDDC.v +notimingchecks +access+r


```
--------------------------------------------------

Congratulations! All data have been generated successfully!

------------------------PASS------------------------

Simulation complete via $finish(1) at time 166667801 NS + 0
./testfixture_30fps.v:290       $finish;
ncsim> exit
[b04015@cad29 final_v2]$ ncverilog testfixture_30fps.v LEDDC.v +notimingchecks +access+r
```

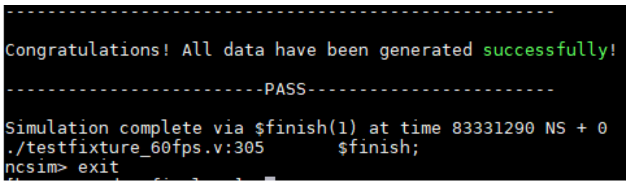

```
--------------------------------------------------

Congratulations! All data have been generated successfully!

------------------------PASS------------------------

Simulation complete via $finish(1) at time 83331290 NS + 0
./testfixture_60fps.v:305       $finish;
ncsim> exit
[b04015@cad29 final_v2]$ ncverilog testfixture_60fps.v LEDDC.v +notimingchecks +access+r
```

Gate Level Simulation Result

ncverilog +ncmaxdelays testfixture_30|60fps.v LEDDC_syn.v sram_512x16/sram_512x16.v sram_256x16/sram_256x16.v tsmc13_neg.v +define+SDF +access+r



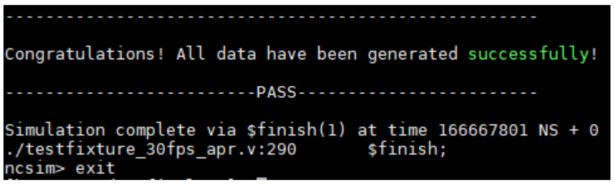
```
----------------------------------------------------

Congratulations! All data have been generated successfully!

-----------------------PASS-----------------------

Simulation complete via $finish(1) at time 166667801 NS + 0
./testfixture_30fps.v:290       $finish;
ncsim> exit
```



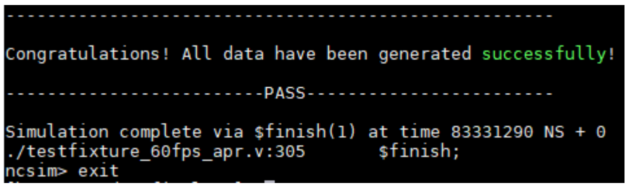
```
----------------------------------------------------

Congratulations! All data have been generated successfully!

-----------------------PASS-----------------------

Simulation complete via $finish(1) at time 83331290 NS + 0
./testfixture_60fps.v:305       $finish;
ncsim> exit
```

Post-layout Simulation Result

ncverilog +ncmaxdelays testfixture_30|60fps_apr.v CHIP_apr.v sram_512x16/sram_512x16.v sram_256x16/sram_256x16.v tsmc13_neg.v +define+SDF +access+r



```
----------------------------------------------------

Congratulations! All data have been generated successfully!

-----------------------PASS-----------------------

Simulation complete via $finish(1) at time 166667801 NS + 0
./testfixture_30fps_apr.v:290       $finish;
ncsim> exit
```



```
----------------------------------------------------

Congratulations! All data have been generated successfully!

-----------------------PASS-----------------------

Simulation complete via $finish(1) at time 83331290 NS + 0
./testfixture_60fps_apr.v:305       $finish;
ncsim> exit
```

4. Applied Technique Explanation
   See Section B. and Section C. for more details.

B. Design Description

The block diagram of our design is shown in the Figure 1.

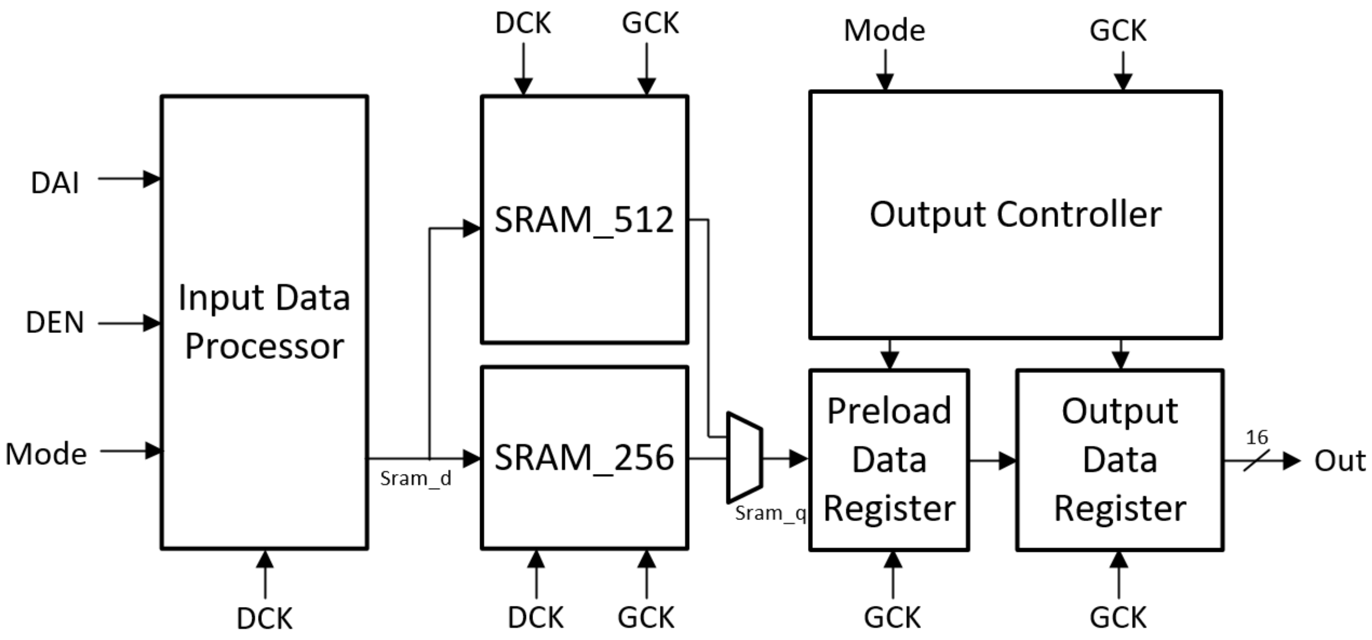


Figure 1. Block diagram of our design

1. Input Processor

   Since the number of rising edges of DCK when the DEN is high is exactly 16. We use a shift register to store the data bits sequentially. When the DEN is low, we store the data into the sram.

2. Output Controller and Register

   The finite state of our design is shown in the Figure 2.

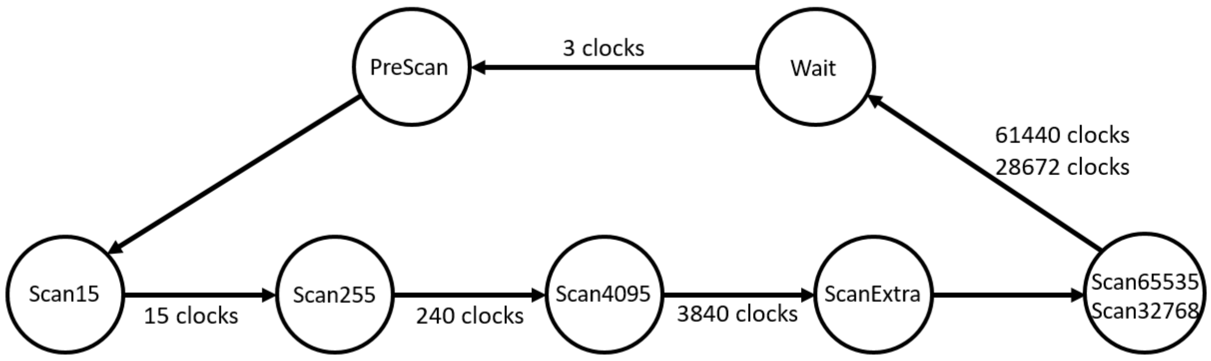


Figure 2. The Finite state machine of the output for 60FPS/30FPS

According to the specification, the number of clocks is fixed for any input sequence. Therefore the transition of states entirely depends on the number of clocks.

a. 30 FPS mode

Each output in the scanlines is determined by a 16 bits pixel value. To reduce the number of the output registers, we split the original 65536 clocks into 5 state. The number of clocks in each state is listed in the table below. The number of 1's in the outputs of each state is therefore determined by the correponsding 4 bits of the pixel value as shown in the Figure 3. Besides, before every state, the values of all pixels must be prepared. As a result we need additional preload registers for each pixel. These additional register is inevitable for either 4-bit or 16-bit register. Therefore, instead of 16*2 bits, only 4*2 bits is required for each output.

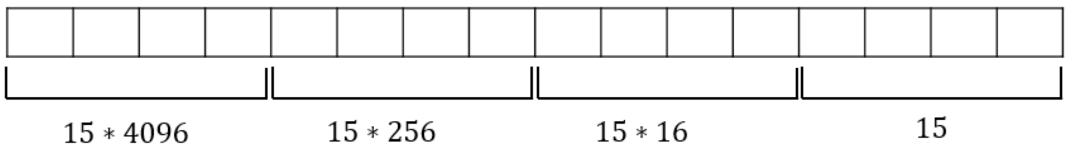


Figure 3. 4-bit data required in each state in 60 FPS mode

State	Clocks	Accumulated Clocks
Scan15	15	15
Scan255	240	255
Scan4095	3840	4095
ScanExtra	1	4096
Scan65535	61440	65536

Table 1. the PWM clock distribution in each state(30FPS)

b. 60 FPS mode

In the 60 FPS mode, the concept of our design is similar to that in the 30 FPS mode. However, in 60 FPS mode, when the pixel value is odd, there will be an additional bit in the first round compared to the second round. Hence, we store the data slightly differently in the 60 FPS mode by rotating the data to right. As the result, The first bit is an indicator of whether the value is odd and this indicator will be used to determine the output in the ScanExtra state, as shown in the Figure 5.

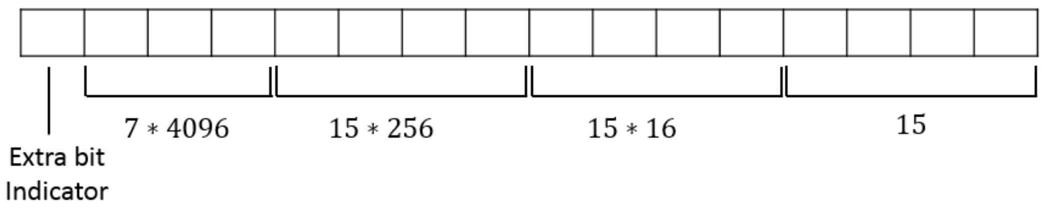


Figure 4. 4-bit data required in each state in 60 FPS mode

State	Clocks	Accumulated Clocks
Scan15	15	15
Scan255	240	255
Scan4095	3840	4095
ScanExtra	1	4096
Scan32767	28672	32768

Table 2. the PWM clock distribution in each state(60FPS)

3. Sram Control:

   In the 60FPS mode, the data have to be read from the sram again in the second round, which means all data of the frame have to kept intact in the sram until the second round begins. Therefore a 512-word sram is insufficient because the data of next frame is read in simultaneously.

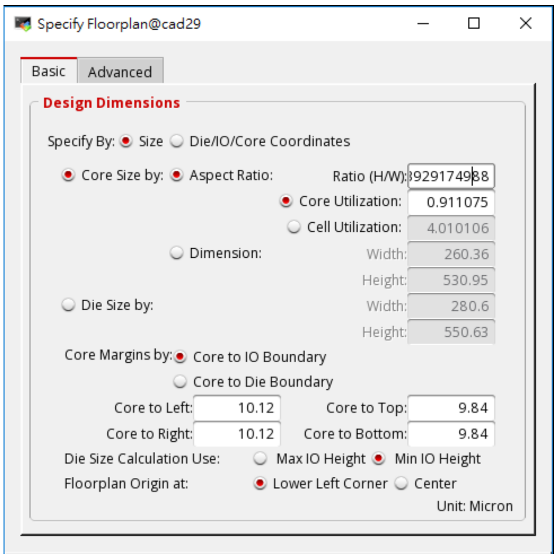
   To address this issue, we combine a 512-word sram and a 256-word sram into a 768-word sram. The address of sram is 0-767.

   To decode the address, since when the MSB is 1, the address is larger than 511 , we can easily use its MSB as the chip enable, and the real address to the sram is the remaining bits.

C. APR Details & Results

Most of the settings and APR flow follow the ones described in Lab7 and Lab8. We focus on planning the position of macros (e.g. SRAMs) and techniques for fixing DRC errors.

4. Floorplan Design
   a. Specify floorplan & SRAM locating



Specify Floorplan@cad29

Basic | Advanced

Design Dimensions

Specify By: [✓ Size] Die/IO/Core Coordinates

[✓ Core Size by:] [✓ Aspect Ratio:] Ratio (H/W): }929174988

[✓ Core Utilization:] 0.911075

Cell Utilization: 4.010106

Dimension: Width: 260.36 Height: 530.95

Die Size by: Width: 280.6 Height: 550.63

Core Margins by: [✓ Core to IO Boundary] Core to Die Boundary

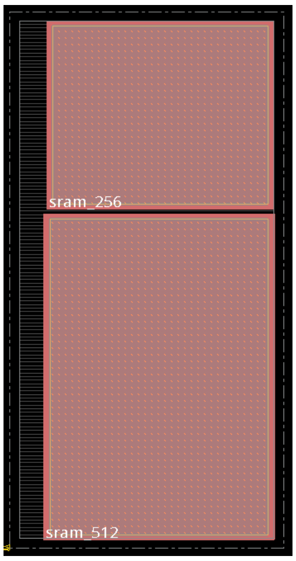
Core to Left: 10.12 Core to Top: 9.84

Core to Right: 10.12 Core to Bottom: 9.84

Die Size Calculation Use: Max IO Height [✓ Min IO Height]

Floorplan Origin at: [✓ Lower Left Corner] Center

Unit: Micron



b. Halo settings

{Top, Bottom, Left, Right} = {6um, 6um, 5um, 5um} for both sram_256 & sram_512

5. Power Design

a. Power ring

Most of the settings follow those in Lab7, except setting "Number of bits" to 1 in order to reduce the chip margin, and also the design is small so such power planning is enough.

b. Power stripe

Set the stripe at the middle of the design, and "Number of bits" is set to 1.

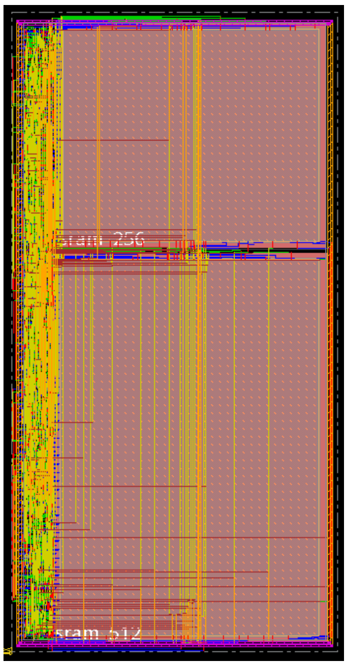
6. Routing & Fixing DRC errors

a. "Cut short error" is the most frequently encountered DRC error, which occurs when an via is directly set between SRAM pin (metal 3) and wire (metal 4). A reasonable work-around is to insert a "routing blockage" if metal 3 and metal 4 covering all SRAM pins before nano-route, and thus forces the EDA tool to avoid using metal 4 wire to gain access to SRAM.

b. We found that the floorplan design strongly affects the result of routing. The way we locate SRAMs and the halo we set will latter cause the routing to success or fail with lot of DRC errors.

c. As doing routing, we found that there sometimes exist hold time violation. However, such violation may not be activated referring to the functionality specified by the project. For example, some violated path is from mode pin to register, but the signal is stable during the usage of module. Another example is the path from DEN to registers, however the DEN is set before the clock triggered, and thus not influences the functionality correctness.

7. Layout Result


sram_256
sram_512