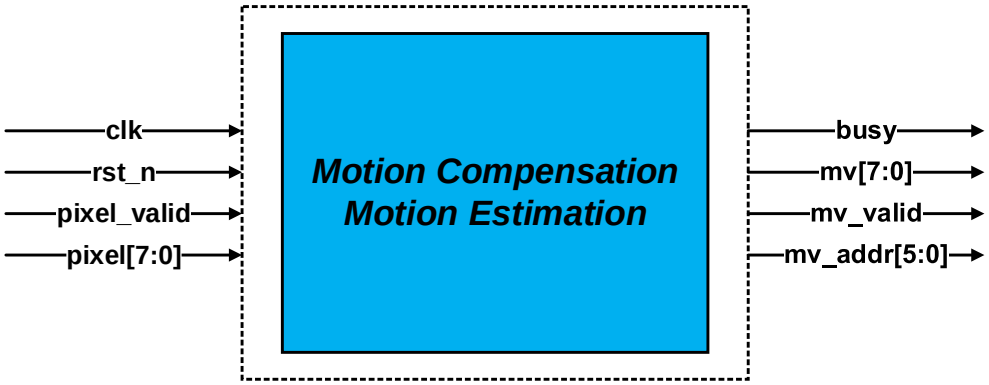
Computer-Aided VLSI System Design, Fall 2019

Motion Estimation/Compensation

1. Problem Discription

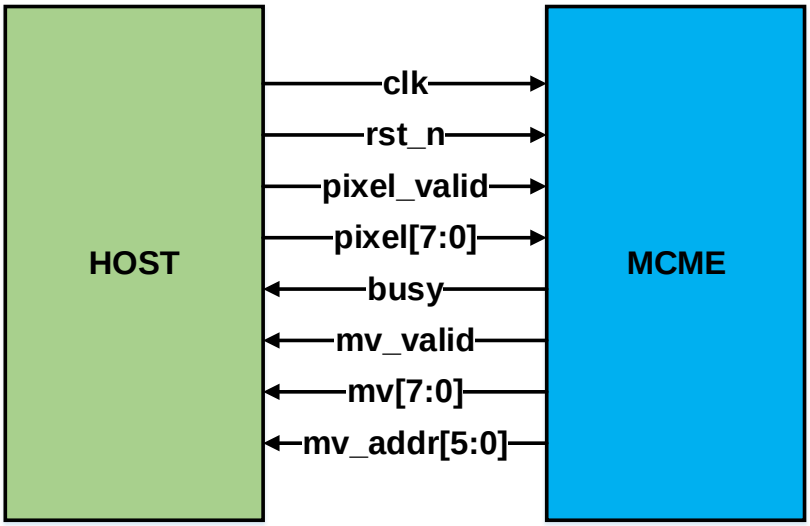
Please complete a circuit design of Motion Estimation / Compensation (hereinafter referred to as MEMC). This circuit can receive multiple 8bits Gray Level Image signals of 48x64 pixels (Pixels), and then output a Motion Vector (hereinafter referred to as mv) signal of the image according to the Block-Matching Motion Estimation algorithm, and Reach the specified PSNR. Detailed specifications for Motion Estimation will be described later. For the function description of each input and output signal of this circuit, please refer to Table 1. Each team must complete the design verification according to the design specifications given in the next section and the test samples in Appendix A.



圖一、MCME之方塊圖

2. Design Specification

2.1. System Block Diagram



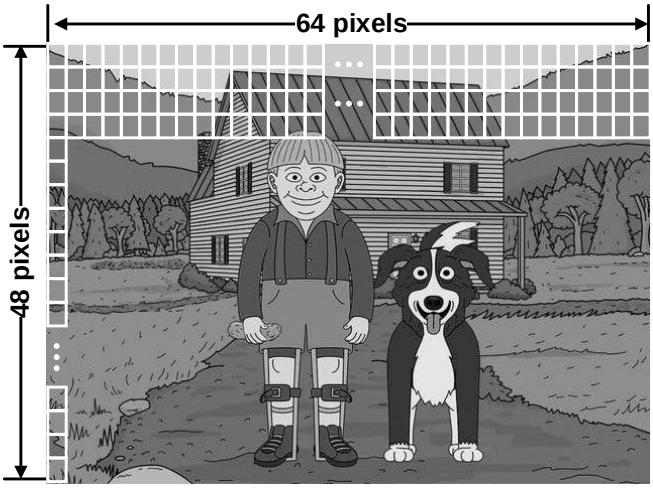
圖二、系統方塊圖

2.2. Input and Output Ports

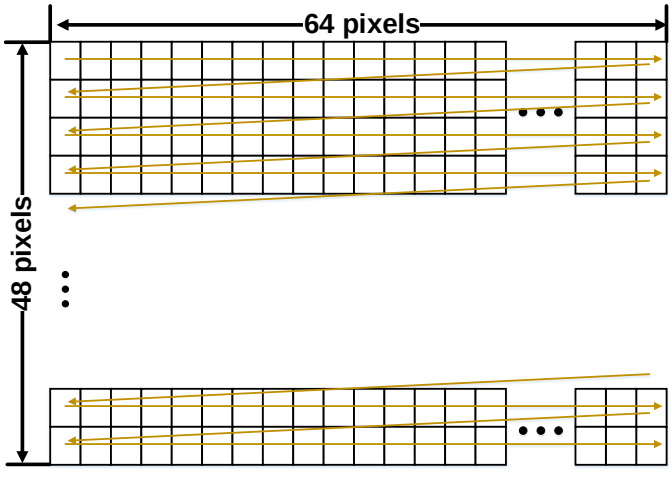
表一-輸入/輸出訊號

Signal Name	I/O	Width	Simple Description
clk	I	1	Design Clock, its default period is 10ns, which can be adjusted under the condition of timing rule.
rst_n	I	1	Reset Signal。(Active Low)
pixel_valid	I	1	Input pixel signal valid. When the signal is high, the pixel is a valid value.
pixel	I	8	The input grayscale pixel signal (8 bits) is used as the pixel signal input for the entire frame. The input order is from top left to bottom P (0, 0), P (0, 1), P (0, 2)… P (0, 63), P (1,0), P (1,1)… P (47, 63). For detailed input methods, please refer to 2.3 System Description.
busy	O	1	When the signal is high, the host will suspend updating the input pixel signal.
mv_valid	O	1	The output motion vector is a valid signal. When the signal is high, mv, mv_addr, and mv_frame are valid values.
mv	O	8	Motion Vector Signal. The signal is 8 bits, and the first 4 bits are the row coordinates of the vector. The next 4 bits are the column coordinates of the vector. **mv = {4'bRow, 4'bColumn}**。
mv_addr	O	6	This signal is the coordinates of the Motion Vector in the frame. There are 6 bits in total, the first 3 bits are row coordinates, and the last 3 bits are column coordinates.

2.3. System Discription



圖三、48x64 灰階影像範例



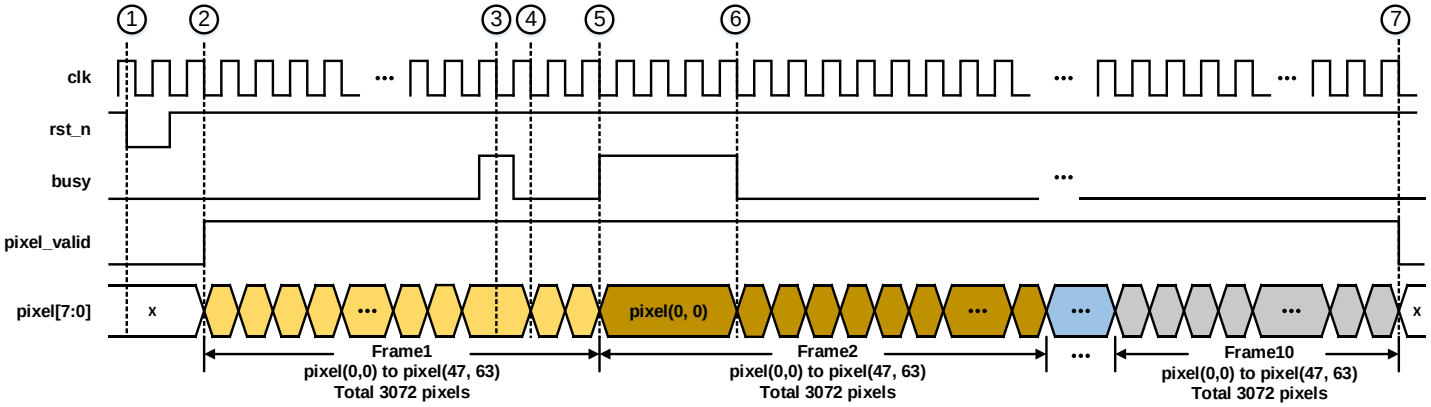
圖四、pixel輸入順序

Figure 3 is a 48x64 grayscale image. Such a picture is defined as a frame. A white box in the picture is

called a pixel, and each pixel is represented by 8 bits. When the MEMC is busy, the HOST will start from the first pixel (0,0) of the first frame, and continuously input pixels until the last pixel (47, 63) of the last frame. When the bus of MEMC is pulled up to high, the HOST will stop updating the input pixel. At this time, pixel_valid will remain high again and the pixel will remain unchanged. Until busy returns to low again, the pixel value will continue to update. Note that MEMC cannot read previously entered pixels to the HOST terminal.

At the output of the MEMC, the corresponding mv must be output in the order of the input frames, that is, the mv of the second frame cannot be output before the mv of the first frame is output. But the order of mv in the same frame is not specified, and each group can be controlled through mv_addr. The HSOT end will judge the mv and mv_addr as valid values when mv_valid is high, and output the frame per second (fps) of the frame and its PSNR after the mv of each frame is input. At the same time, considering the simulation time, only 10 frames need to be processed.

2.3.1. MEMC Input Waveform

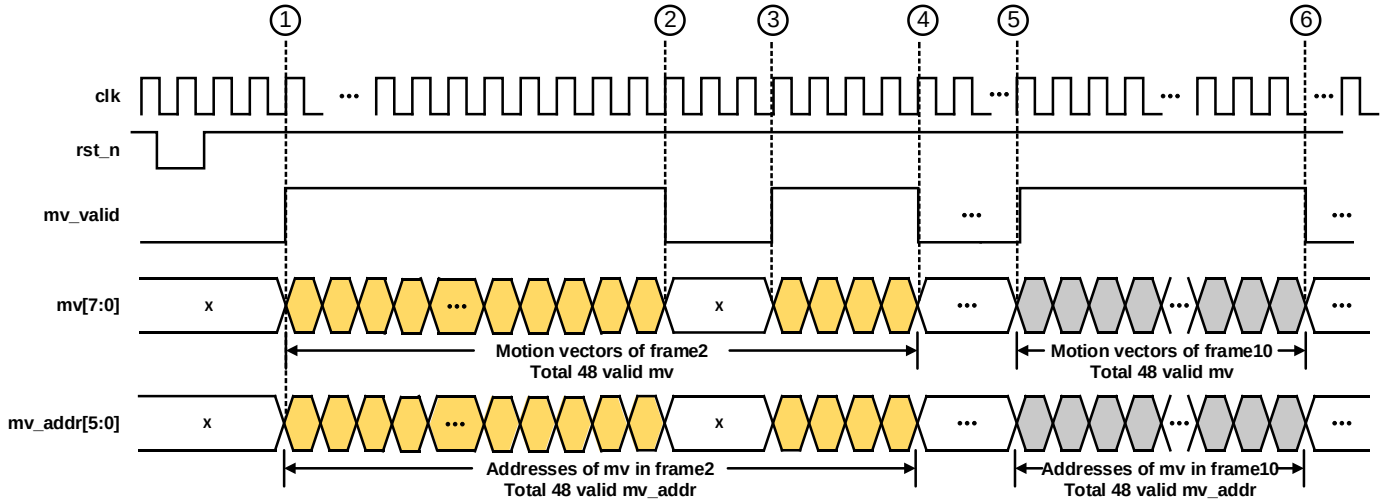


圖五、系統輸入時序圖

The description of HOST to MEMC input method is as follows:

1. Figure 5. When the system circuit simulates the time axis of 10ns, the HOST terminal will first give rst_n as low as about 1 ~ 3 cycles. If the rst_n signal is used, please use asynchronous reset to design the circuit.
2. Figure 5. After the reset signal returns to high, the negative edge system starts to judge the busy signal of MEMC after another period. If the busy signal is low, the HOST terminal starts to input a valid pixel value. Note that the HOST input is triggered with a negative edge.
3. Figure 5. When the busy signal received by HOST is high, the pixel_valid and pixel signals remain unchanged.
4. Figure 5. When the busy signal received by HOST is low, pixel_valid remains high and the pixel signal starts to update.
5. Figure 5. When HOST has finished outputting 3072 pixels, it will start to output the first pixel signal of the next frame. The busy signal is high, so the pixel signal remains unchanged.
6. Figure 5. When the busy signal is low, HOST starts to update the output pixel signal.
7. Figure 5. When the last pixel output of the last frame is successful, HOST stops outputting valid pixel signals. Note that the so-called output is successful, the output must be successful once the busy signal is low when HOST outputs a valid pixel value.

2.3.2. MEMC Output Waveform



圖六、MCME 輸出時序圖

MEMC outputs HOST as follows:

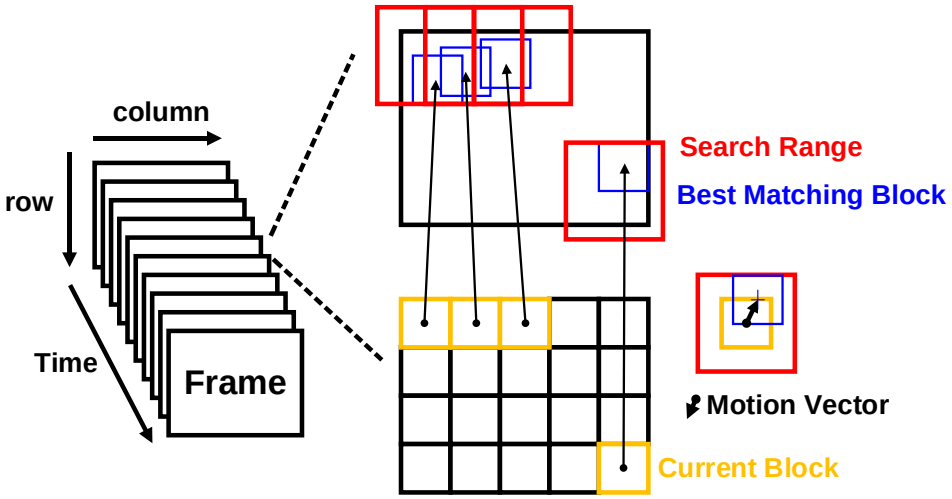
1. Figure 6. After reset, when mv_valid is high, it is determined to be valid mv and valid mv_addr.
2. Figure 6. When mv_valid output from MEMC is low, mv and mv_addr are invalid values, and no action will be taken on the HOST side.
3. Figure 6. When mv_valid returned by MEMC returns to high again, HOST will continue to read the valid mv and mv_addr.
4. Figure 6. MEMC completes the output of mv and mv_addr for a frame. HOST will have a counter to determine the number of mvs to output. Each frame has 48 mvs. When the 48 mvs are all output, the PSNR value of the frame is calculated.
5. Figure 6. MEMC starts to output mv and mv_addr for the next frame.
6. Figure 6. MEMC outputs all mv and mv_addr without 10 frames at all. The simulation ends.

2.3.3. Single-Port Memory

This question provides 256x8, 512x8, 4096x8, and 16384x8 single-port SRAMs for each team. Each team should consider what kind of memory to use. It can be used arbitrarily or not. For the timing diagram of the above memory read/write, please refer to the sram_256x8.pdf, sram_512x8.pdf, sram_4096x8.pdf, sram_16384x8.pdf files.

3. Motion Estimation/ Compensation Algorithm

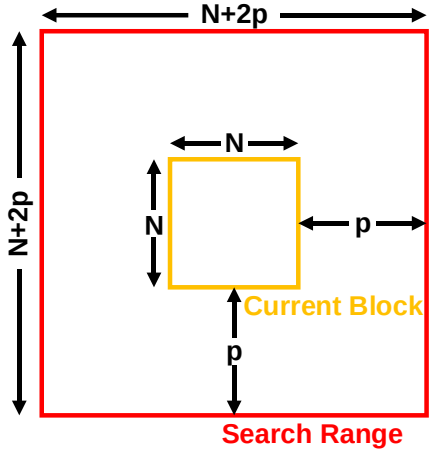
Motion Compensation (Motion Compensation) is a method to describe the difference between adjacent frames, specifically to describe that a block in the previous frame of image is moved to a certain position in the current frame of image. law. Motion Estimation is an algorithm that uses motion vectors to describe the conversion between two frames of images. It is mainly divided into Pixel-based and Block-based. This topic requires each team to implement the MEMC with a block-based algorithm.



圖七、MEMC 演算法

Figure 7 explains how the MEMC algorithm works.

1. Split the current frame image into several Current Block (CB)
2. Each CB must search for Best Matching Block (BMB) on the previous frame in the specified Search Range.
3. Use the difference between BMB and CB as the Motion Vector output. That is $mv(k,l) = BMB(i,j) - CB(i,j)$



圖八、影響Best Matching Block的參數

The indicator of Best Matching Block can be Sum of Square Pixel Difference (SSD) or Sum of Absolute Pixel Difference (SAD). In this question, the current block size is limited to 8x8, the search range is [+7, -7], and the total PSNR must reach 23 (this question uses grayscale images, so MAXI is 255).

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right)$$

In addition, each group of students can refer to the MEMC.pdf file in the folder, which contains a more detailed description of MEMC.

4. Scoring Criteria

The scoring method will be based on the design completion level, A, B, C, and D. The ranking order is A> B> C> D. The scoring items are Time-to-Area values. The ranking in the same level is determined based on the Time-to-Area value.

✧ **Scoring item: Based on "Time-to-Area" value**

✧ **Example:**

innovus > analyzeFloorplan

```
encounter 2> analyzeFloorplan
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 79981 sites (135760 um^2) / alloc_area 79981 sites (135760 um^2).
Pin Density = 0.122.
            = total # of pins 23994 / total Instance area 196550.
************************** Analyze Floorplan **************************
    Die Area(um^2)             : 333039.25
    Core Area(um^2)            : 314558.83
    Chip Density (Counting Std Cells and MACROs and IOs): 84.316%
    Core Density (Counting Std Cells and MACROs): 89.270%
    Average utilization        : 100.000%
    Number of instance(s)      : 14105
    Number of Macro(s)         : 2
    Number of IO Pin(s)        : 23
    Number of Power Domain(s) : 0
************************** Estimation Results **************************
************************************************************************
```

⇨ Area = 333039.25 um^2

Note: The instruction analyzeFloorplan will destroy the results of the completed routing, and it must never be archived after executing this instruction

Four levels of design completion are explained below:

✧ **Level A： Be sure to meet the following three requirements**

a. The two sets of pattern data have reached the PSNR specified in the title, that is, the RTL simulation results are correct.

b. Both sets of test data have reached the PSNR specified in the title and completed Synthesis, and the gate-level pre-layout simulation results are correct.

c. The two groups of funds have reached the PSNR specified in the title and completed the APR, and reached the necessary APR items. The Gate-Level Post-layout Simulation results are correct.。

Note: Necessary items to complete APR

i. Just do the Marco layout (i.e. don't include IO Pad, Bonding Pad).

ii. Please set the width of VDD and VSS Power Ring to 2um each, only one set is required.

iii. No need to add Dummy Metal.

iv. For all built-in memory SRAM, the VDD and VSS pins must be connected to the Core Power Ring. Please set the width to 2um.

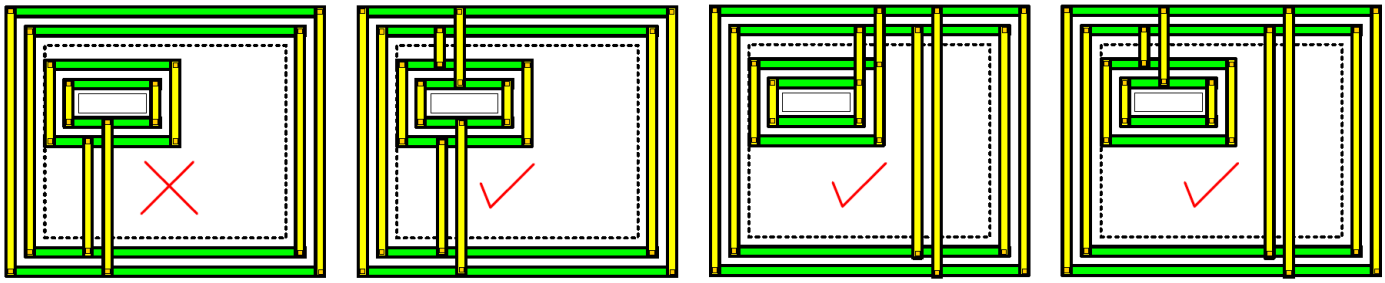
v. Be sure to add at least one Power Stripe, and set the VDD and VSS widths to 2um each. (At least one group of Power Stripe in the vertical direction, but not in the horizontal direction)

vi. Be sure to add Power Rail (follow pin).

vii. Core Filler must be added.

viii. Complete APR, DRC / LVS is completely correct (see description in Appendix C).

Note: Power Stripe refers to the power line running directly through the core area, see the figure below

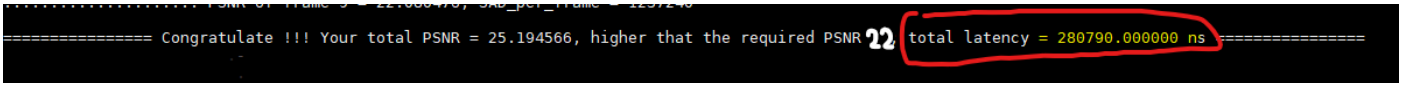


Level A scoring method：

Score = Time*Area

E.g:

This example (Innovus) is,



Score = Time *Area = 280790.0*333039.25

Note: The smaller the score, the better the ranking of the same level!

Level B: Must meet the following three requirements

a. The two sets of pattern data have reached the specified PSNR, that is, the RTL simulation results are correct.

b. Both sets of test data have reached the specified PSNR and completed Synthesis, and the gate-level pre-layout simulation results are correct.

c. Complete APR, but DRC / LVS has some errors.

Level B scoring method：

The smaller the total number of DRC / LVS errors, the better, the same is better than Score

Score = Time*Area

Note: Area is based on the Cell Area of Design Compiler, and Time is based on the Latency of Gate-Level Simulation. The smaller the Score, the better the ranking of the same level!

✧ **Level C： Be sure to meet the following two requirements**

a. The two sets of pattern data have reached the PSNR specified in the title, that is, the RTL simulation results are correct.

b. Both sets of test data have reached the PSNR specified in the title and completed Synthesis, and the gate-level pre-layout simulation results are correct.

Level C scoring method：

Score = Time*Area

Note: Time is mainly based on Latency of Gate-Level Simulation; Area is mainly based on cell area after Synthesis. The smaller the score, the better the rank of the same level!

✧ **Level D：Be sure to meet the following requirement**

a. The two sets of pattern data have reached the PSNR specified in the title, that is, the RTL simulation results are correct.

Score = Time

Note: Level D is mainly based on RTL Simulation Latency. The smaller the score, the better the ranking of the same level.

Appendix

Appendix A is a description of the design files provided by each student; Appendix B is a description of the test samples provided; Appendix C is a design verification description; Appendix D is a scoring file, that is, a scoring file that students must submit; Instructions for compressing and collating the design files; Appendix F describes the software environment of this competition; Appendix G describes the design database used in this competition.

Appendix A - Design File (For Verilog)

1. The table below is the design files provided by the students.

Table 3.Design file description

File name	Description
01_RTL/MEMC.v	The design file of this question already includes the declaration of the system Input / Output.
01_RTL/tb1.v 01_RTL/tb2.v	There are two TestBench in this question, calling two sets of patterns respectively.
Patterns/**pattern_MrPickles_gray_4s_sr5.hex** Patterns/**pattern_RickandMorty_gray_2s_sr5.hex**	It is used as the input signal of frames in the MEMC circuit simulation. Note: This file has been added to TestBench.
SRAM/sram_OOOx8.ooo	Memory related files, each size folder has the following files sram_OOOx8.pdf sram_OOOx8.vclef sram_OOOx8.vp (this is an encrypted behavior model) sram_OOOx8.ant.lef sram_OOOx8_slow_syn.db sram_OOOx8_slow_syn.lib
02_SYN/.synopsys_dc.setup	Use Design Compile for synthesis or IC Compiler Layout initialization profile. Participants should modify the search path setting according to the actual placement of Library Note: Regardless of synthesis or APR, just use the worst case library.
02_SYN/MEMC_DC.sdc	Constraint file created by Design Compiler. This file is for reference only. Students can adjust Constraints according to the actual situation of the circuit. Note: Do not change environment-related parameters.
03_GATE/ tsmc13.v	For Gate-level simulation
~~04_APR/MEMC_APR.sdc~~	APR use .sdc Please modify the .sdc file generated after synthesis (After the APR course will teach)

Please use MEMC.v to design the MEMC circuit. Its module name and IO declaration is as follows:

```
module MEMC(
        clk,
        rst_n,
        pixel_valid,
        pixel,
        busy,
        mv_valid,
        mv,
        mv_addr
);

input clk;
input rst_n;
input pixel_valid;
input [7:0] pixel;
output busy;
output mv_valid;
output [7:0] mv;
output [5:0] mv_addr;

endmodule
```

2. There are two Patterns in this question. The TA will not use hidden Patterns when correcting.

3. The testBench file mentioned in this question has a few more lines of special purpose description as follows:

```
`define SDFFILE     "MEMC_syn.sdf"
`ifdef SDF
      initial $sdf_annotate(`SDFFILE, u_MEMC);
 `endif
```

Note:

1. SDF file, please modify the SDF file name and path before simulation
2. In the Test Bench, the TA provides the description of `ifdef SDF, so that this Test Bench can be used as RTL simulation, post-synthesis simulation, and post-layout simulation. Note: When students simulate after compositing or layout, be sure to add one more parameter "+ define + SDF" for successful simulation.

For example: After synthesis, use NC-Verilog simulation and execute the following instructions under UNIX

> ncverilog **+ncmaxdelays** tb1.v MEMC_syn.v sram_512x8.vp tsmc13.v **+define+SDF** +access+r

Appendix B - Test samples

Pixel signals for two sets of ten frames have been stored in the patterns folder. Each pattern has more than 10x3072 pixels, but each group only needs to use the first ten frames.

Note: The left side of the data is hexadecimal. Note: The left side of the data is hexadecimal.

Appendix C - Design Verification Instructions

Each team should complete three stages of verification: RTL, Gate-Level and Physical before submitting materials to ensure the correctness of the design.

- RTL and Gate-Level stages: Participants must perform RTL simulation and Gate-Level simulation. The simulation results must meet the period specified in this question, and the functions are completely correct.
- Physical phase, including three key verification points:
  1. Achieve complete and correct layout according to the requirements of the topic (please refer to the grading standards for detailed requirements).

  2. Complete post-layout simulation: teams must use netlist and sdf files written by APR software to complete post-layout gate-level simulation. The following Innovus software explains the steps for writing netlist and sdf.

i. For Cadence Innovus, perform the following steps:

Click in the Innovus window:

“ File → Save → Netlist … ”

Netlist File	MEMC_pr.v
All other options	Default value

press OK。

“ Timing → Extract RC… ” ，press OK。

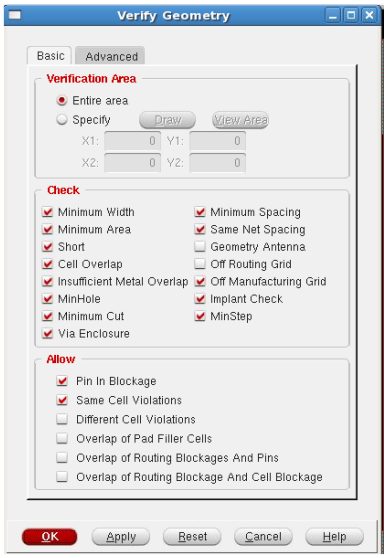
“ Timing → Write SDF… ”

Ideal Clock	Disable
SDF Output File:	MEMC_pr.sdf

press OK。

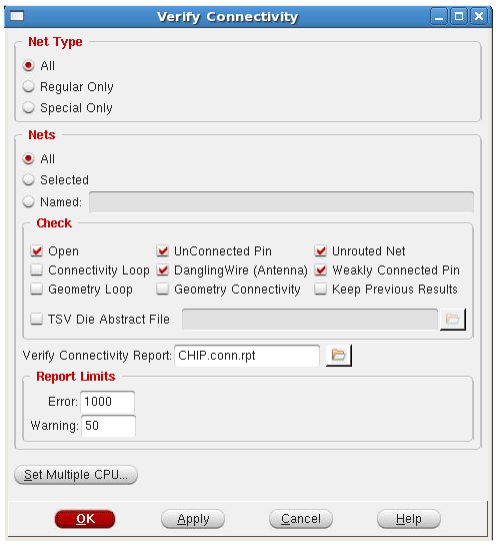
ii. For Cadence Innovus, verify the DRC and LVS steps as follows:

Click in the Innovus window DRC verification：select **“ Verify → Verify Geometry … ”** Default，press OK 。



Note: If there is an error in the DRC, please select “Tool → Violation Browser…” to find out why

1. LVS verification: Please select “Verify → Verify Connectivity…” Default value, press OK。



Note: If there is an error in the LVS, please select “Tool → Violation Browser…” to find out why

Appendix D - Scoring Files

The files required for scoring can be divided into the following parts: (1) RTL design, which is the RTL code for each participating team's design for this competition. If the design is modular and there are multiple design files, be sure to use the synthesis Put in the module files to avoid simulation when reviewing and scoring; (2) Gate-Level design, which is the gate-level netlist generated by the synthesis software, and the corresponding SDF file; (3) Physical design, use For Cadence Innovus, please compress the Innovus-related design database (including the .enc file and the .enc.dat directory, which simply means that the TA can open the Innovus interface to reproduce the files of each set of designs) into a single file. The compressed file format is as follows: Assuming that the contestant's design database directory name is "teamID_lib", please execute the UNIX instructions below, and finally you will get the "teamID_lib.tar" file. Note: Please use two digits for the ID number part. ex. If the first group is team01

- tar cvf teamID_lib.tar teamID_lib

ex: tar cvf team01_lib.tar team01_lib

Before executing the above instructions, please make sure to save and close the APR Tool you use, and then execute the above instructions, otherwise an error will occur during the compression process.

Table 4

RTL category		
Design Stage	*File*	*Description*
N/A	`N/A`	Design Report Form
RTL Simulation	`*.v or *.vhd`	Verilog (or VHDL) synthesizable RTL code
Gate-Level category		
Design Stage	*File*	*Description*
Pre-layout Gate-level Simulation	`*_syn.v`	Verilog gate-level netlist generated by Synopsys Design Compiler
	`*_syn.sdf`	Pre-layout gate-level sdf
Physical category		
Design Stage	*File*	*Description*
P&R	`*.tar`	archive of the design database directory
	`*.gds`	GDSII layout
	`DRC/LVS report`	Screenshot DRC / LVS Report file! Fill in the total number of DRC / LVS errors on the Design Report Form. (The goal is to make 0 errors!)
Post-layout Gate-level Simulation	`*_pr.v`	Verilog gate-level netlist generated by Cadence Encounter or Synopsys IC Compiler
	`*_pr.sdf`	Post-layout gate-level sdf

Appendix E - File Arrangement Steps

When all the documents are prepared as listed in Table 4, please follow the instructions below to submit the relevant design files and copy all the files to the same folder:

1. Create a new directory， **< teamID_vk>　　e x** ：

   > ***mkdir　team01_v1***

2. Copy the files required by Appendix D to the <teamID_vk> directory. E.g:

   > ***cp　MEMC.v　　teamID_v1***

   > ***cp　MEMC_syn.v　 teamID_v1***

   … …

3. Fill in the required information in the In Design Report Form.
4. **Upload < teamID_vk> to the /final_project/ directory of following FTP。**

FTP IP	140.112.175.174
Account	1081CVSD_student
Password	ilovecvsd

Important deadlines for the end of the topic

- **01/12 (Sun)　before 23:59　Final Project Submission Deadline**
- **01/16 (Thur)　14:20-17:20　Project Presentation**