Lab 7
Design of Local Binary Pattern Facial Recognition System

Instructor: Lih-Yih Chiou

TA: Leo

Date: 04/17/2024

EE

LPHPLAB
VLSI Design LAB

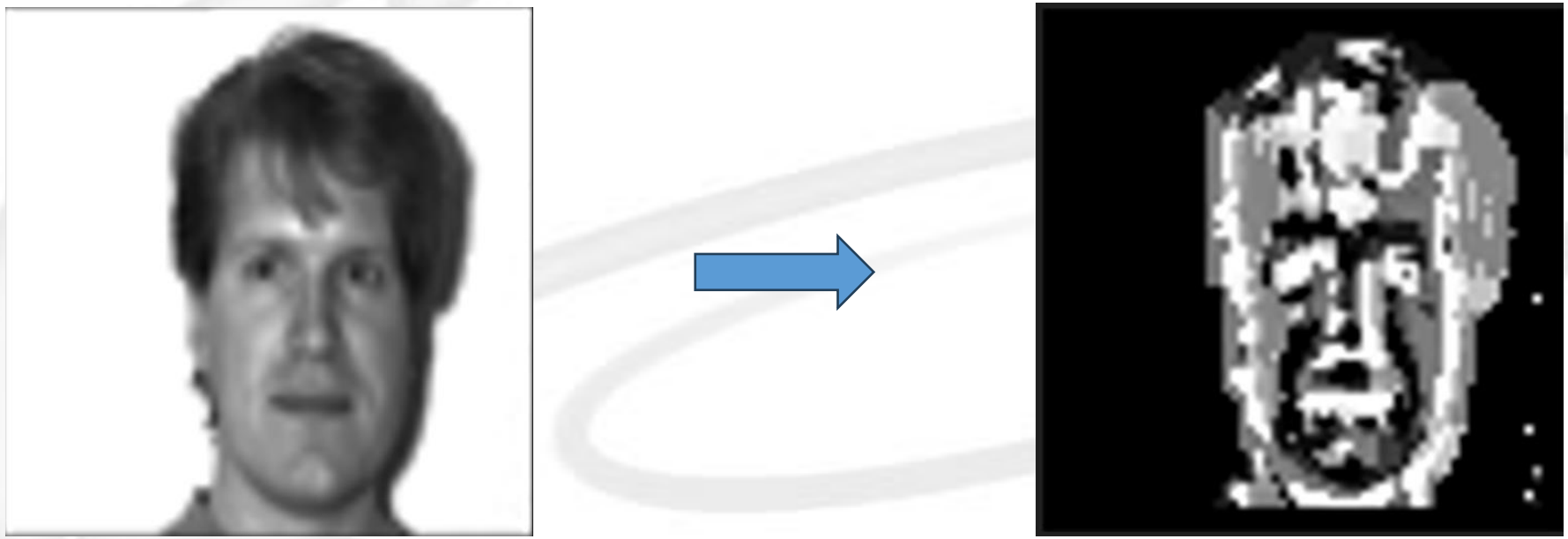
Outline

- LBP
- Histogram
- Recognition
- Architecture
- Data Mapping in Memory
- I/O Definition
- Waveform
- Simulation
- Grading Policy

LBP

LBP

- Local binary pattern (LBP) is a unique, efficient textural operator that are suitable for facial recognition tasks.



LBP – Circular Local Binary Pattern

1. Choose a pixel(center) in the image and select its neighboring pixels
2. Calculate bilinear interpolation
   1. Determine r_x & r_y
   2. Determine x_1, x_2, y_1, y_2
   3. Determine t_x, t_y
   4. Determine w_1, w_2, w_3, w_4
   5. Determine $f(0,0), f(0,1), f(1,0), f(1,1)$
   6. Determine neighbor → Rounding to nearest integer
3. Repeat 2 to calculate all neighbors' values
4. Construct the mask using threshold function
5. Combine the binary values for all neighboring pixels to obtain a binary code for the central pixel and convert it to a decimal value
6. Repeat steps 1–5 for each pixel in the image to obtain a binary code for each pixel

LBP – Circular Local Binary Pattern

- Threshold function

$$LBP(x,y) = \sum_{k=0}^{P-1} s\,(g_k - g_c)2^k$$

$$s = \text{Threshold}(g_k)\begin{cases} g_k \geq g_c, 1 \\ otherwise, 0 \end{cases}$$

$$where, g_k, k \in 0,1,2,3,4,5,6,7(P-1)$$

Pay attention to the inequalities!

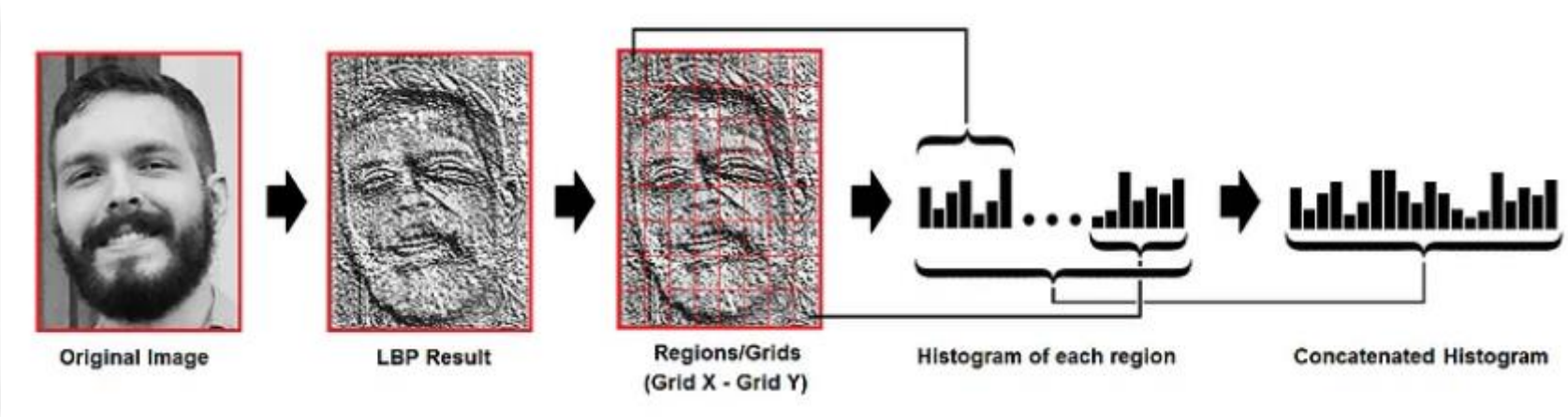
LBP – Circular Local Binary Pattern

- In this lab, *neighbor* needs to be rounded to the nearest integer
  - $neighbor = f(0,0)\mathrm{w}_1 + f(0,1)\mathrm{w}_2 + f(1,0)\mathrm{w}_3 + f(1,1)\mathrm{w}_4$
    - 246.979 -> 247.000
    - 48.580 -> 49.000
    - 215.013 -> 215.000

Histogram

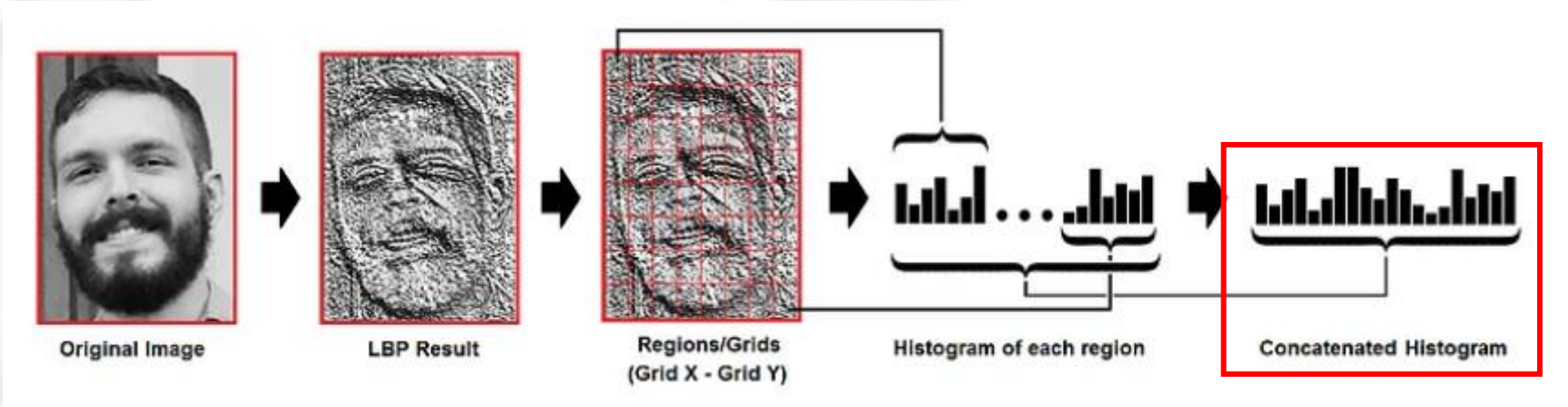
Histogram

- This step happens after LBP computation is done
- Histogram construction
  - ➔ Use *GridX* and *GridY* to divide the LBP image into multiple grids
- LBP result is in grayscale, each histogram(from each grid) will contain only 256 positions(0-255) representing the **occurrences** of each pixel intensity



Histogram

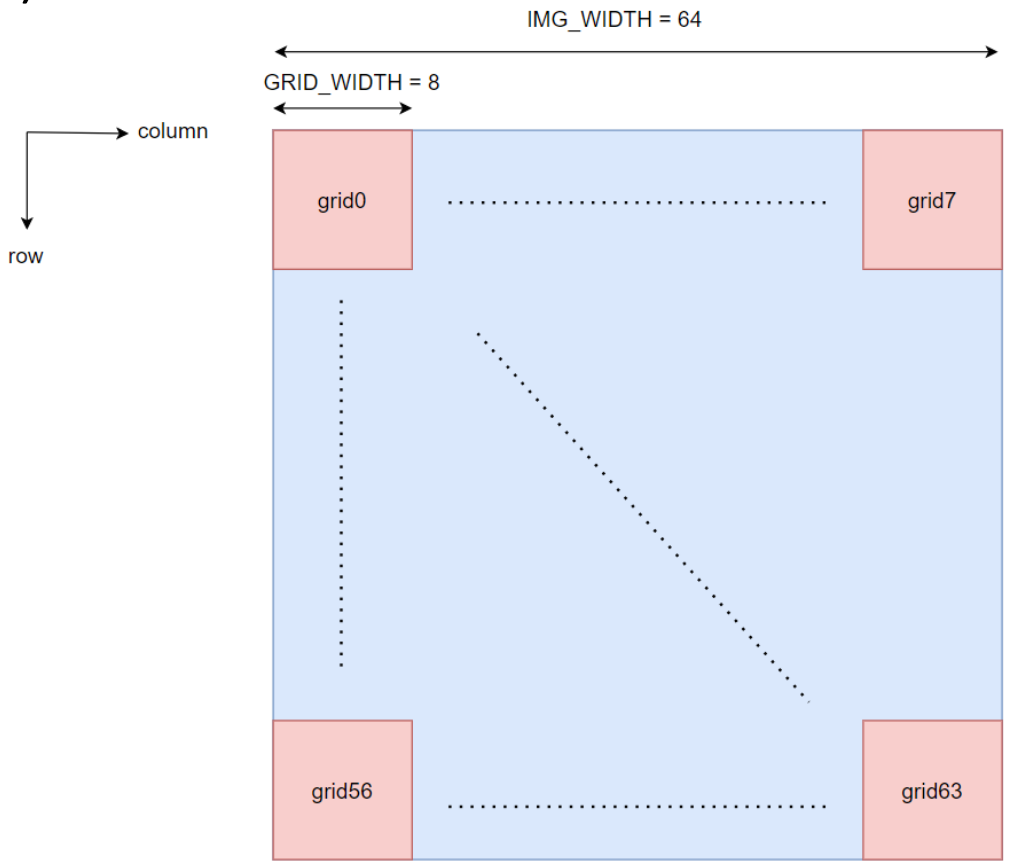
- Concatenate each histogram to create a new and bigger histogram
- Suppose we have 8x8 grids, we'll have 8x8x256=16384 positions in the final histogram
- The final histogram represents the characteristics of the original image
  - ➔ **Feature vector** of size 16384



Feature vector = histogram

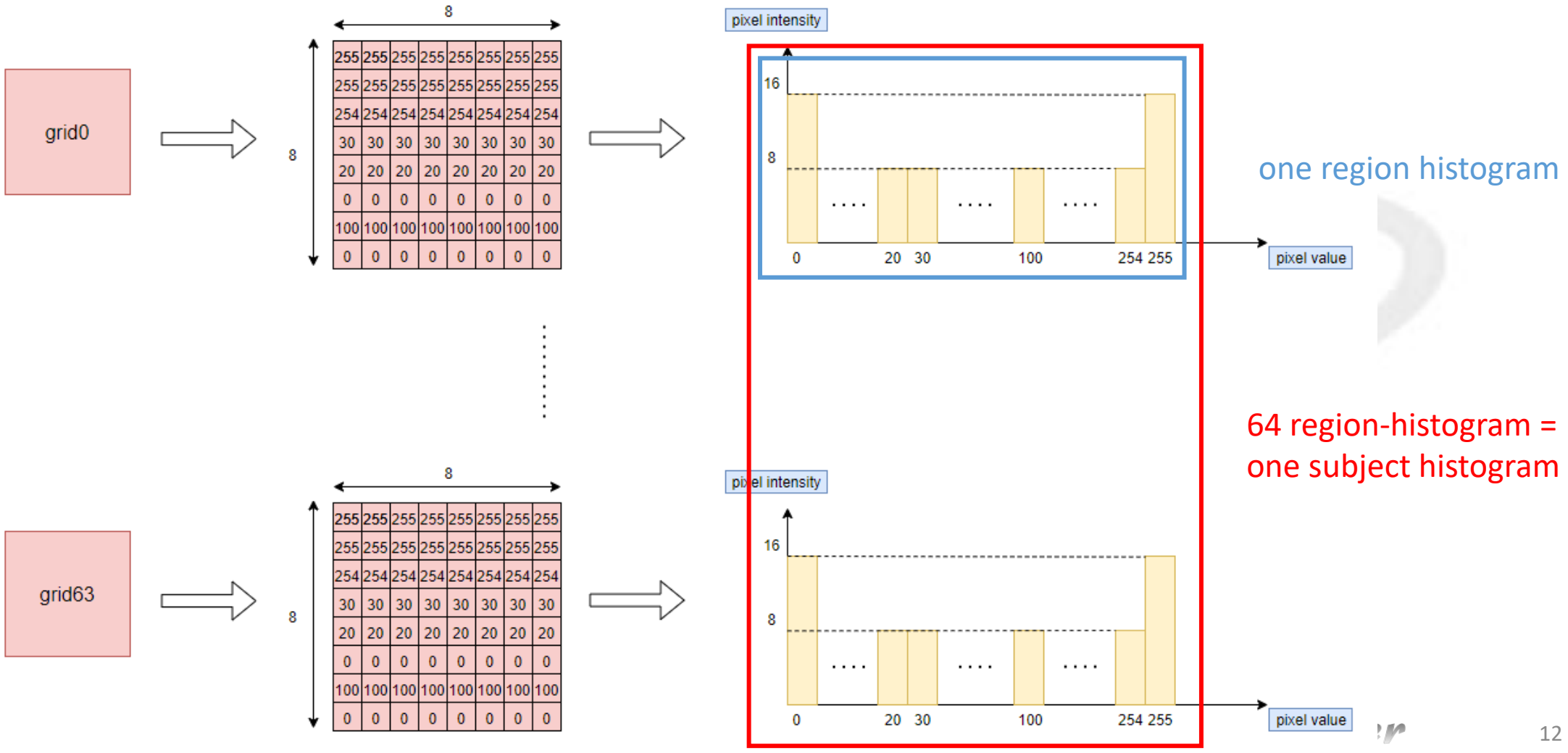
Histogram

- Image size of 64 x 64 is cropped into 64 equal girds
  - Each row has 64 columns, totally 64 rows
  - Each grid is of size 8(row) x 8(col)



Histogram

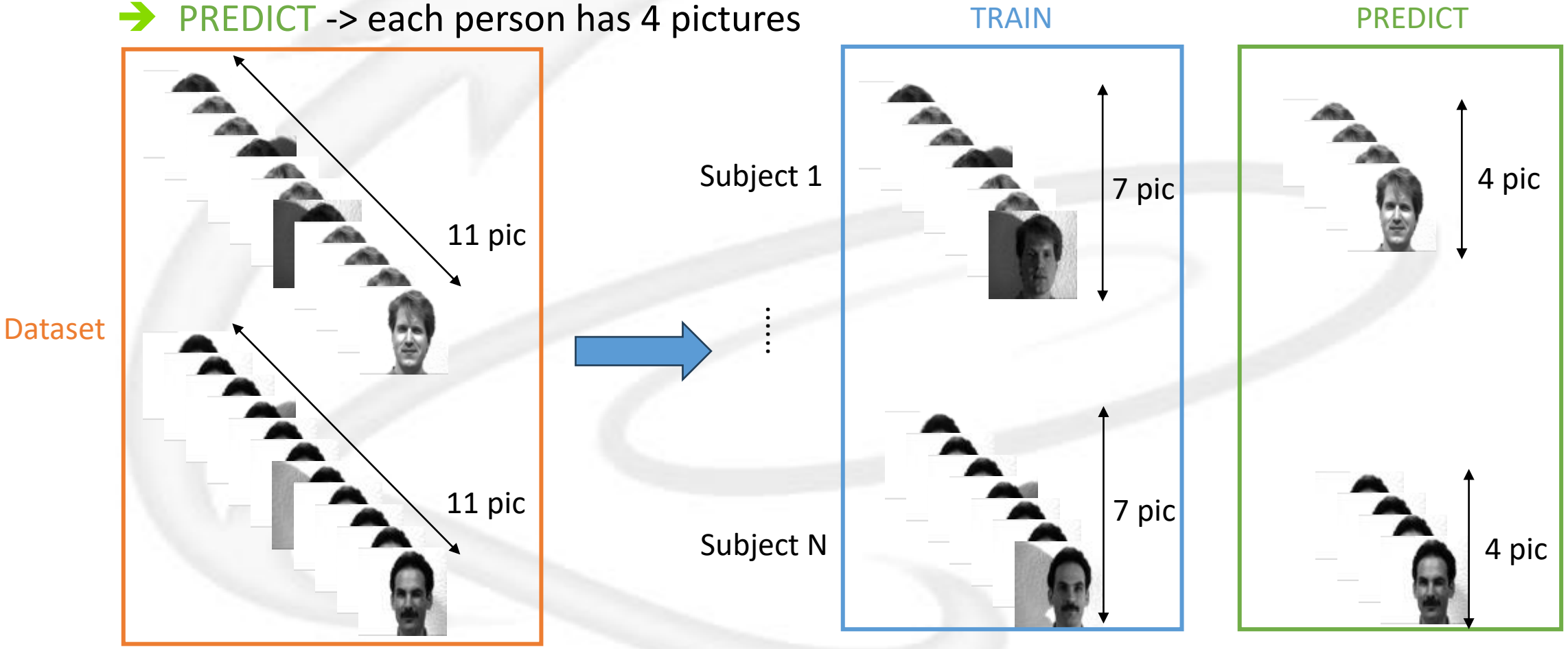
- Image grid converts to region-histogram



Recognition

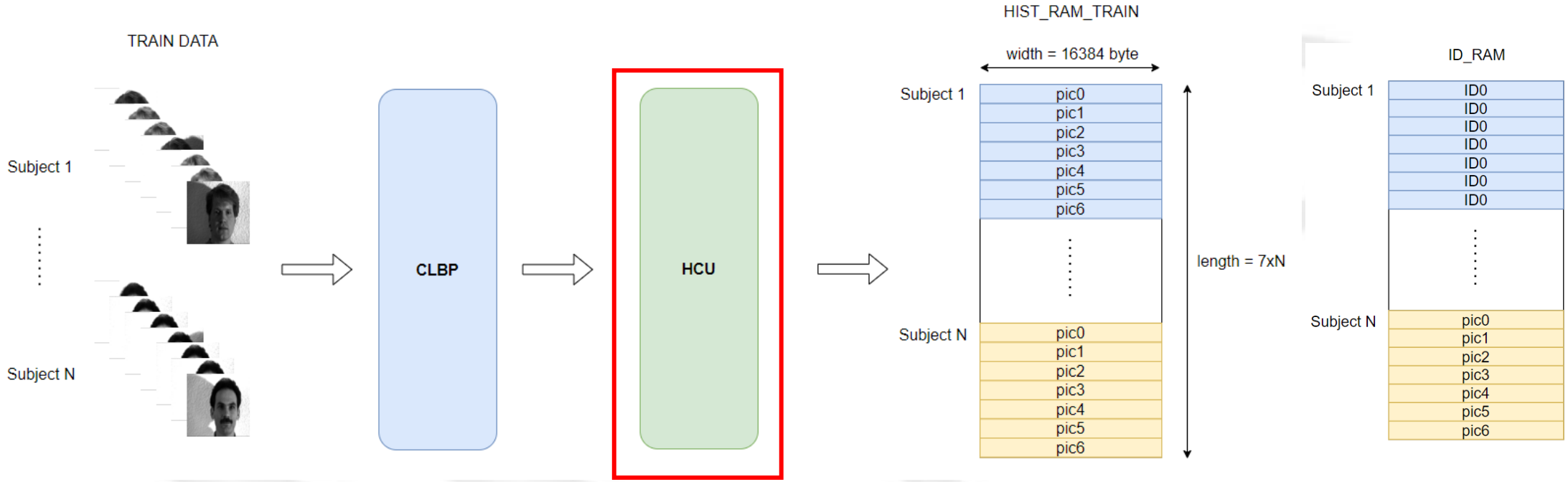
Recognition

- Dataset is divided into two parts, totally 5 persons
  - TRAIN -> each person has 7 pictures
  - PREDICT -> each person has 4 pictures



Recognition – mode TRAIN

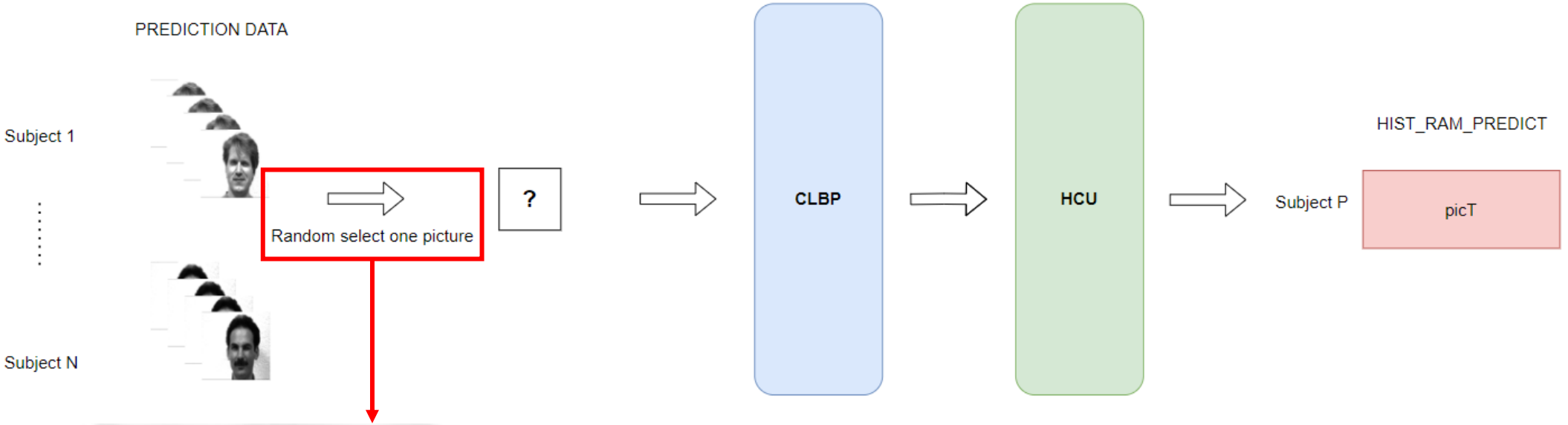
- For each image in the training dataset, we construct its corresponding histogram, and store it to the histogram training database with tagged ID
  - HIST_RAM_TRAIN
  - ID_RAM



Histogram Construction Unit

Recognition – mode PREDICT

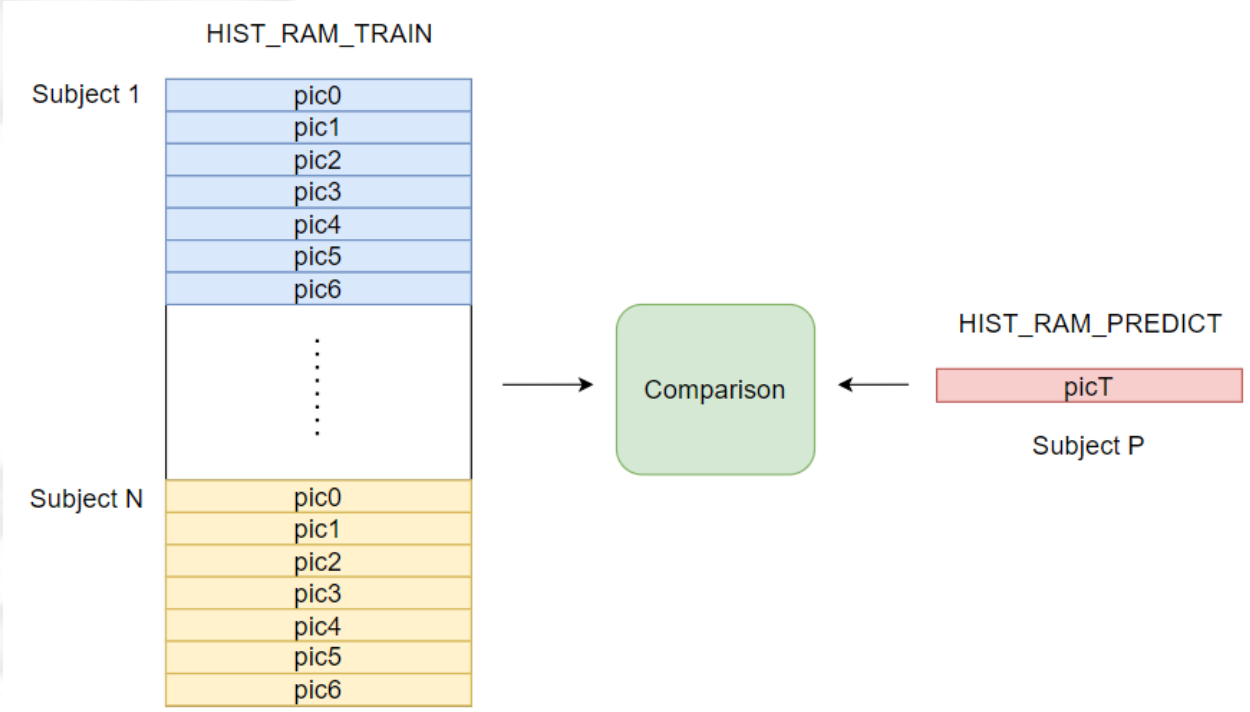
- For the given prediction image in the prediction dataset, we construct its histogram, and store it to the histogram prediction database
  - ➔ HIST_RAM_PREDICT



Randomly selects subject P with picture T

Recognition – mode PREDICT

- With the histogram of prediction image being established, we find the closest histogram compared to those in training database



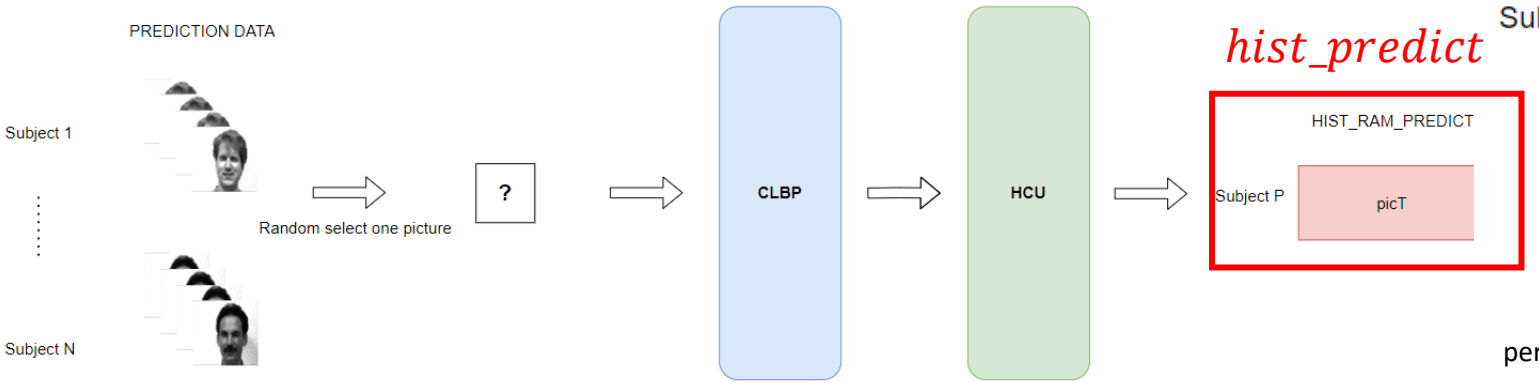
Recognition – mode PREDICT

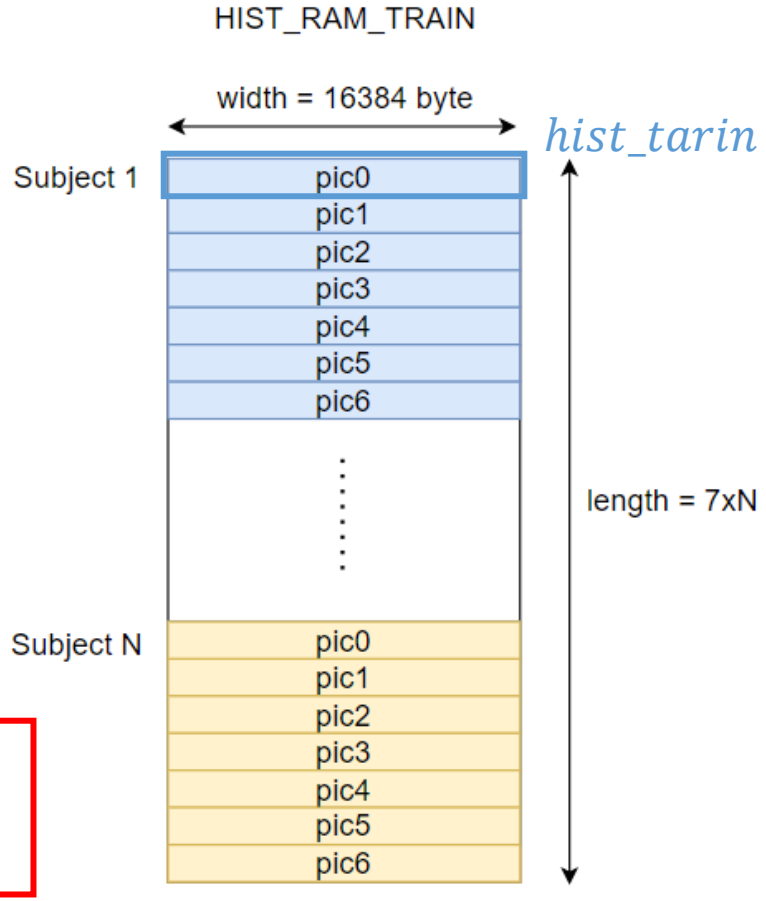
- Compare histograms using distance calculation
  - ➔ Euclidean distance

- Suppose we have *hist_predict* and *hist_train*, the equation to compute their distance *D*
  - ➔ $D = \sum_{p=1}^{n} (hist\_predict_p - hist\_train_p)^2$
    - Where *hist_predict* is the prediction histogram, and *hist_train* is the histogram in training database
    - Where *n* = 8x8x256= 16384

Recognition – mode PREDICT

- After training database is established, system starts to predict
- Suppose we have *hist_predict* from subject P
- Pseudo code
  - *initialize minDistance*
  - *for i = 0 to* length(HIST_RAM_TRAIN)
    - *hist_tarin* = HIST_RAM_TRAINi
    - *distance* = *compute D*(*hist_predict*, *hist_train*)
    - *if distance < minDistance*
      - *update minDistance*
      - *record ID*

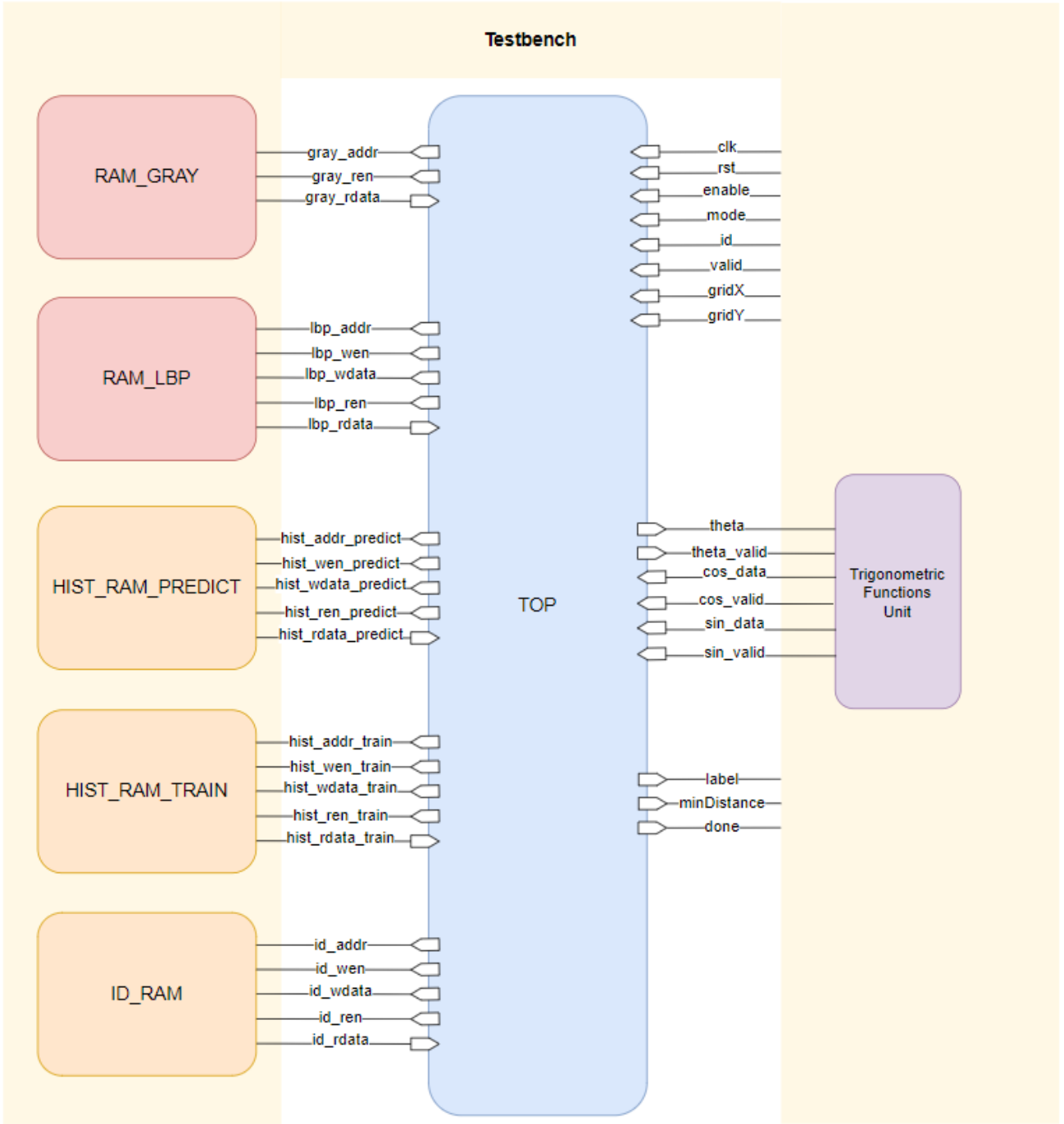




permission

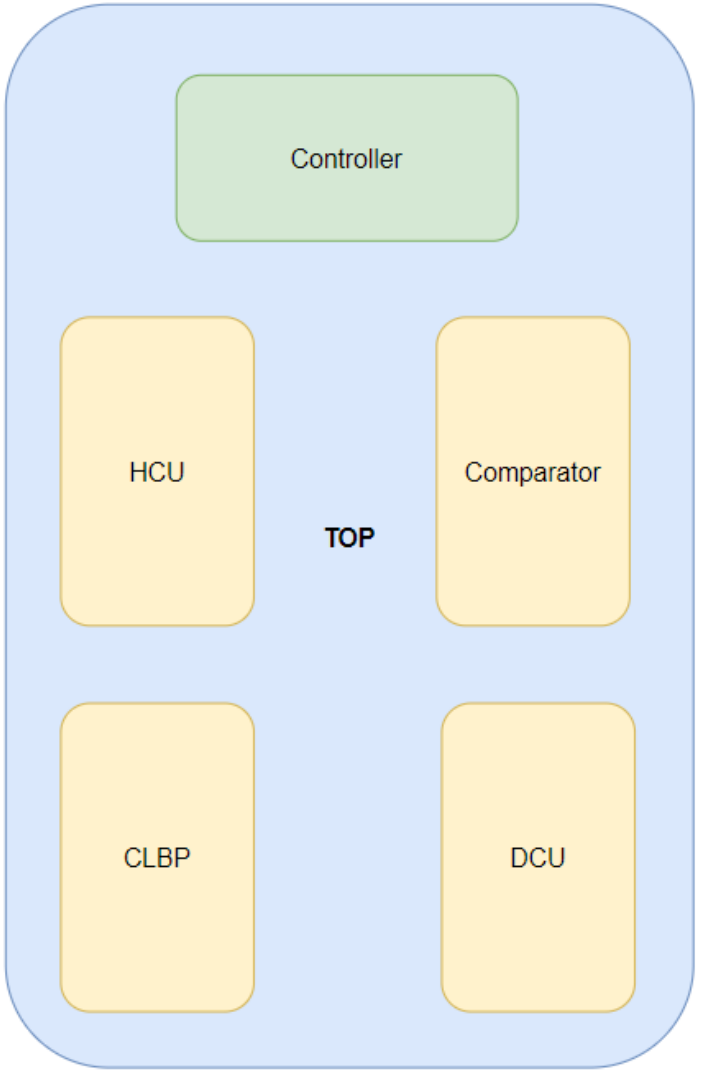
Architecture

Architecture



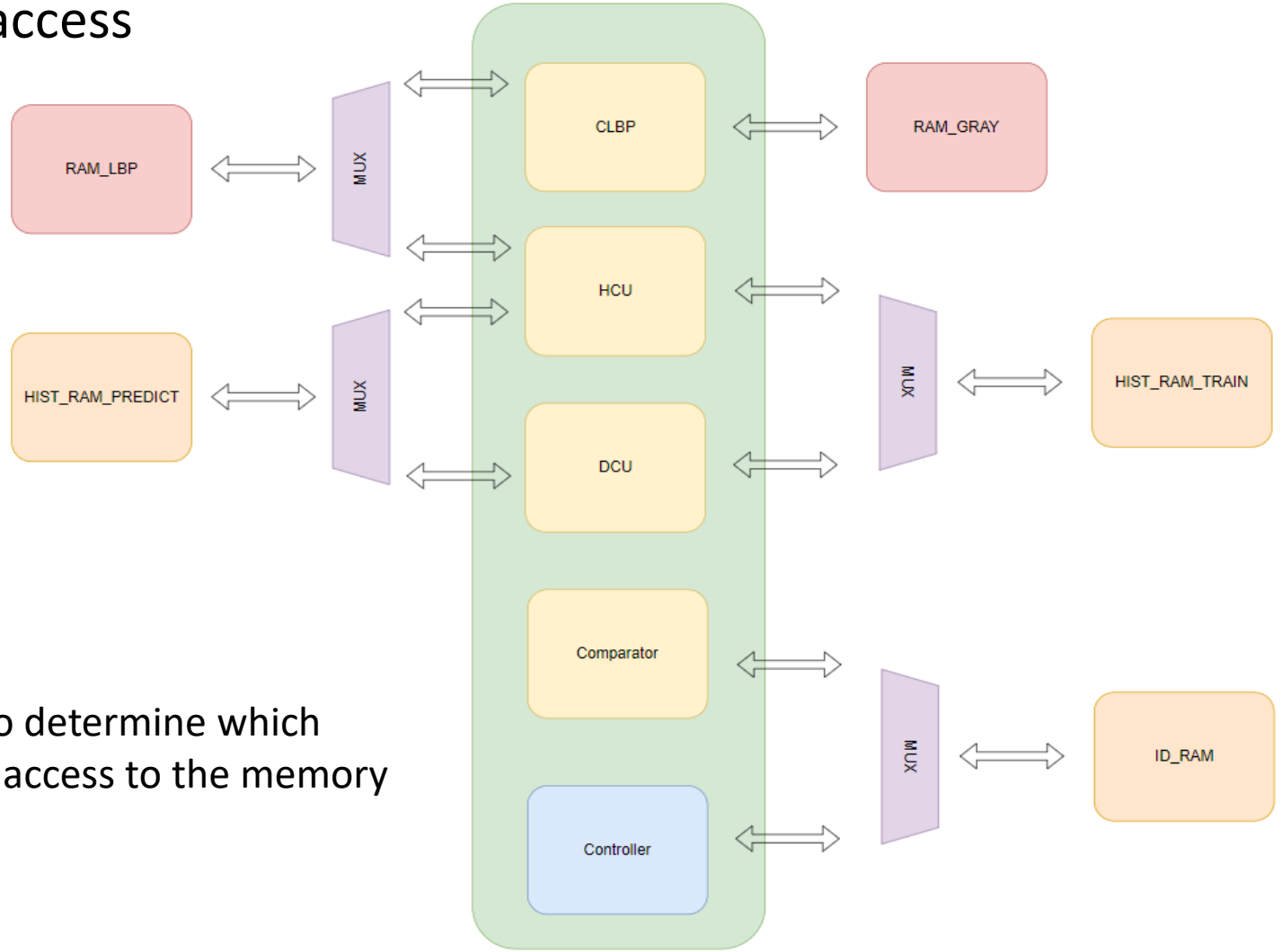
Architecture

- Submodules in TOP
  - CLBP (Circular Local Binary Pattern)
    - Compute the LBP value of the current subject
  - HCU (Histogram Construction Unit)
    - Compute the histogram of the current subject using CLBP result
  - DCU (Distance Compute Unit)
    - Compute the distance between *hist_predict* and *hist_train*, see Recognition pseudo code
  - Comparator
    - Controls DCU circuit
    - Using DCU computed value to determine the prediction *label* and *minDistance* value, see I/O Definition - TOP
  - Controller
    - Controls each submodule



Architecture

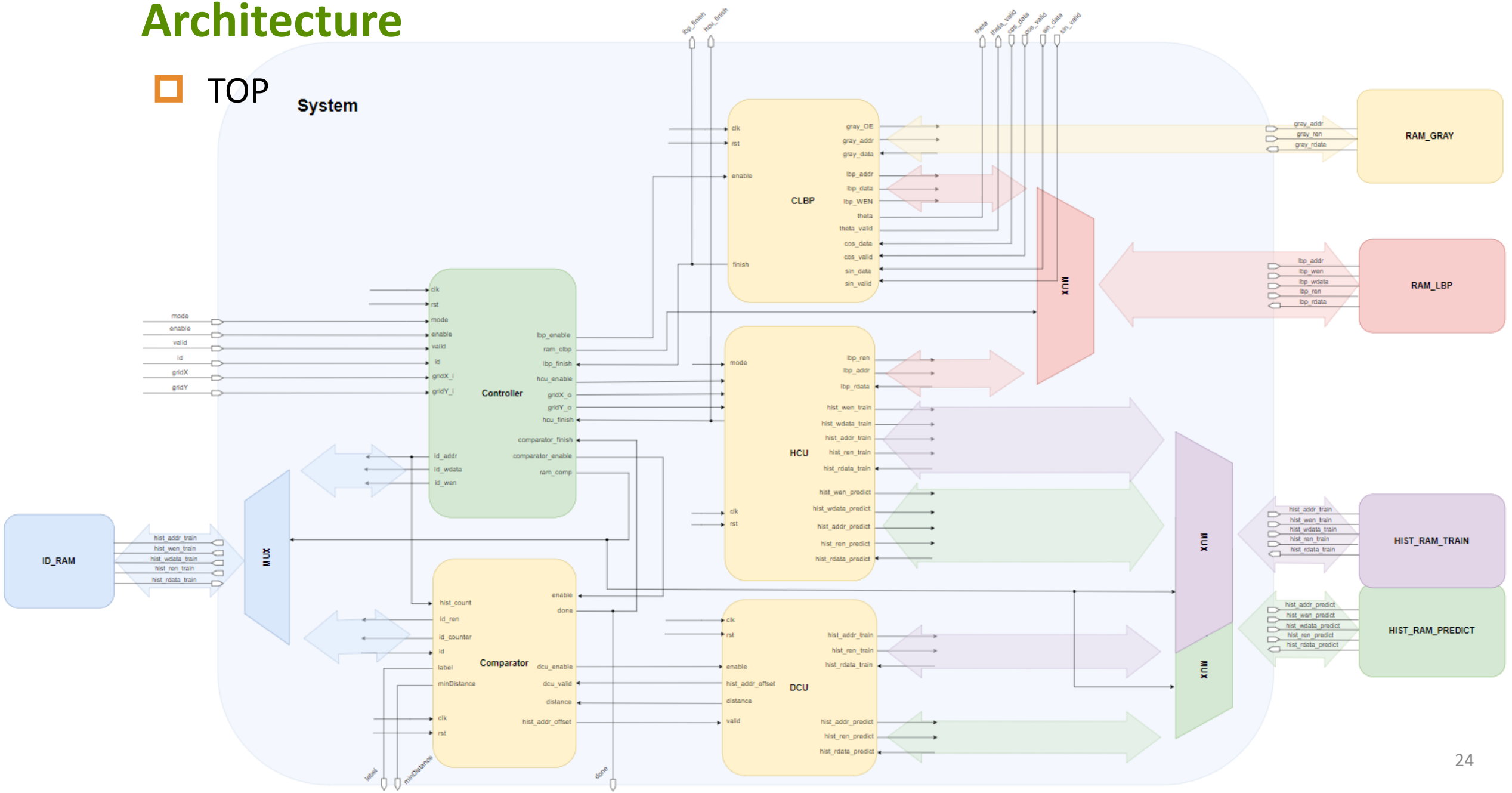
- Memory access



MUX are used to determine which module has the access to the memory

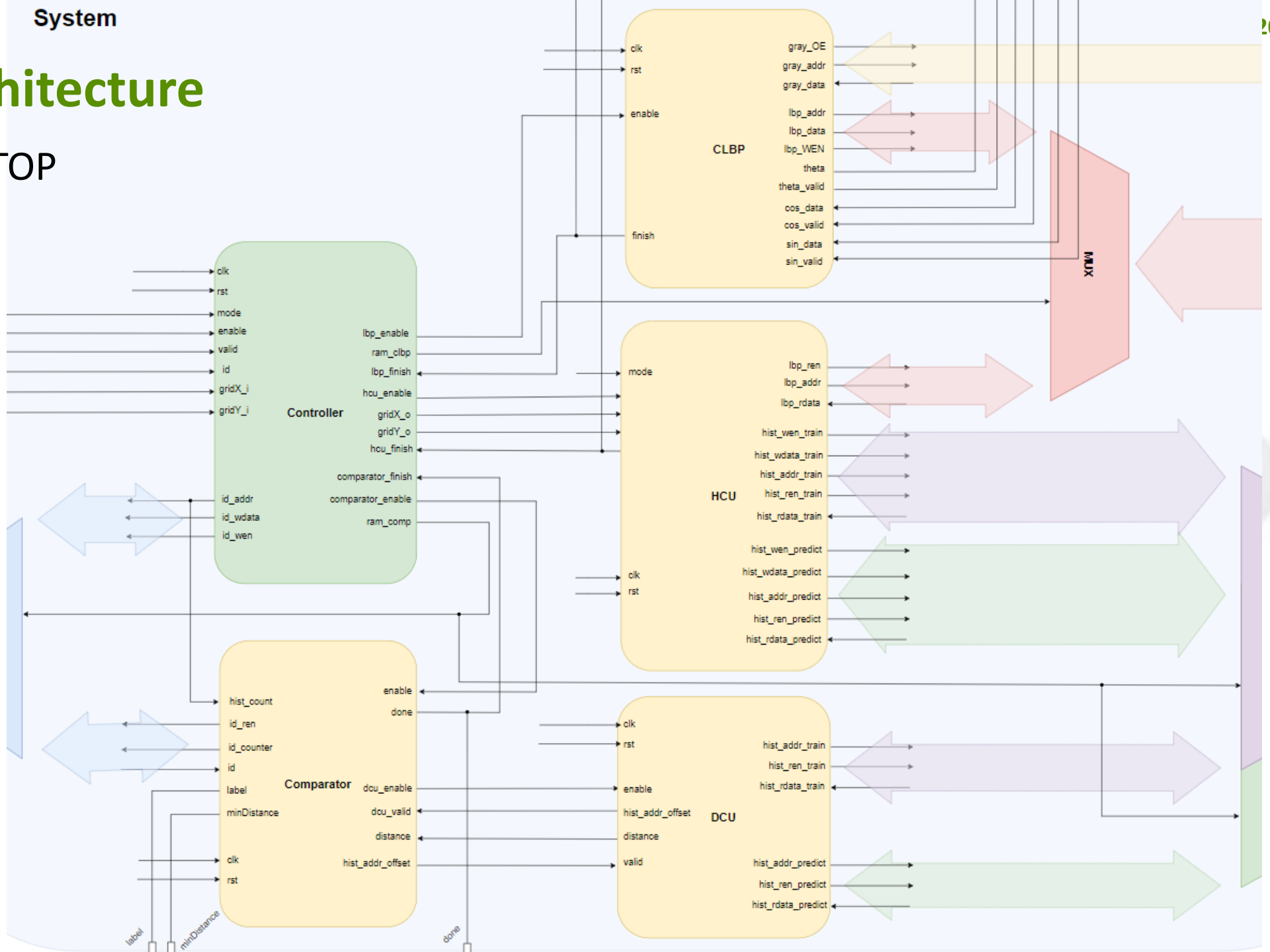
Architecture

- TOP



Architecture

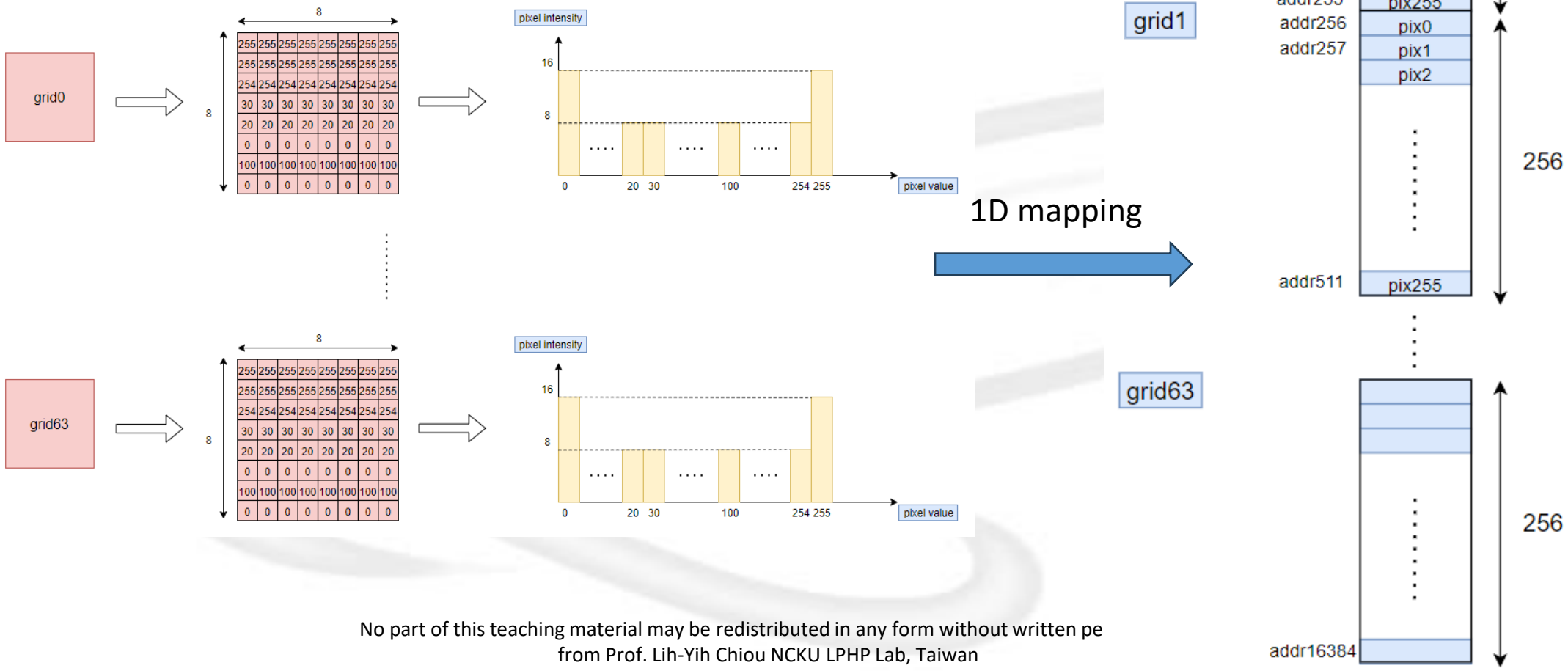
- TOP



Data Mapping in Memory

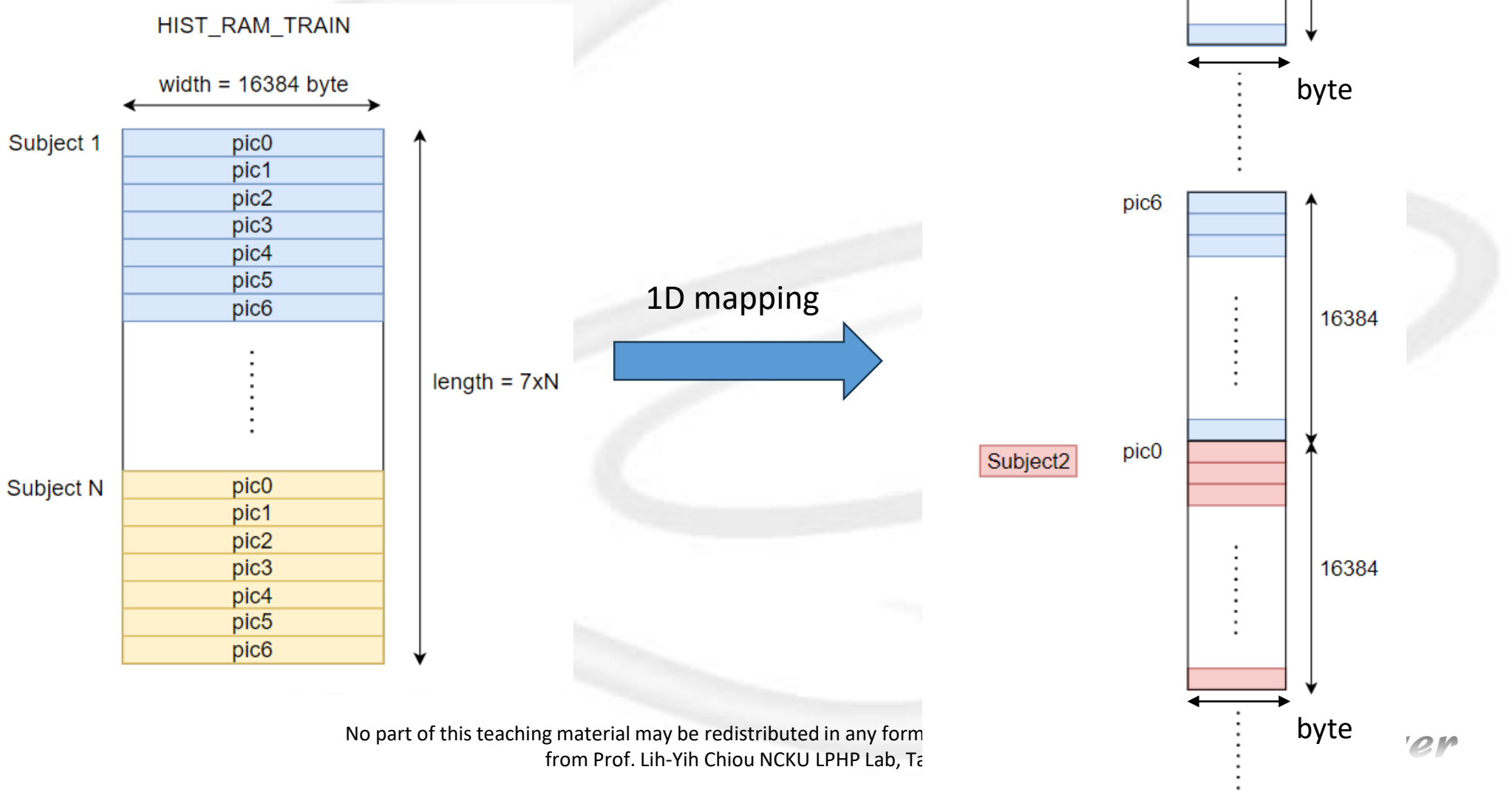
Data Mapping in Memory

- One subject histogram



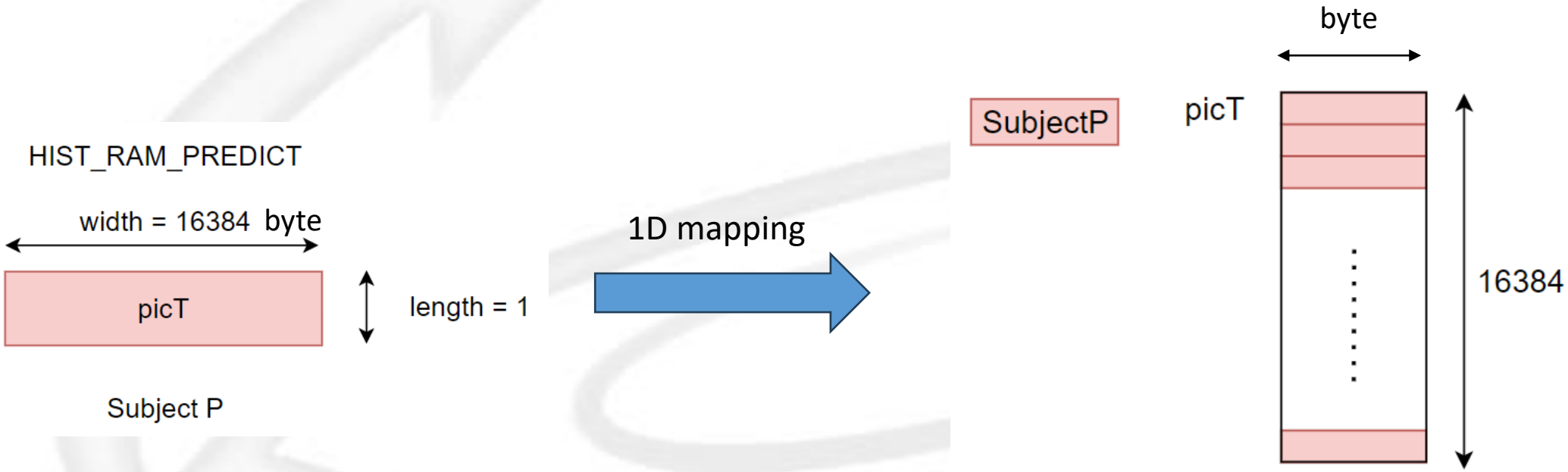
Data Mapping in Memory

- HIST_RAM_TRAIN, size of 8*8*256*(7+1)*N byte



Data Mapping in Memory

- HIST_RAM_PREDICT, size of 8*8*256 byte



I/O Definition - Top module

I/O Definition - TOP module (1/3)

Signal	I/O	Bit-width	Description
clk	I	1	Clock signal
rst	I	1	Reset signal
enable	I	1	Circuit enabling signal
mode	I	1	Indication signal of whether the system is in prediction or training phase, 0 is training, 1 is prediction
gridX	I	4	Image sliced portion in X direction, value is 8
gridY	I	4	Image sliced portion in Y direction, value is 8
valid	I	1	Indication that current subject ID is valid
id	I	5	Subject ID
hcu_finish	O	1	Indication that HCU circuit is done(should be asserted every time a subject picture is finished computing histogram)
label	O	5	Prediction result, output ID value
minDistance	O	18	Minimum distance between the prediction histogram and the closest histogram in HIST_TRAIN_RAM
done	O	1	Indication that prediction of one picture is finished

I/O Definition - TOP module (2/3)

Signal	I/O	Bit-width	Description
gray_addr	O	12	Address signal connected to RAM_GRAY
gray_ren	O	1	Read enable signal to RAM_GRAY
gray_rdata	I	8	Read data signal from RAM_GRAY
lbp_addr	O	12	Address signal connected to RAM_LBP
lbp_wen	O	1	Write enable signal to RAM_LBP
lbp_wdata	O	8	Write data signal to RAM_LBP
lbp_ren	O	1	Read enable signal to RAM_LBP
lbp_rdata	I	8	Read data signal from RAM_LBP
theta	O	25(fixed-point)	Current neighbor's theta signal(unit is in radian)
theta_valid	O	1	Indication signal of current neighbor's theta is valid
cos_data	I	25(fixed-point)	Cosine value of the theta(from testbench)
cos_valid	I	1	Indication signal of cosine value is valid
sin_data	I	25(fixed-point)	Sine value of the theta(from testbench)
sin_valid	I	1	Indication signal of sine value is valid
lbp_finish	O	1	Indication signal of the CLBP circuit is finished

I/O Definition - TOP module (3/3)

Signal	I/O	Bit-width	Description
id_addr	O	8	Address signal connected to ID_RAM
id_ren	O	1	Read enable signal to ID_RAM
id_rdata	I	5	Read data signal from ID_RAM
id_wen	O	1	Write enable signal to ID_RAM
id_wdata	O	5	Write data signal to ID_RAM
hist_addr_train	O	21	Address signal connected to HIST_RAM_TRAIN
hist_wen_train	O	1	Write enable signal to HIST_RAM_TRAIN
hist_wdata_train	O	8	Write data signal to HIST_RAM_TRAIN
hist_ren_train	O	1	Read enable signal to HIST_RAM_TRAIN
hist_rdata_train	I	8	Read data signal from HIST_RAM_TRAIN
hist_addr_predict	O	14	Address signal connected to HIST_RAM_PREDICT
hist_wen_predict	O	1	Write enable signal to HIST_RAM_PREDICT
hist_wdata_predict	O	8	Write data signal to HIST_RAM_PREDICT
hist_ren_predict	O	1	Read enable signal to HIST_RAM_PREDICT
hist_rdata_predict	I	8	Read data signal from HIST_RAM_PREDICT

I/O Definition - Submodule

I/O Definition - CLBP

Signal	I/O	Bit-width	Description
clk	I	1	Clock signal
rst	I	1	Reset signal
enable	I	1	CLBP circuit enabling signal
gray_addr	O	12	Address signal connected to RAM_GRAY
gray_OE	O	1	Read enable signal to RAM_GRAY
gray_data	I	8	Read data signal from RAM_GRAY
lbp_addr	O	12	Address signal connected to RAM_LBP MUX to memory
lbp_WEN	O	1	Write enable signal to RAM_LBP
lbp_data	O	8	Write data signal to RAM_LBP
theta	O	25(fixed-point)	Current neighbor's theta signal(unit is in radian)
theta_valid	O	1	Indication signal of current neighbor's thetas is valid
cos_data	I	25(fixed-point)	Cosine value of the theta (from testbench)
cos_valid	I	1	Indication signal of cosine value is valid
sin_data	I	25(fixed-point)	Sine value of the theta(from testbench)
sin_valid	I	1	Indication signal of sine value is valid
finish	O	1	Indication signal of the LBP circuit is finished

I/O Definition – HCU (1/2)

Signal	I/O	Bit-width	Description
clk	I	1	Clock signal
rst	I	1	Reset signal
mode	I	1	Indication signal of whether the system is in prediction or training phase, 0 is training, 1 is prediction
enable	I	1	HCU circuit enabling signal
gridX	I	4	Image sliced portion in X direction, value is 8
gridY	I	4	Image sliced portion in Y direction, value is 8
lbp_addr	O	12	Address signal connected to RAM_LBP MUX to memory
lbp_ren	O	1	Read enable signal to RAM_LBP
lbp_rdata	I	8	Read data signal from RAM_LBP

I/O Definition – HCU (2/2)

Signal	I/O	Bit-width	Description
hist_addr_train	O	21	Address signal connected to HIST_RAM_TRAIN MUX to memory
hist_wen_train	O	1	Write enable signal to HIST_RAM_TRAIN
hist_wdata_train	O	8	Write data signal to HIST_RAM_TRAIN
hist_ren_train	O	1	Read enable signal to HIST_RAM_TRAIN MUX to memory
hist_rdata_train	I	8	Read data signal from HIST_RAM_TRAIN
hist_addr_predict	O	14	Address signal connected to HIST_RAM_PREDICT MUX to memory
hist_wen_predict	O	1	Write enable signal to HIST_RAM_PREDICT
hist_wdata_predict	O	8	Write data signal to HIST_RAM_PREDICT
hist_ren_predict	O	1	Read enable signal to HIST_RAM_PREDICT MUX to memory
hist_rdata_predict	I	8	Read data signal from HIST_RAM_PREDICT
done	O	1	Indication that HCU circuit is done(should be asserted every time a subject picture is finished computing histogram)

I/O Definition - DCU

Signal	I/O	Bit-width	Description
clk	I	1	Clock signal
rst	I	1	Reset signal
enable	I	1	DCU circuit enabling signal
hist_addr_offset	I	21	Current computing histogram address offset
hist_addr_train	O	21	Address signal connected to HIST_RAM_TRAIN MUX to memory
hist_ren_train	O	1	Read enable signal to HIST_RAM_TRAIN MUX to memory
hist_rdata_train	I	8	Read data signal from HIST_RAM_TRAIN
hist_addr_predict	O	14	Address signal connected to HIST_RAM_PREDICT MUX to memory
hist_ren_predict	O	1	Read enable signal to HIST_RAM_PREDICT MUX to memory
hist_rdata_predict	I	8	Read data signal from HIST_RAM_PREDICT
distance	O	1	The computed distance value
valid	O	1	Indication that the current distance value is valid

I/O Definition - Comparator

Signal	I/O	Bit-width	Description
clk	I	1	Clock signal
rst	I	1	Reset signal
enable	I	1	Comparator circuit enabling signal
histcount	I	8	# IDs encountered during training mode
distance	I	1	DCU computed distance value
dcu_valid	I	1	Indication that the current distance value is valid
id	I	5	Id read data from ID_RAM
id_ren	O	1	Read enable signal to ID_RAM
id_counter	O	8	The current ID address it is processing MUX to memory
dcu_enable	O	1	DCU circuit enabling signal
label	O	5	Prediction result, output ID value
minDistance	O	18	Minimum distance between the prediction histogram and the closest histogram in HIST_TRAIN_RAM
hist_addr_offset	O	21	The address offset in HIST_RAM_TRAIN of the id it is processing currently
done	O	1	Indication signal of the Comparator circuit is finished

I/O Definition – Controller (1/2)

Signal	I/O	Bit-width	Description
clk	I	1	Clock signal
rst	I	1	Reset signal
mode	I	1	Indication signal of whether the system is in prediction or training phase, 0 is training, 1 is prediction
enable	I	1	Comparator circuit enabling signal
valid	I	1	Indication that current subject ID is valid
id	I	5	Subject ID
id_addr	O	8	Address signal connected to ID_RAM MUX to memory
id_wen	O	1	Write enable signal to ID_RAM
id_wdata	O	5	Write data signal to ID_RAM
lbp_enable	O	1	CLBP circuit enabling signal
lbp_finish	I	1	Indication of the CLBP circuit is finished
ram_clbp	O	1	Indication that the CLBP circuit has the access to RAM_LBP

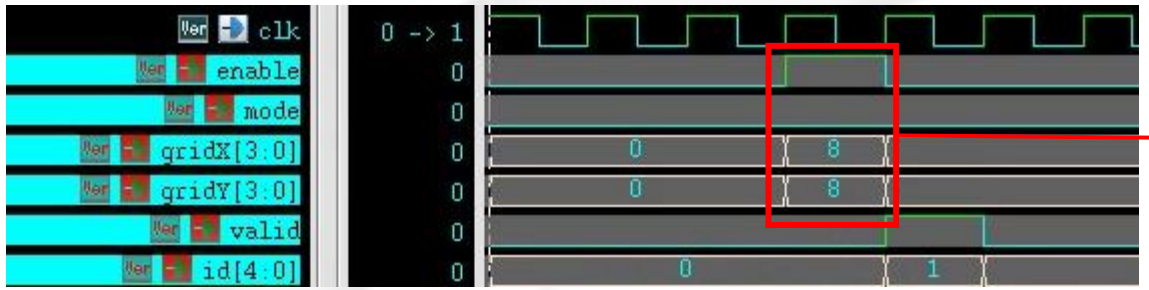
I/O Definition – Controller (2/2)

Signal	I/O	Bit-width	Description
gridX_i	I	4	Image sliced portion in X direction, value is 8, from testbench
gridY_i	I	4	Image sliced portion in Y direction, value is 8, from testbench
hcu_enable	O	1	HCU circuit enabling signal
gridX_o	O	4	Image sliced portion in X direction, value is 8, to HCU
gridY_o	O	4	Image sliced portion in Y direction, value is 8, to HCU
hcu_finish	I	1	Indication of the HCU circuit is finished
comparator_finish	I	1	Indication of the Comparator circuit is finished
comparator_enable	O	1	Comparator circuit enabling signal
ram_comp	O	1	Indication that the Comparator circuit & DCU circuit has the access to ID_RAM, HIST_RAM_TRAIN, HIST_RAM_PREDICT

Waveform

Waveform

- TOP module signal, see TOP I/O, training mode



gridX & *gridY* are sent with *enable* signal

- *id* signal & *valid* signal will be renewed 2 cycles after *hcu_finish* is asserted

- Prediction mode

mode signal transition 0-> 1

Waveform

- Overall system behavior
  - Note: *hcu_finish & lbp_finish & done* signal should be asserted **only one cycle** every time when it is supposed to pull high

Testbench checks the content in RAM_LBP whenever *lbp_finish* is asserted

Testbench checks the content in HIST_RAM_TRAIN whenever *hcu_finish* is asserted

Testbench checks *label* & *minDistance* when *done* is asserted

Lab7 Session

- Lab7
  - Lab7_1
    - Complete **training mode**, no error occurred in training phase
  - Lab7_2
    - Complete **prediction mode**, all predictions are correct

Simulation

Simulation

Command

Lab7	Commands
superlint	% cd script % jg –superlint superlint.tcl
synthesis	% cd script % dv –f synthesis.tcl
Pre-sim	% cd sim % vcs -R -sverilog top_tb.sv -debug_access+all -full64
Post-sim	% cd sim % vcs -R -sverilog top_tb.sv -debug_access+all -full64 +define+SDF+SYN
Dump waveform	+define+FSDB

Don’t use +define+FSDB when running post-sim, it’ll occupy substantial amount of memory!

Simulation

- Training phase (lab7_1)
  - Check content in RAM_LBP first
  - Check content in HIST_RAM_TRAIN secondly
  - Simulation will be aborted if any error occurred

```
================== Training begins!!! ========================

        ********************************
        **   subject1_0 CLBP PASS!!   **
        ********************************


        ************************************
        **   subject1_0 Histogram PASS!!   **
        ************************************


        ********************************
        **   subject1_1 CLBP PASS!!   **
        ********************************


        ************************************
        **   subject1_1 Histogram PASS!!   **
        ************************************
```

Simulation

- Prediction phase (lab7_2)
  - Check content in RAM_LBP first
  - Check content in HIST_RAM_PREDICT secondly
  - Check predicted *label* signal & *minDistance* signal thirdly
  - Simulation will be aborted if any error occurred

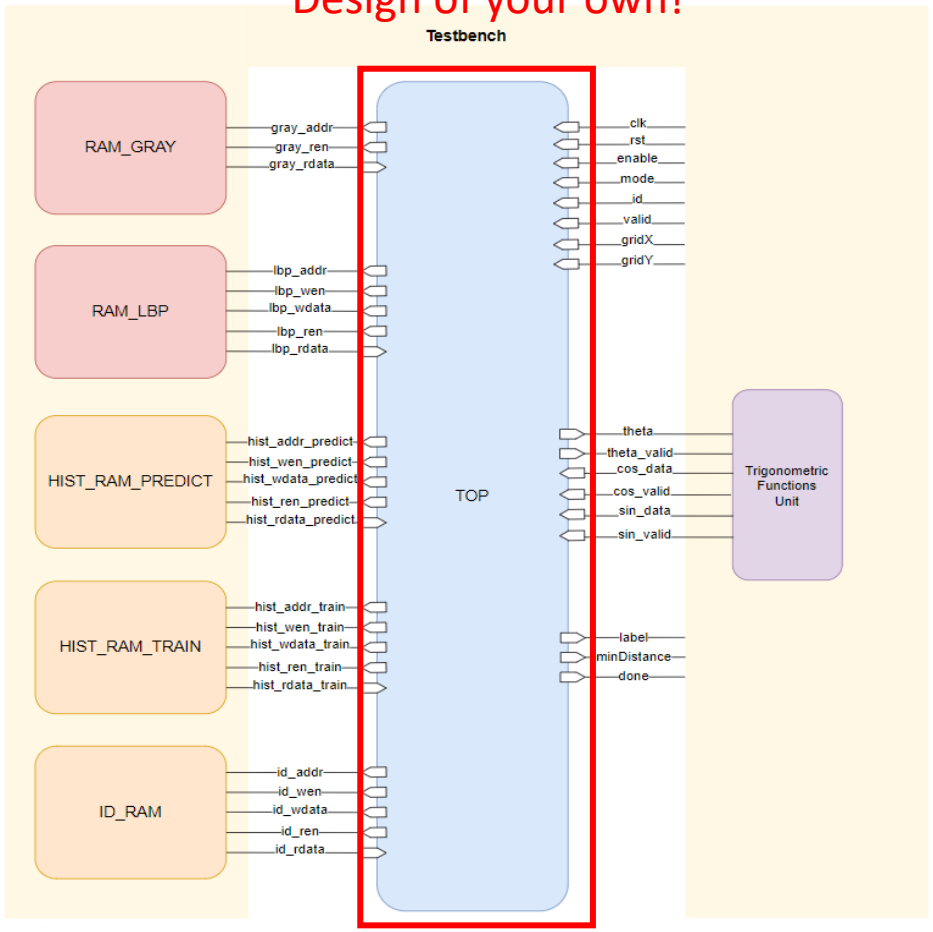
```
================== Prediction begins!!! ========================

Prediction of subject P:          2 with pic T:          9.

        *********************************
        ** Prediction of Subject  2 PASS!!   **
        *********************************


Prediction of subject P:          3 with pic T:          7.

        *********************************
        ** Prediction of Subject  3 PASS!!   **
        *********************************


Prediction of subject P:          2 with pic T:          10.

        *********************************
        ** Prediction of Subject  2 PASS!!   **
        *********************************
```

Grading Policy

Grading Policy

- Based on this top diagram architecture, no any I/O port should be modified
  - ➔ However, the internal architecture can be the design of your own
  - ➔ Submodules doesn't have to be as same as those provided
- Special design will get higher score!

Design of your own!



Grading Policy

- PA Rank

Note: clock period must be No more than 2.0 ns!!!

→ P : Post-simulation time (= total cycle*period, unit in ns)

```
**************************
**                      **       |__||
**  Congratulations !!  **      / 0.0  |
**                      **     /_____   |
**  Simulation PASS!!   **    /^ ^ ^ \  |
**                      **   |^ ^ ^ ^ |w|
**************************    \m___m__|_|

total simulation time: 332169000 ns
$finish called from file "top_tb.sv", line 639.
$finish at simulation time         33216920034
           V C S   S i m u l a t i o n   R e p o r t
Time: 33216920034 ps
CPU Time:   1829.890 seconds;       Data structure size:   7.1Mb
```

→ A : Total cell area

```
Number of ports:                         2518
Number of nets:                         11103
Number of cells:                         7766
Number of combinational cells:           7144
Number of sequential cells:               577
Number of macros/black boxes:               0
Number of buf/inv:                       1093
Number of references:                      11

Combinational area:              3454.410320
Buf/Inv area:                     179.573766
Noncombinational area:            537.943687
Macro/Black Box area:               0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:                 3992.354007
Total area:                 undefined
```

Grading Policy

- Lab7 (110%)
  - Lab7_1 (30%)
    - RTL pass (10 %)
    - SYN pass (clock period <= 2.0ns) (15 %)
    - Superlint >= 90% (5 %)
  - Lab7_2 (65%)
    - RTL pass (10 %)
    - SYN pass (clock period <= 2.0ns) (20 %)
    - PA (30 %)
    - Superlint >= 90% (5 %)
  - Report (5%)
  - Demo (10%)

Note: clock period must be No more than 2.0 ns!!!

Note: clock period must be No more than 2.0 ns!!!

Note: clock period must be No more than 2.0 ns!!!

Friendly reminder

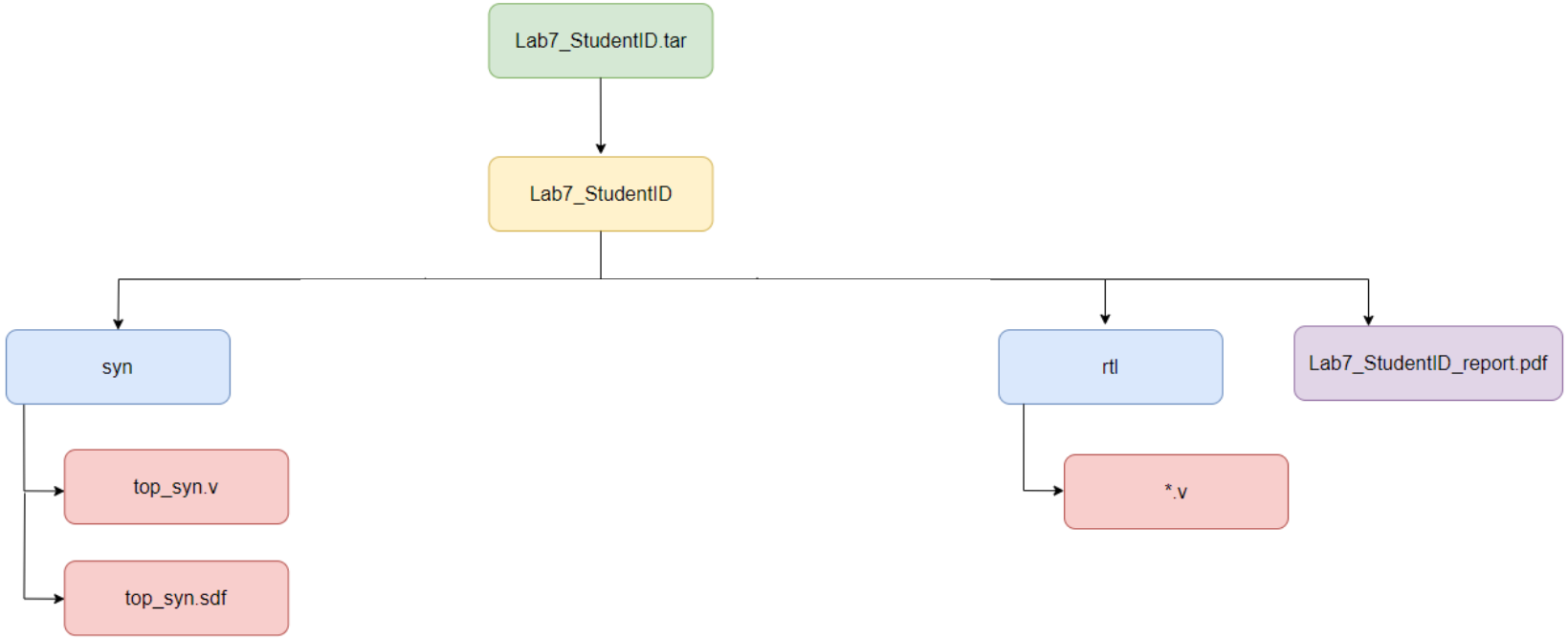
- Friendly reminder
  - Discussion with peers is recommended, but do not cheat
  - Warning! Any dishonesty found will result in zero grade
  - Warning! Do not submit in the last minute, any late submission will be handled by the late submission policy
  - Warning! Please make sure you submit the correct file; submitted the wrong file and already pass the deadline will be viewed as late submission and will also be handled by late submission policy
  - Warning! Please make sure that your code can be compiled in the SoC environment, any dead body that we cannot compile, will also receive zero

Late submission policy

- Within 24 hours: **30% deduction of graded score**
- Within 48 hours: **60% deduction of graded score**
- Within 72 hours: **90% deduction of graded score**
- After 72 hours: **100% deduction of graded score**

File Hierarchy

- Please follow the file hierarchy when submitting homework
  - ➔ Warning!!! Deduction of 5 points if not allowed!



Note

Note

- Use asynchronous reset
  - ➔ Always **reset to 0** to avoid unanticipated behavior

```
always@(posedge clk or posedge rst) begin
    if(rst) begin
        signal <= 1'b0;
    end
    else begin
        // ...
    end
end
```

Note

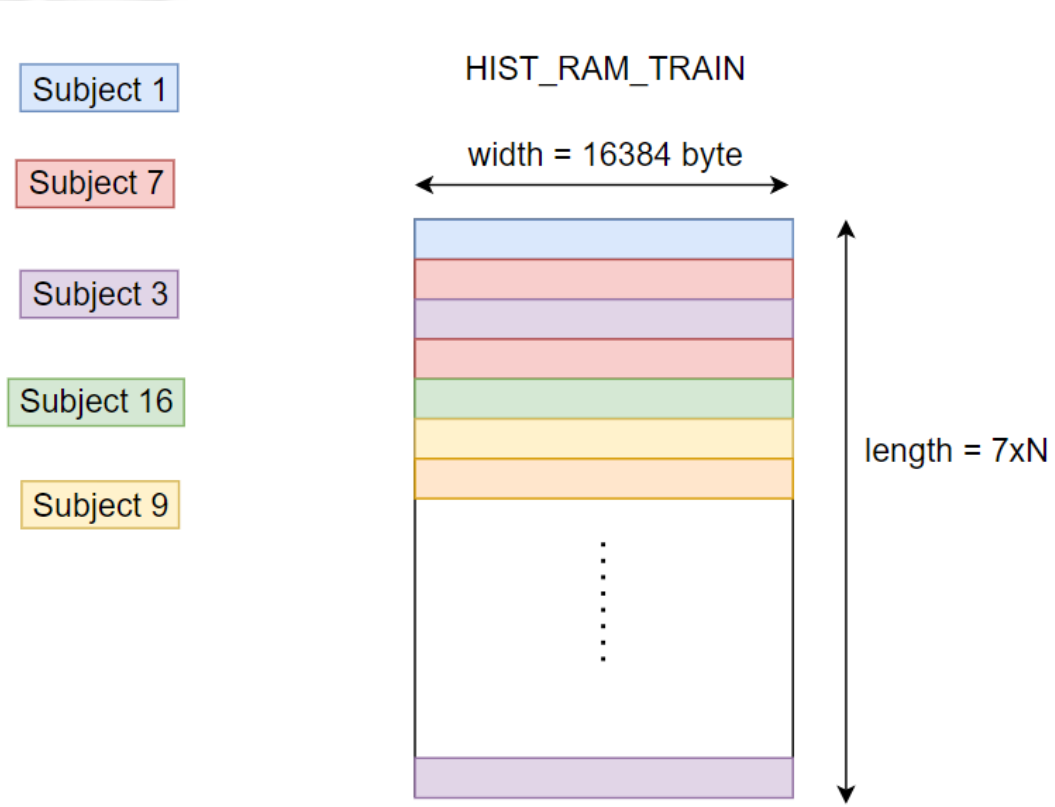
- Synthesis script, added **–attribute** in report_timing command
  - ➔ See clearly what path is infeasible (i.e. why your slack is always negative no matter how large the clock period you adjust)

```
report_timing -attributes > ../syn/timing.log
```

```
Attributes:
    d - dont_touch
    u - dont_use
   mo - map_only
   so - size_only
    i - ideal_net or ideal_network
  inf - infeasible path
```

Note

- Training mode, ID does not necessary given in a continuous order
  - For instance, N=5, meaning 5 different subjects
  - Each subject has 7 pictures fed into the system, but not in order



Appendix

TCL file

- How to synthesize without GUI
  - It's no need to go through every synthesis process.
  - By creating a TCL script, you can automate the design process
  - You can modify synthesis.tcl file on your own

```
1  #Read All Files
2  read_file -format verilog  {../rtl/top.v}
3  #read_file -format sverilog  top.sv
4  current_design top
5  link
6
7  #Setting Clock Constraints
8  source -echo -verbose DC.sdc
9  check_design
10 set high_fanout_net_threshold 0
11 uniquify
12 set_fix_multiple_port_nets -all -buffer_constants [get_designs *]
13
14 #Synthesis all design
15 compile -exact_map -map_effort high
16
17 write -format ddc     -hierarchy -output "../syn/top_syn.ddc"
18 write_sdf -version 2.1  ../syn/top_syn.sdf
19 write -format verilog -hierarchy -output ../syn/top_syn.v
20 report_area > ../syn/area.log
21 report_timing -attributes > ../syn/timing.log
22 report_power > ../syn/power.log
23 report_qor   >  ../syn/top_syn.qor
```

Source SDC file

Compile your design with high map_effort

Save synthesized design file

Write out report file

SDC file

- SDC file
  - Can only change cycle period
  - To ensure proper functionality, it is essential to adjust the testbench cycle period to match that of the SDC file configuration

```
# operating conditions and boundary conditions #

set clk_period  2.0

set input_max   [expr {double(round(1000*$clk_period * 0.6))/1000}]
set input_min   [expr {double(round(1000*$clk_period * 0.0))/1000}]
set output_max  [expr {double(round(1000*$clk_period * 0.6))/1000}]
set output_min  [expr {double(round(1000*$clk_period * 0.0))/1000}]

create_clock -period $clk_period [get_ports clk]
set_dont_touch_network      [get_clocks clk]
set_clock_uncertainty  0.02  [get_clocks clk]
set_clock_latency      0.2   [get_clocks clk]

set_input_delay  -clock clk  -max $input_max  [all_inputs]
set_input_delay  -clock clk  -min $input_min  [all_inputs]

set_output_delay -clock clk  -max $output_max [all_outputs]
set_output_delay -clock clk  -min $output_min [all_outputs]

set_driving_cell -library N16ADFP_StdCellss0p72vm40c -lib_cell BUFFD4BWP16P90LVT -pi
set_driving_cell -library N16ADFP_StdCellss0p72vm40c -lib_cell DFQD1BWP16P90LVT  -pi
set_load [load_of "N16ADFP_StdCellss0p72vm40c/DFQD1BWP16P90LVT/D"] [all_outputs]

set_operating_conditions -min_library N16ADFP_StdCellff0p88v125c -min ff0p88v125c \
                         -max_library N16ADFP_StdCellss0p72vm40c -max ss0p72vm40c
set_wire_load_model -name ZeroWireload -library N16ADFP_StdCellss0p72vm40c

set_max_fanout 20 [all_inputs]
```

No more than 2.0 is allowed

LPHPLAB
VLSI Design LAB

Thanks for Your Attention