HW4
QR Code Decoder

張嘉祐

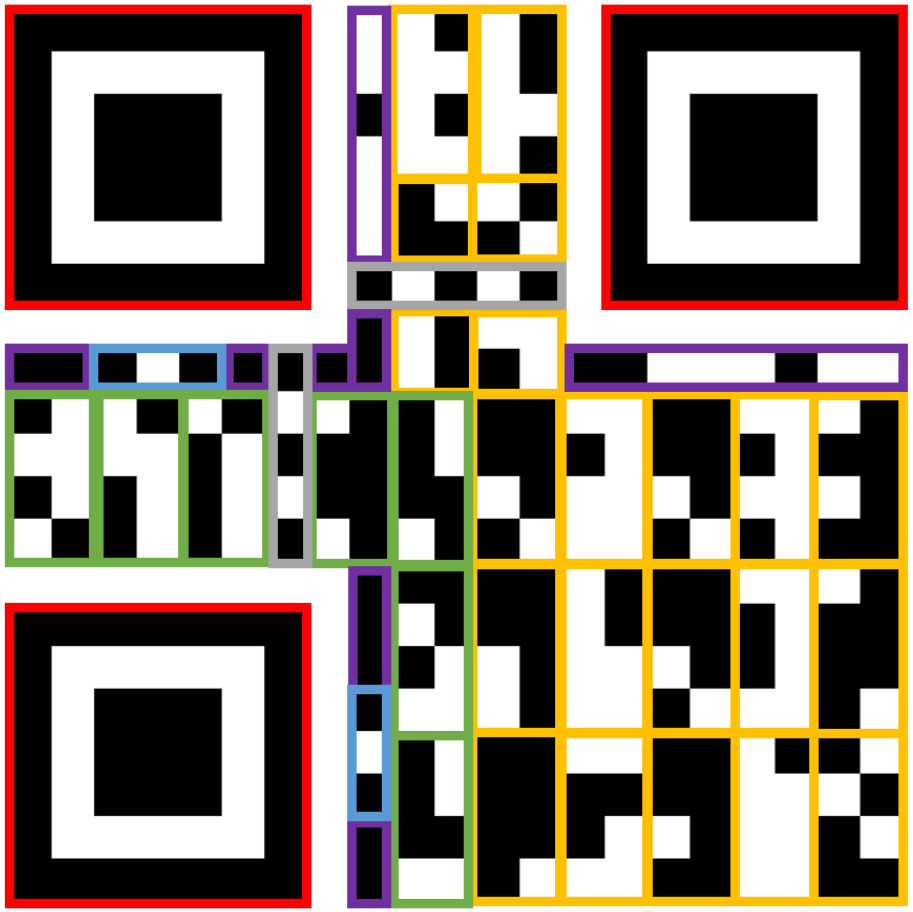
2022/11/03

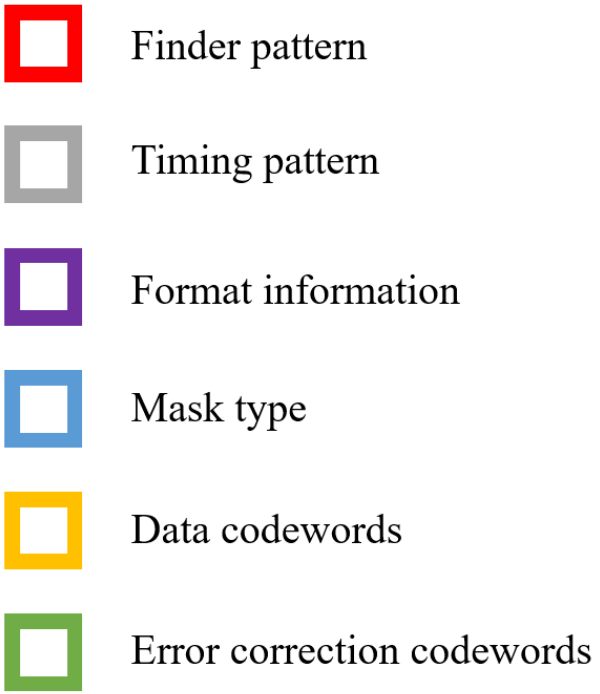
Introduction

14-byte “www.google.com”

- QR Code
  - Quick Response Code
  - Widely used for URL (Uniform Resource Locator) instant access
    網址
  - A variety of spec
    - Version 1 to 40 (from 21x21 to 177x177 pixels)
    - Different character set
    - Different level of error correction capability
  - Spec used in this assignment
    - Version 1 (21x21 pixels)
    - 8-bit Byte mode with ISO 8859-1 code
    - Error correction level L (7% recovery capacity)

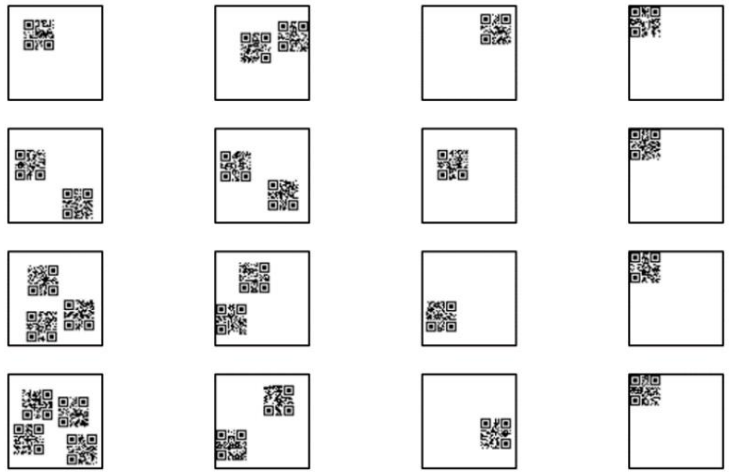
Structure of version 1-L QR Code





Four types of test pattern

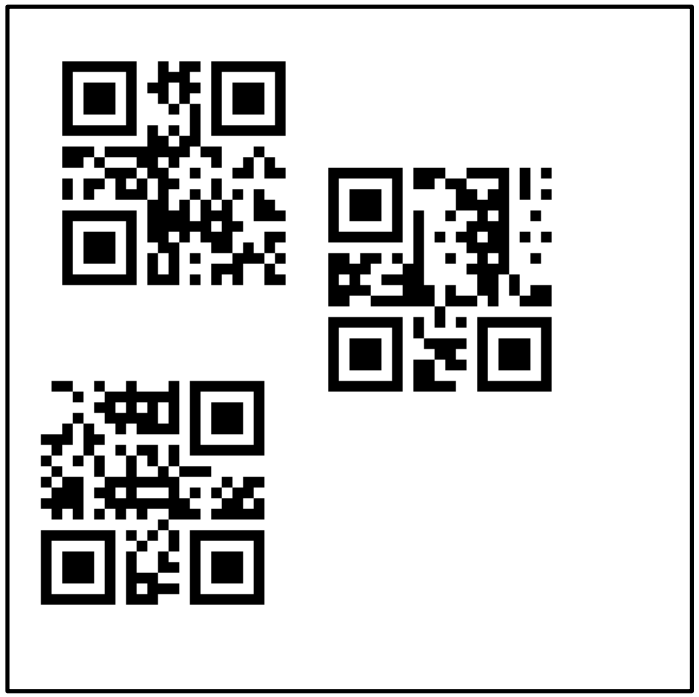
Pattern	# of QR Code in a pattern	Location	Rotation
Rank A	Arbitrary (1 to 4)	Arbitrary	Arbitrary
Rank B	2	Arbitrary	Arbitrary
Rank C	1	Arbitrary	Arbitrary
Rank D	1	Upper-Left	0-degree



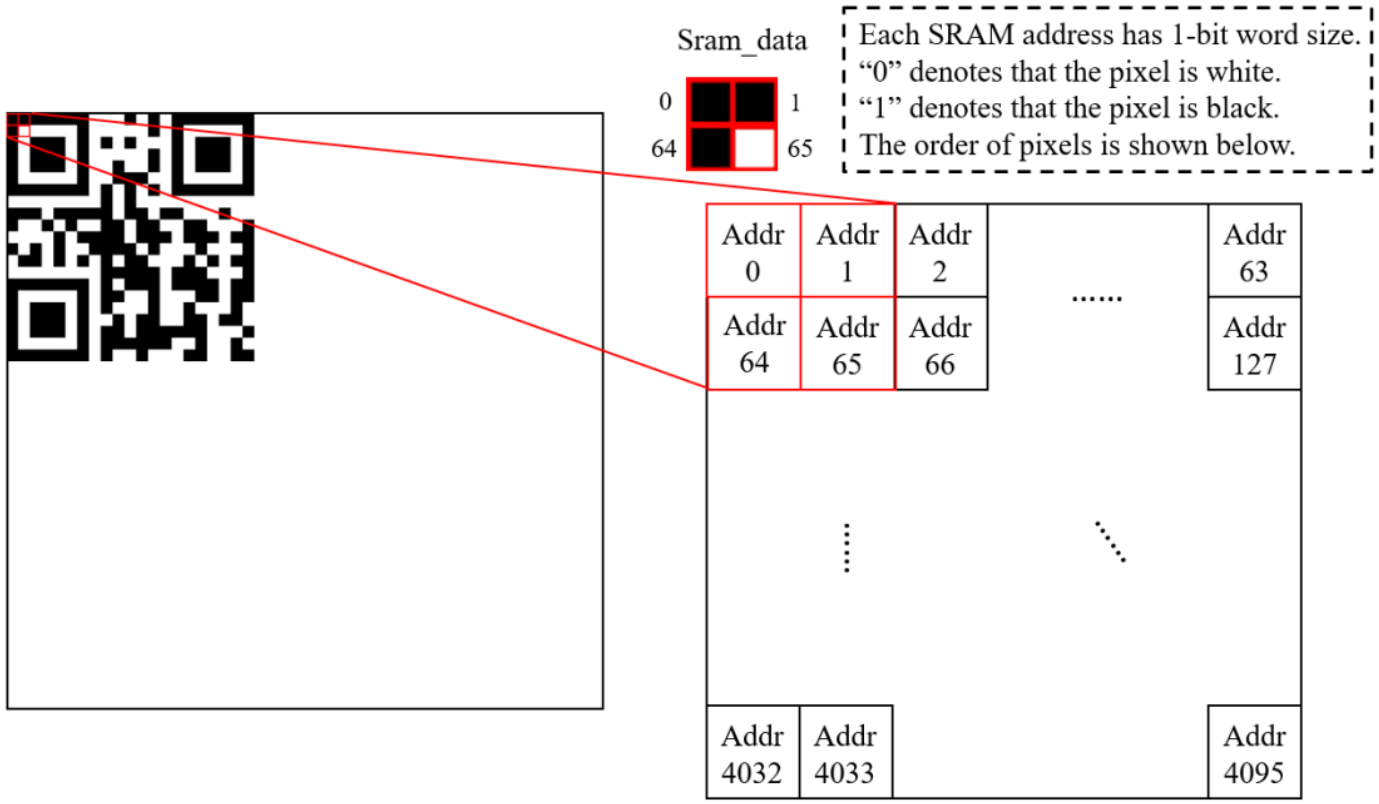
In the real world

What will happen if we scan multiple QR codes at a time?

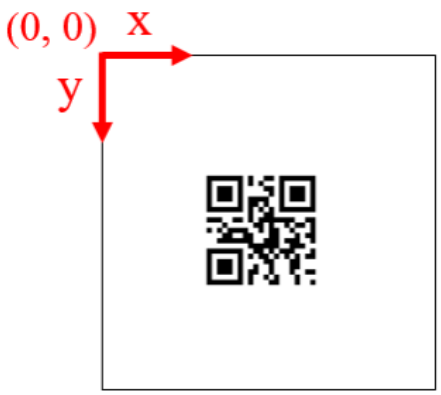
1. Download app “Google Lens”
2. Scan below QR Codes with this app

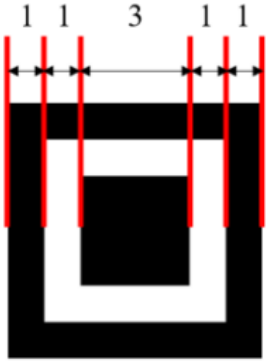


Data arrangement in SRAM



Rotation and Location of QR code





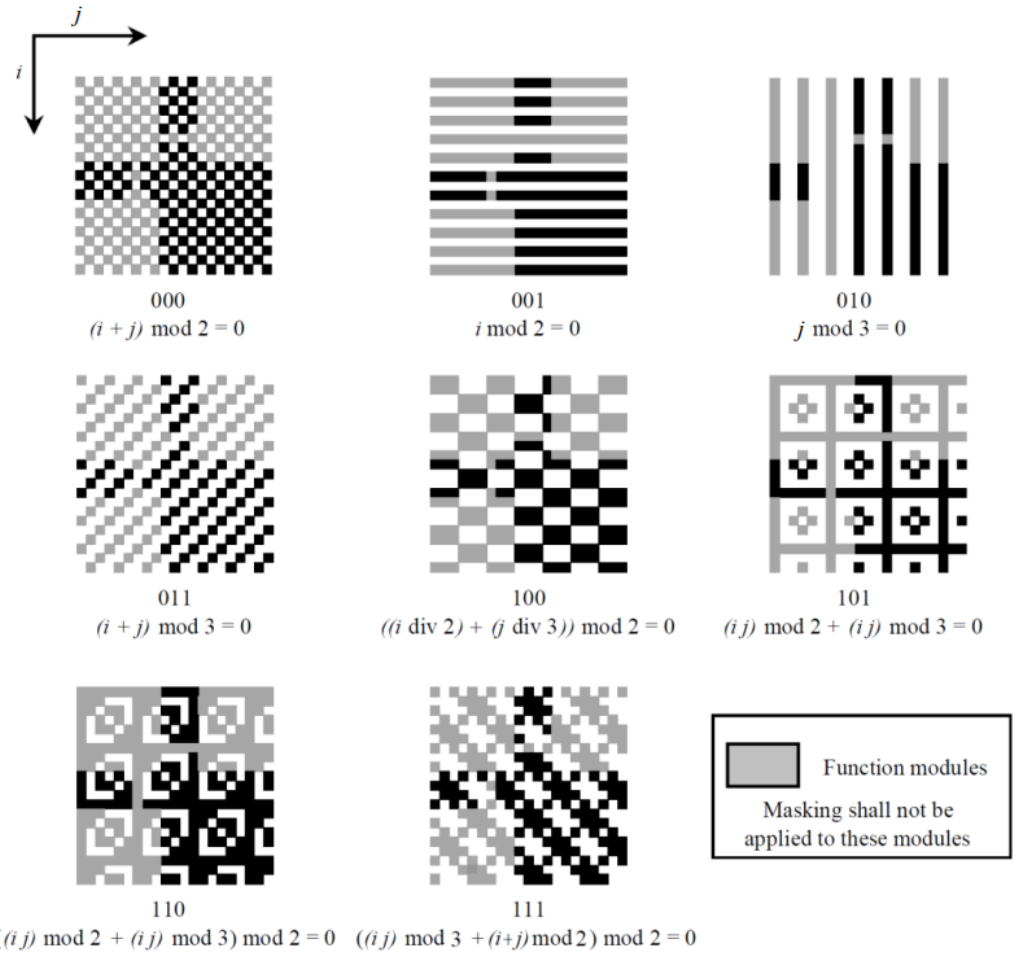
Finder Pattern

Rotation 0° Rotation 90° Rotation 180° Rotation 270°

De-masking process

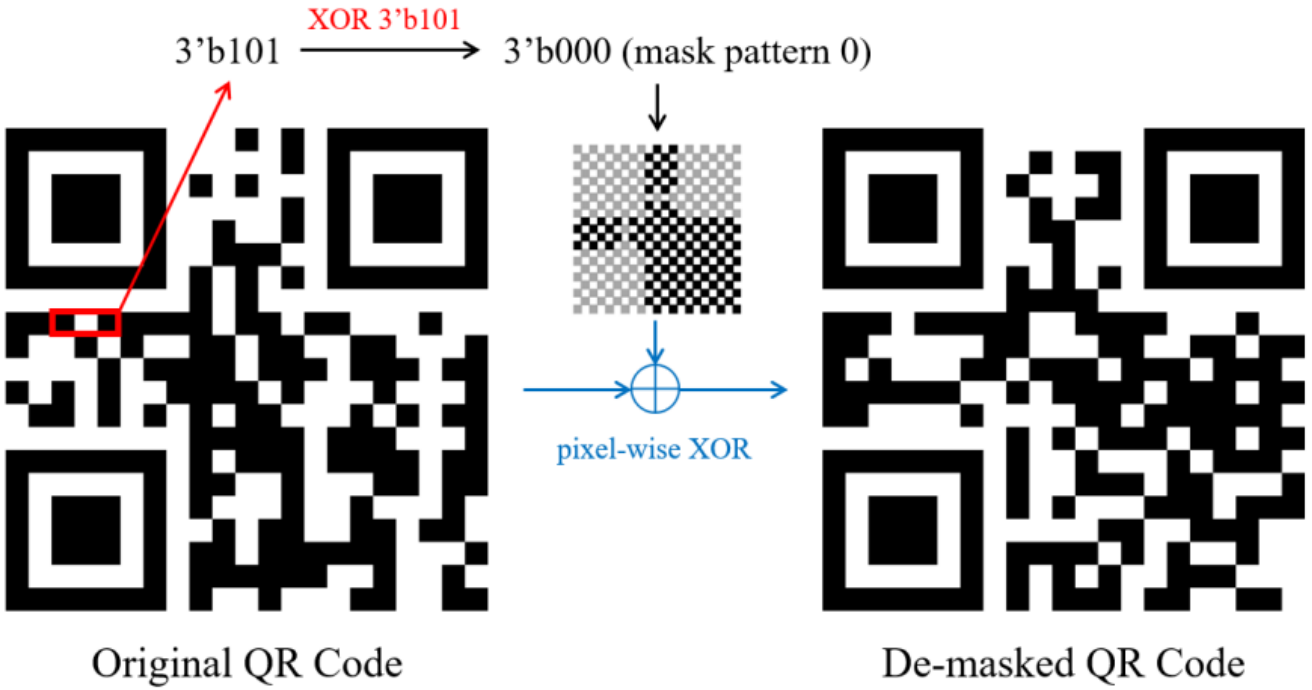
Mask Pattern Reference	Condition
000	$(i + j) \bmod 2 = 0$
001	$i \bmod 2 = 0$
010	$j \bmod 3 = 0$
011	$(i + j) \bmod 3 = 0$
100	$((i \text{ div } 2) + (j \text{ div } 3)) \bmod 2 = 0$
101	$(i\,j) \bmod 2 + (i\,j) \bmod 3 = 0$
110	$((i\,j) \bmod 2 + (i\,j) \bmod 3) \bmod 2 = 0$
111	$((i\,j) \bmod 3 + (i+j) \bmod 2) \bmod 2 = 0$

- Eight kinds of mask pattern

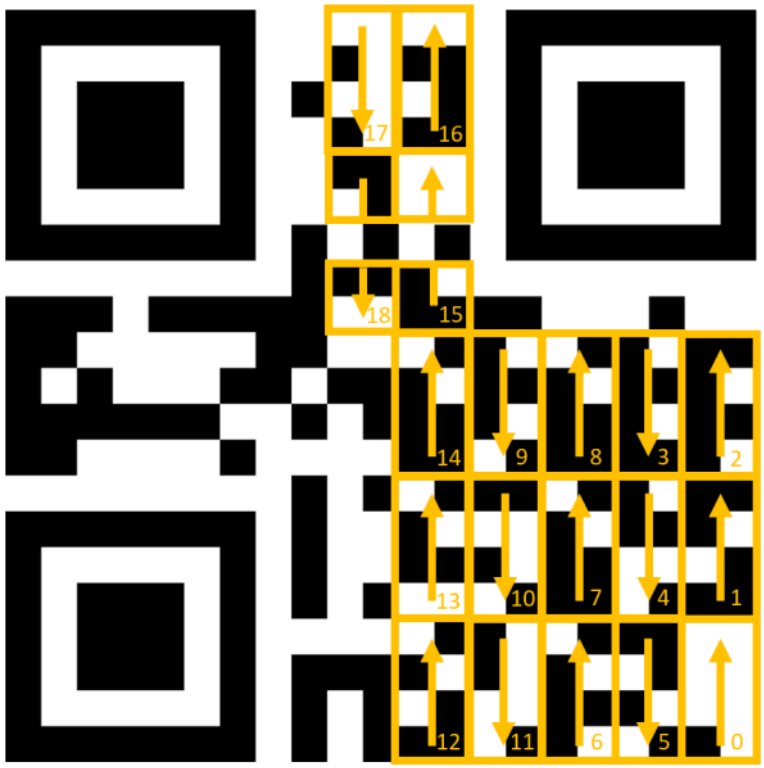


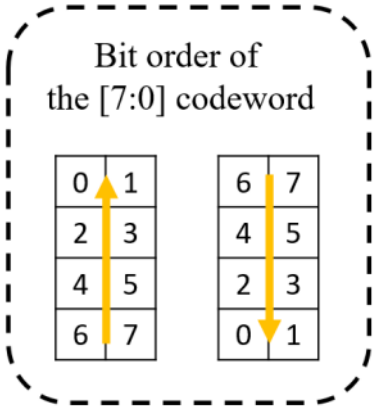
De-masking process

- The 3-bit data should first be XORed with a fixed 3’b101 to derive the real mask ID



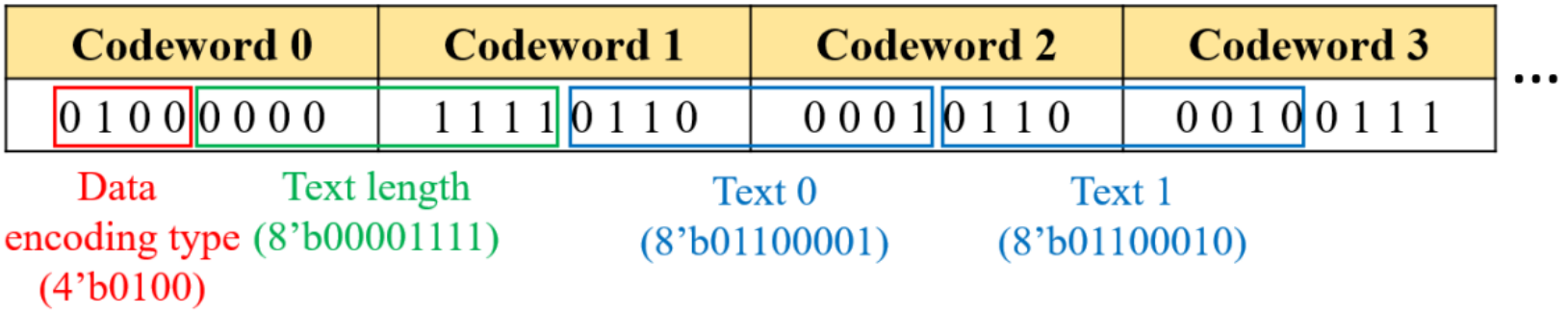
Decode the data codewords





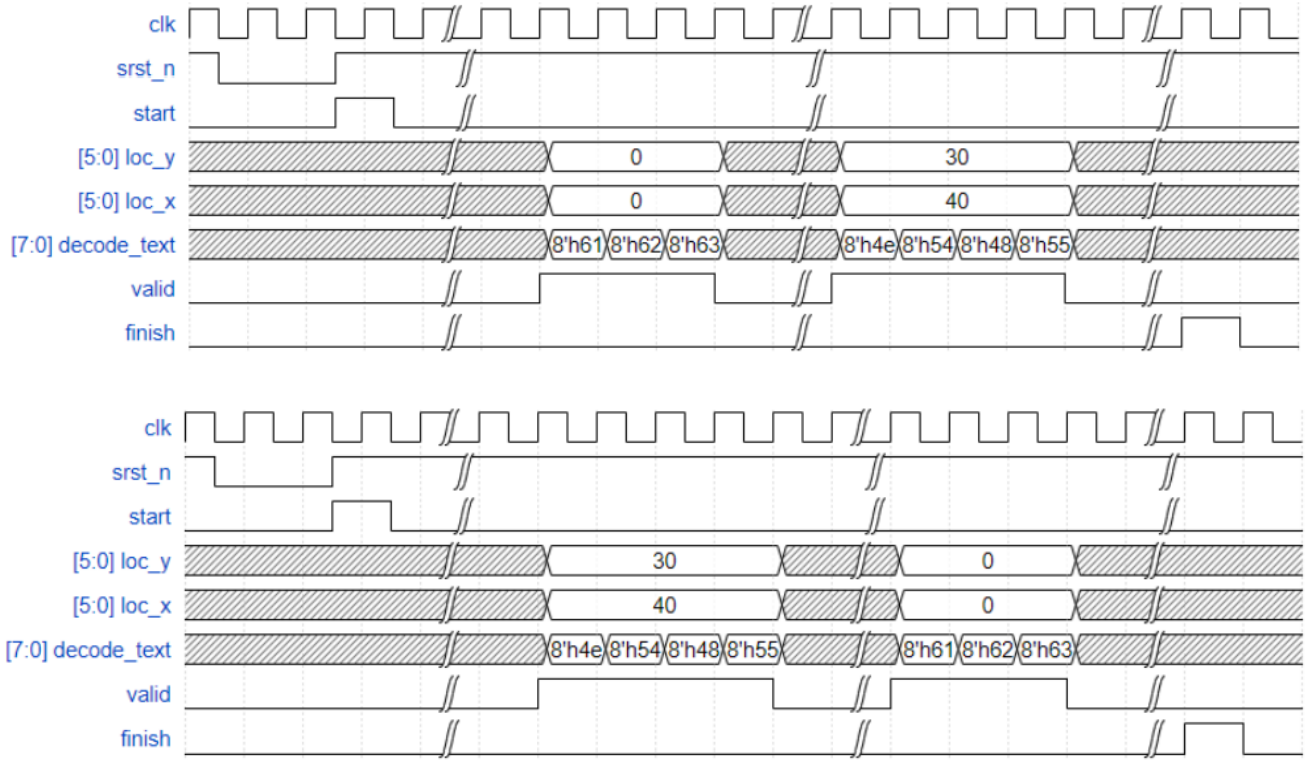
Decode the text sequentially

- Data arrangement in data codewords
  - The text length information tells you how many characters should be decoded
  - Once decoding a text data, you can put it to the output port *[7:0] decode_text* and set the output port *valid* to high



Timing diagram

- Assume two QR codes in a pattern
  - One at (0, 0), {8'h61, 8'h62, 8'h63} which represents "abc"
  - One at (30, 40), {8'h4e, 8'h54, 8'h48, 8'h55} which represents "NTHU"



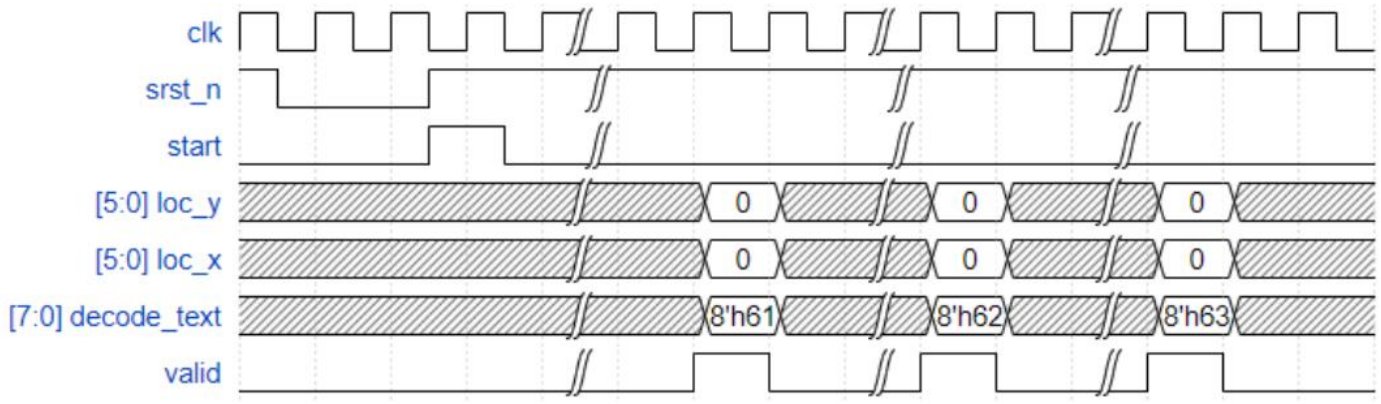
QR Code at (0, 0) is output first

The output order of QR Codes is not limited

QR Code at (30, 40) is output first

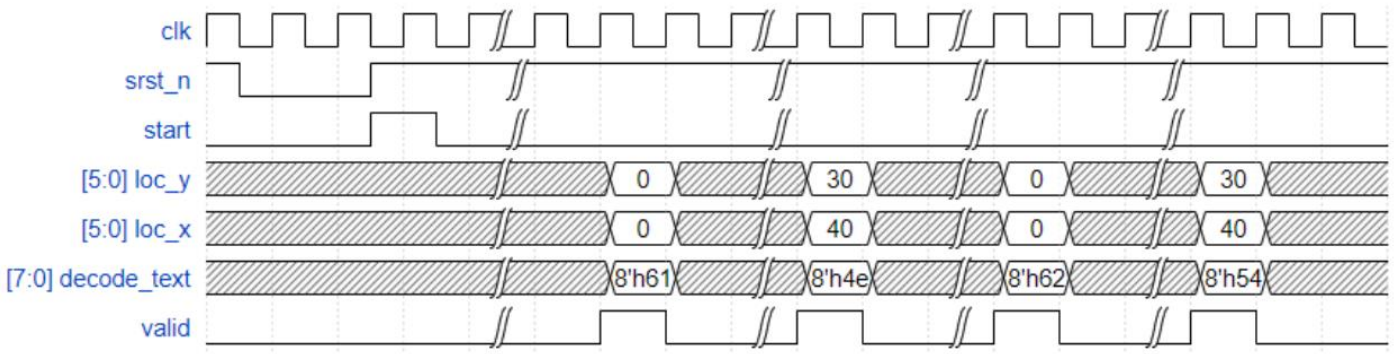
Timing diagram

- The output *decode_text* is not mandatory to be consecutive
- But the order of the output *decode_text* should still be correct
  - If the text is “abc”, you should first output “a”, and then “b”, and then “c”



Timing diagram

- You can only output the decoded result of the next QR Code after all the decoded result of the previous one have been output
- Below is **Not** acceptable output behavior



Simulation

- For Rank A, there are total 300 patterns
- For Rank B, Rank C, and Rank D, there are total 100 patterns for each Rank
- To pass each Rank, your design should pass all patterns in the Rank

```
# Rank A
vcs -f hdl.f -full64 -R -debug_access+all +v2k \
+define+RANK_A+PAT_L=0+PAT_U=299

# Rank B
vcs -f hdl.f -full64 -R -debug_access+all +v2k \
+define+RANK_B+PAT_L=0+PAT_U=99

# Rank C
vcs -f hdl.f -full64 -R -debug_access+all +v2k \
+define+RANK_C+PAT_L=0+PAT_U=99

# Rank D
vcs -f hdl.f -full64 -R -debug_access+all +v2k \
+define+RANK_D+PAT_L=0+PAT_U=99
```

For debugging convenience, you can modify **PAT_L** and **PAT_U** to simulate with any specific range of patterns

Synthesis

- For fairness, please do logic synthesis with the provided scripts
- Only two parts of the scripts can be modified
  - **line 16** in **0_readfile.tcl**
    - Add your hdl files
  - **line 5** in **synthesis.tcl**
    - Change your clock timing constraint

```
12  # Define a lib path
13  define_design_lib $TOPLEVEL -path ./$TOPLEVEL
14
15  # Add your hdl files here
16  analyze -library $TOPLEVEL -format verilog "../hdl/qrcode_decoder.v"
17
18  # Elaborate your design
19  elaborate $TOPLEVEL -architecture verilog -library $TOPLEVEL
20
```

0_readfile.tcl

```
1   # Set your TOPLEVEL here
2   set TOPLEVEL "qrcode_decoder"
3
4   # Change your timing constraint here
5   set TEST_CYCLE 10.0
6
```

synthesis.tcl

Performance index (PI)

$$PI = A \times T \times C^2$$

A: Total cell area which is shown in report_area_qrcode_decoder.out

T: TEST_CYCLE (your setting of clock timing constraint when synthesis)

C: Total cycle count to complete the whole simulation. It is shown on the terminal such as below when the whole simulation is passed and finished:

```
========================== Summary ==========================
Congratulation! All patterns are successfully passed! \(O v O)/
Total cycle count C = xxxxxx
```

- Note

  We take the square of C when computing PI in this assignment, which means the effect of C on the PI is larger than A and T.

Grading policy

Total Cycle Count C in PI depends on the simulation result of pattern

Passed Pattern	PI Grade	Synthesis PI ($A \times T \times C^2$)	Functionality Score (pass pre-sim)	Synthesis PI Score	Total Score
Rank A	A1	$PI \leq 4.7 \times 10^{15}$	7	8	15
Rank B Rank C	A2	$4.7 \times 10^{15} < PI \leq 8.4 \times 10^{15}$	7	6	13
Rank D	A3	$PI > 8.4 \times 10^{15}$	7	4	11
Rank B	B1	$PI \leq 2.8 \times 10^{14}$	5	4	9
Rank C	B2	$2.8 \times 10^{14} < PI \leq 5 \times 10^{14}$	5	3	8
Rank D	B3	$PI > 5 \times 10^{14}$	5	2	7
	C1	$PI \leq 8.6 \times 10^{13}$	3	3	6
Rank C	C2	$8.6 \times 10^{13} < PI \leq 1.5 \times 10^{14}$	3	2	5
Rank D	C3	$PI > 1.5 \times 10^{14}$	3	1	4
	D1	$PI \leq 5.4 \times 10^{12}$	1	2	3
Rank D	D2	$PI > 5.4 \times 10^{12}$	1	1	2

Bonus & PI Score Board

- Only for those who submit the homework before the deadline
  - 2% bonus score will be given to the best work
  - 1% bonus score will be given to the second-best work

- HW4 PI Score Board
  - https://docs.google.com/spreadsheets/d/1J3bDhFHsH7z2_NRDR2cns4zYlQ3AzJX-A_W9IAuGdBw/edit?usp=sharing

G	H	I	J	K	L	M	N
Fill your name	Do not fill these columns			Fill your result			
Name	Total Score	PI Grade	PI	Pattern Rank	A (Total cell area)	T (Cycle time)	C (Total cycles)
TA	15	A1	2.61E+15	A	2767	1.7	745,332
Example 安妮亞·佛傑	11	A3	1.20E+16	A	4000	3	1,000,000
Example 洛伊德·佛傑	7	B3	1.92E+15	B	4000	3	400,000
Example 約兒·佛傑	4	C3	4.80E+14	C	4000	3	200,000
Example 達米安·戴斯蒙德	2	D2	1.08E+13	D	4000	3	30,000

Notice

- Do not modify I/O, testbench, or patterns.
- Do not modify synthesis scripts except the two parts mentioned in the document.
- After submission, TA will use the released version testbench and patterns to run the simulation and check the correctness of your design.

Submission

- Deadline
  - 11/17 23:59
- Submit to eeclass
  - Make sure the file delivery and organization meet the requirement
  - Wrong file delivery or organization will get 1% punishment
- If you have any question, feel free to
  - Ask TA during the lab time
  - Ask on eeclass discussion
    (maybe other students also have the same question !)
  - Do not show your code on eeclass discussion or email your code to TA. Coding and debugging by yourself !