

Contents

- **My introduction & skills**
 - Working experiences (*non-quantum parts*)
 - Coding skills (*non-quantum parts*)
- **Quantum error correction code**
 - Code construction based on stabilizer formalism
- **Quantum communication**
 - Quantum walk

DUC Manh Nguyen – Introduction

nguyenmanhduc18@gmail.com

+82-010-6617-1811



Education:

- University of Ulsan (**UOU**), Korea, 02/2015-02/2020
 - *Ph.D. degree*, school of electrical engineering
 - *Thesis*: quantum error correction code & quantum communication
- Ha-Noi university of science and technology (**HUST**), Vietnam, 09/2007-06/2012
 - *Bachelor degree*, school of electronic and telecommunication
 - *Thesis*: GSM telecommunication system

Highlights:

- 4+ years experiences in system on chip (SoC) design & SW engineering.
- 5 years experiences in academic research.
- High-level language simulation for algorithm and digital signal processing.
- Hardware description language, Test-bench simulation.
- FPGA emulation, FPGA porting, FPGA prototype.

SW languages:

- Verilog HDL, Shell.
- MATLAB, C/C++, Python.

Tools:

- RTL: NC-Verilog, ModelSim, Spyglass lint.
- Synthesis: Design compiler.
- FPGA: Quartus (Altera), Xilinx ISE (Vivado).
- C/C++: Visual studio, GNU gcc.
- Python: Anaconda.
- MATLAB: MATLAB R2021a.
- Others: MAGMA for coding theory

OS:

- Window 10, Ubuntu LTS.
- Centos, Red Hat

Work experiences:

- **G2touch**, Korea, 03/2020-present
 - Touchscreen controller IC design
- Advanced network system Vietnam (**ANSV**), 06/2014-02/2015
 - Network engineer
- Samsung Vietnam for mobile R&D center (**SVMC**), 09/2012-05/2014
 - SW engineer
- Panasonic R&D center Vietnam (**PRDCV**), 01/2012-03/2012
 - Internship student.
 - 3D-Graphic software development.

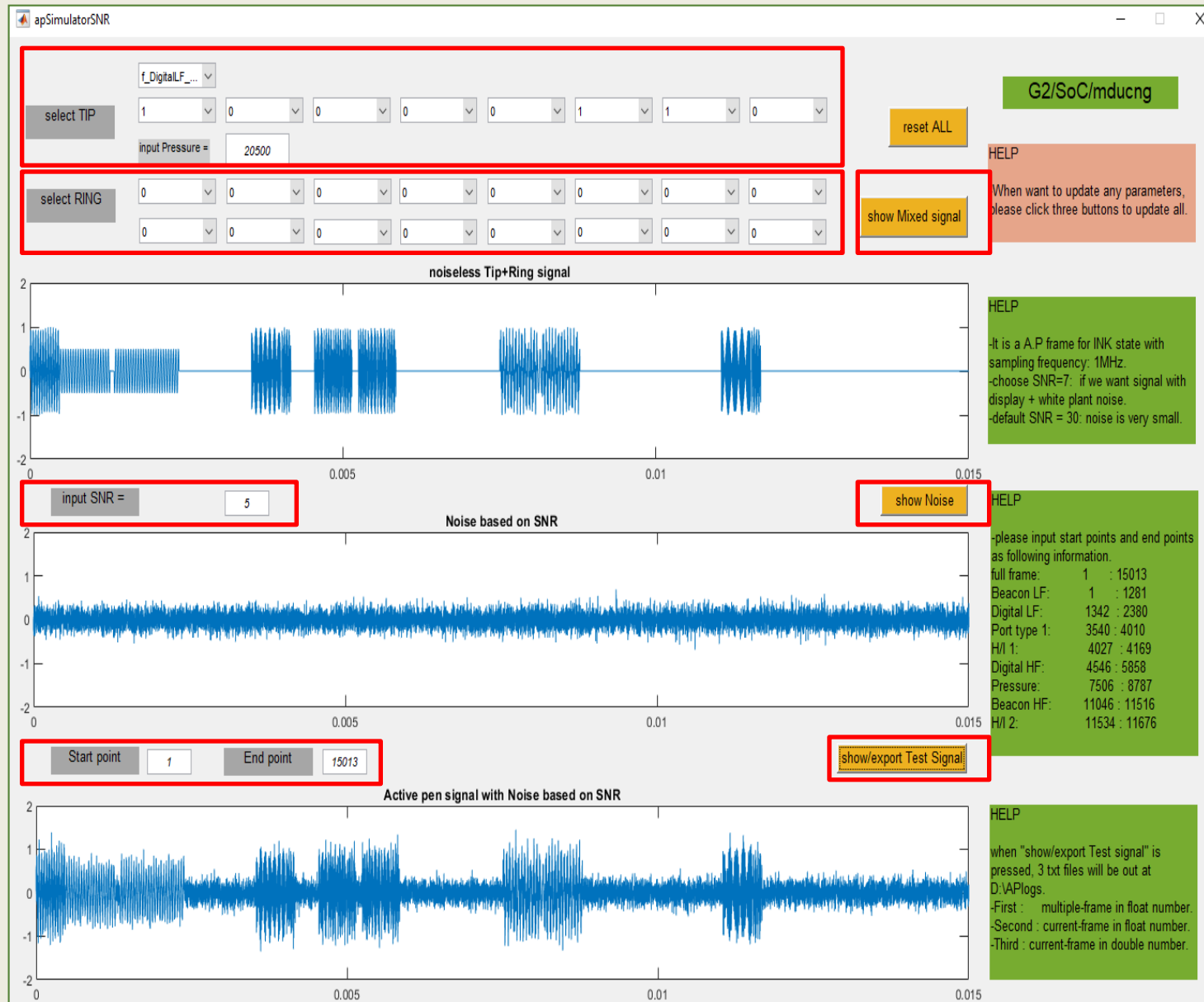
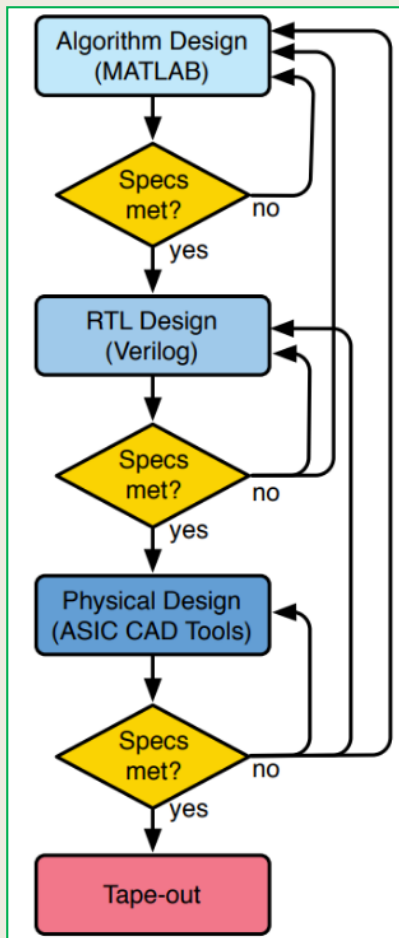
Languages:

- Vietnamese (native)
- English (fluent)
- 한국어 (중급)

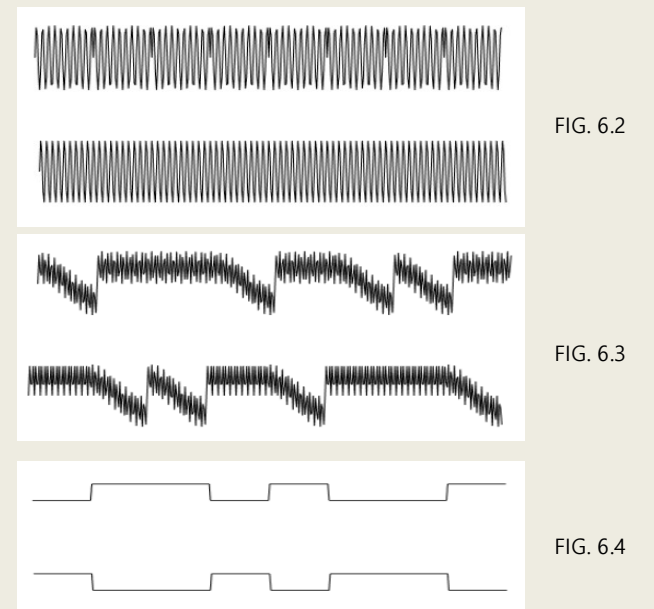
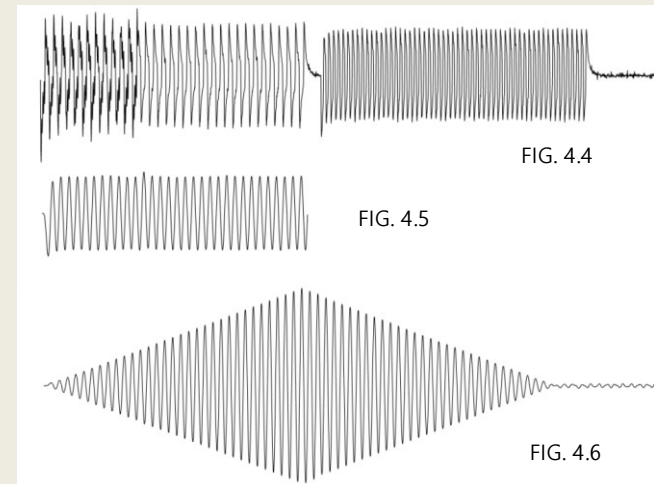
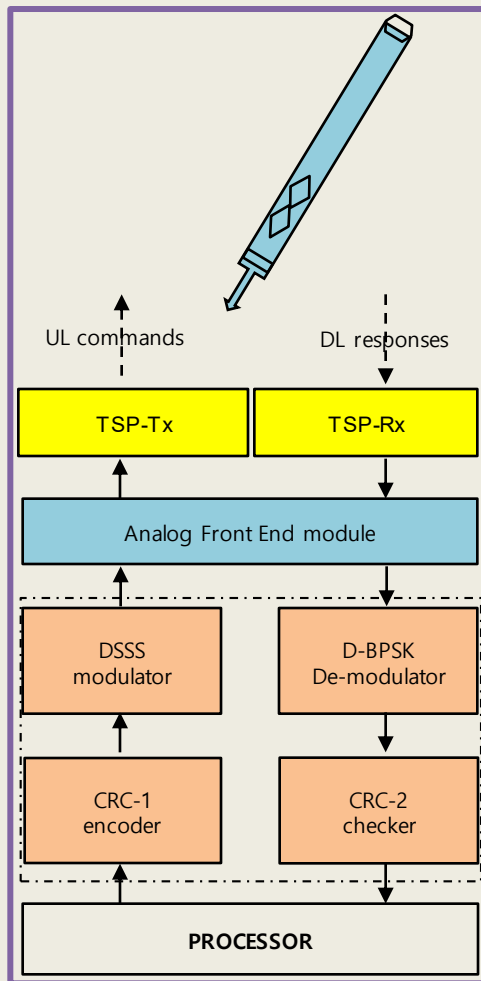
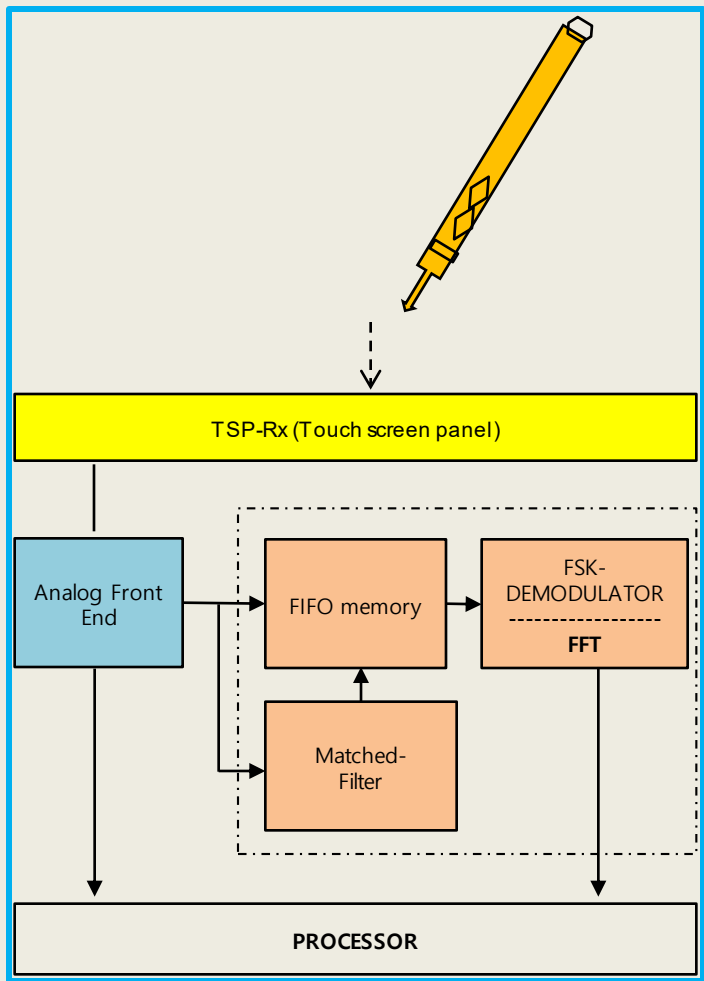
What I have done?

Project	Project descriptions	My roles	Year	Company
GL1T0AZ	. LiDAR on chip (light detection and ranging) for industrial application and self-driving car.	.My proposal for G2touch new business idea. .VCSEL driver (with DSSS sequence) and SPAD detection (with matched filter) .TCSPC: Time-correlated single photon counting.	2021 ~	G2touch
GT7M0AZ	. Touchscreen controller IC with USI (universal stylus initiative) active pen support.	. CRC (Cyclic redundancy check) calculation logic . DSSS (Direct-sequence spread spectrum) logic generator .Pen signal decoder: D-PSK de-modulator . FIR, IIR digital filters design	2020 ~	
GT3T0A	. Touchscreen controller IC with MPP (Microsoft pen protocol) active pen support.	. Matched filter design for pen detector and synchronizer .Pen signal decoder: FFT, M-FSK . SPI bus interface test-bench simulation . FPGA based FTS (Finger touch sensing) module for verification . Differential sensing scheme based on Gray-code algorithm for display noise elimination. (<i>support analog team</i>)	2019-2021	
Papers	.Researches on quantum information theory and quantum algorithm	.Quantum Error correction codes (quantum stabilizer codes) .Quantum communication: Quantum walk , quantum teleportation, quantum dense coding	2015 – 2020	UOU
.VMS phase 5	.Mobile charging service phase-V for MobiFone (telecommunication system operator).	. UNIX Operating system . Intelligent network platform : SIGTRAN, SS7,... . Mobile charging services installation	2014-2015	ANSV
.S4-mini .S5-mini	.Galaxy smartphone S4-mini .Galaxy smartphone S5-mini	.SW binary build set-up . Android software modification for Galaxy phones	2012-2014	SVMC
.eCockpit	.Emergence of a next generation UI for automotive using 3D-GFX	.Implementation a 3D model (Hachune) using ORGE 3D engine	2 months – 2012	PRDCV

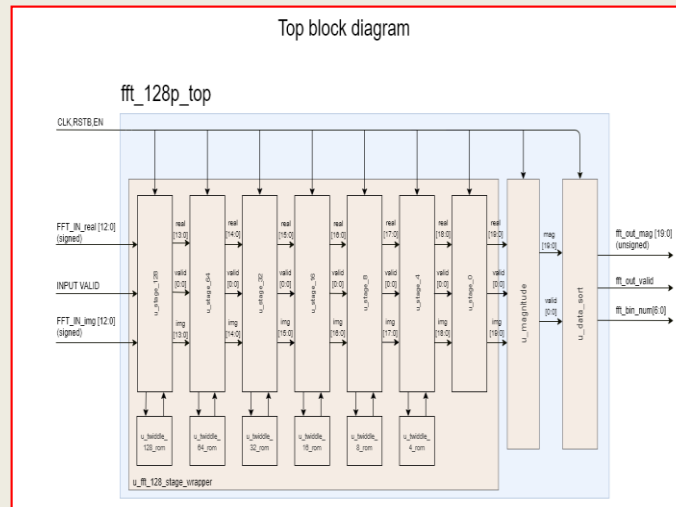
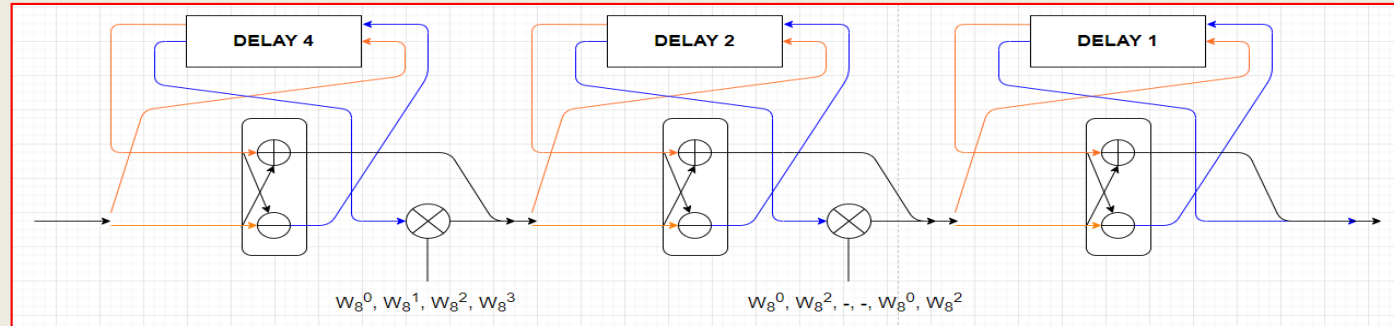
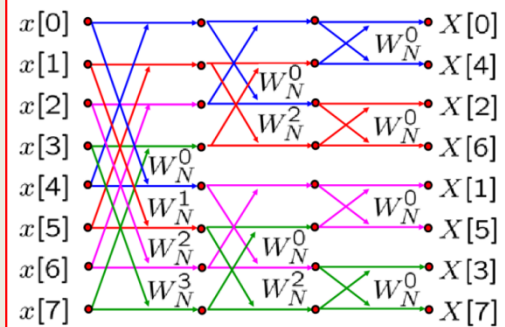
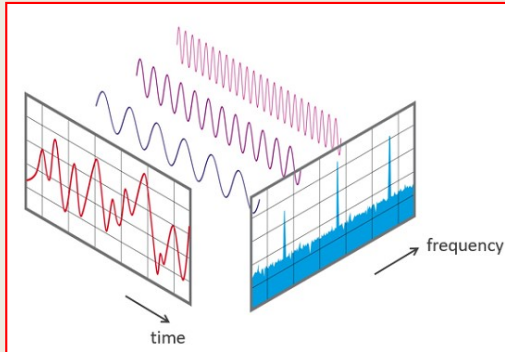
System and algorithm simulation: MATLAB



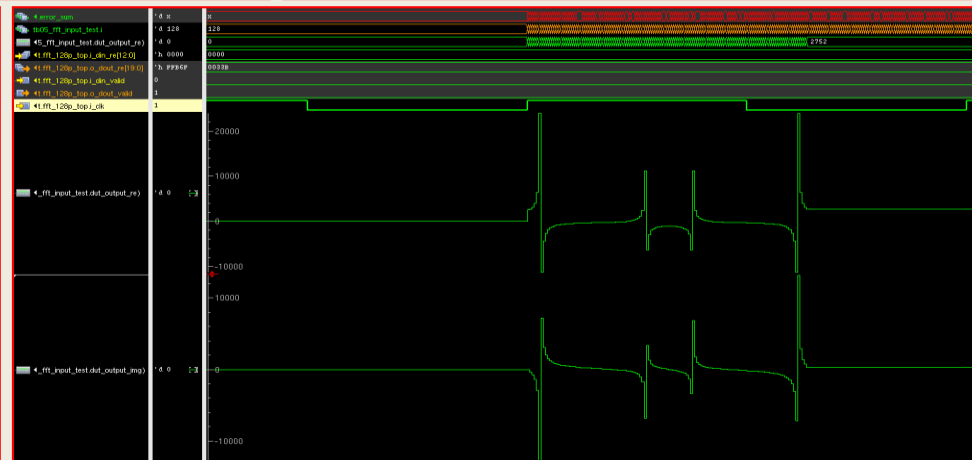
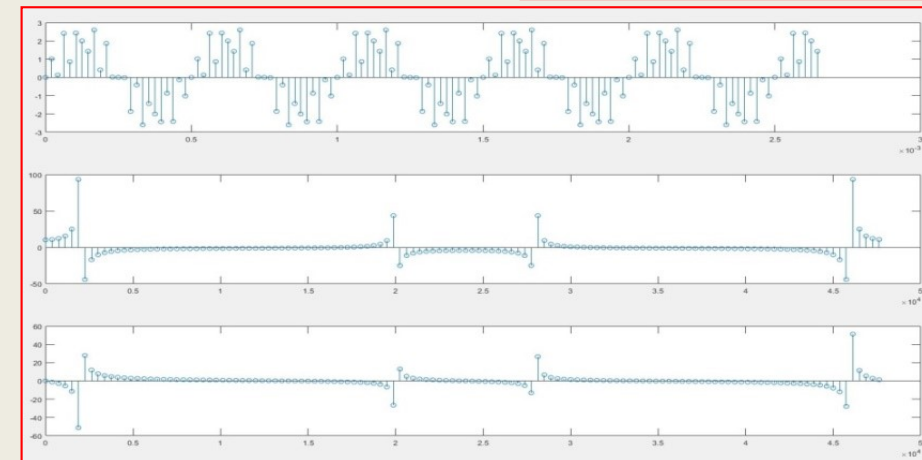
System and algorithm simulation: MATLAB



Digital system design: Verilog HDL



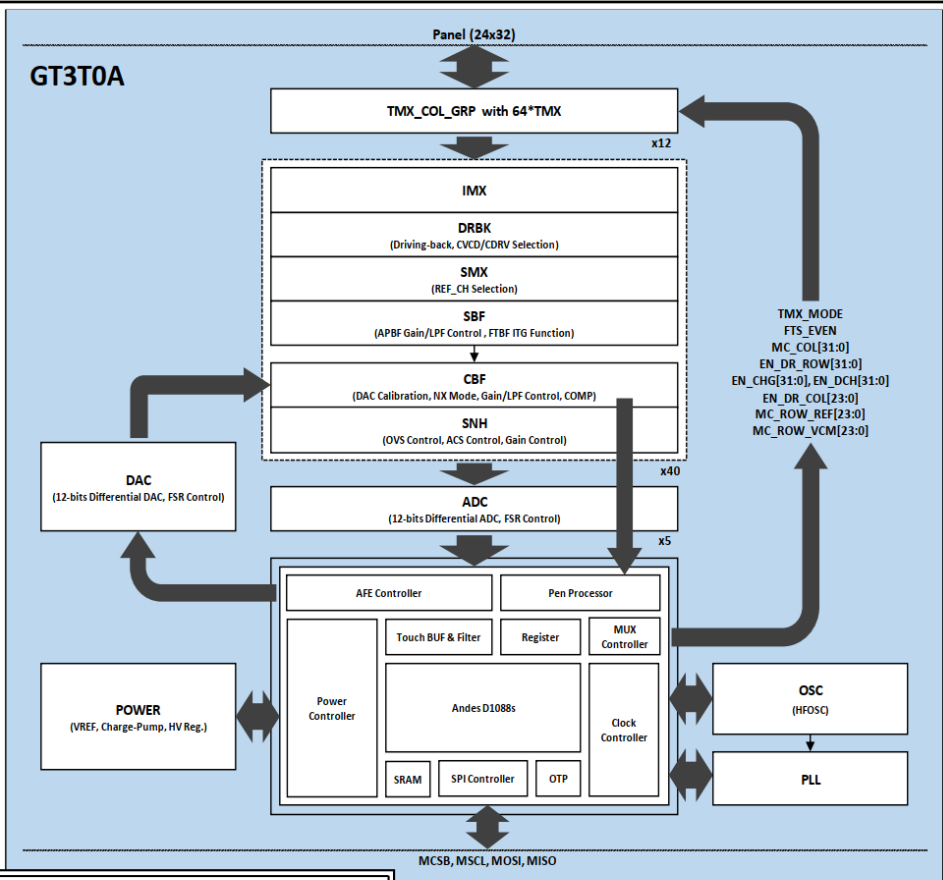
```
module fft_128p_top #(
    parameter IN_W = 13,           // input data bit width
    parameter OUT_W = 20,          // output data bit width
    parameter FFT_PT = 128,        // FFT point
    parameter FFT_PT_CNT_BIT = 7  // FFT point count bit width
) (
    input          i_rst_b,
    input          i_clk,
    input          i_en,
    input [IN_W-1:0] i_din_re,
    input [IN_W-1:0] i_din_img,
    input          i_din_valid,
    output [OUT_W-1:0] o_dout_mag,
    output          o_dout_valid,
    output [FFT_PT_CNT_BIT-1:0] o_dout_bin_num
);
```



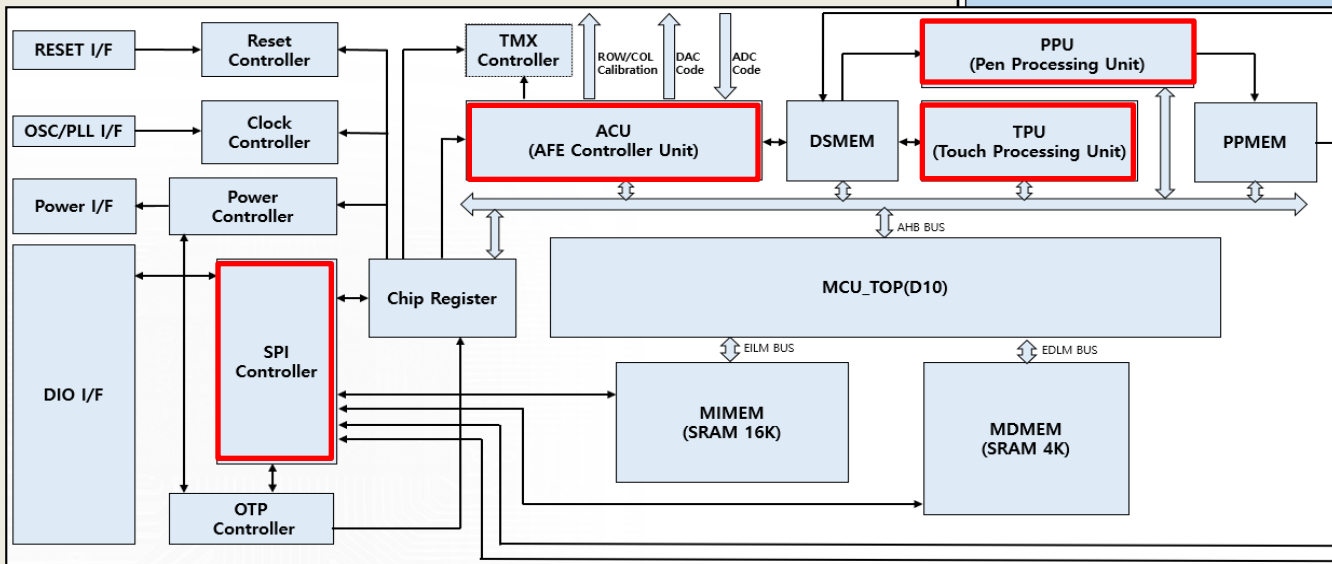
Digital system design: Verilog HDL

GT3T0A project

Mixed IC:	.Analog + Digital signal processing .Digital system based on ARM architecture (ARM Cortex-M 3)
Application	.Touch sensing IC .MPP2.01 active pen → USI (Universal Stylus Initiative) active pen
Process:	.HV18GFL20 (0.18um HV 1-7M 1.8V/3.3V/20V technology)
End-user:	.SDC, BOE



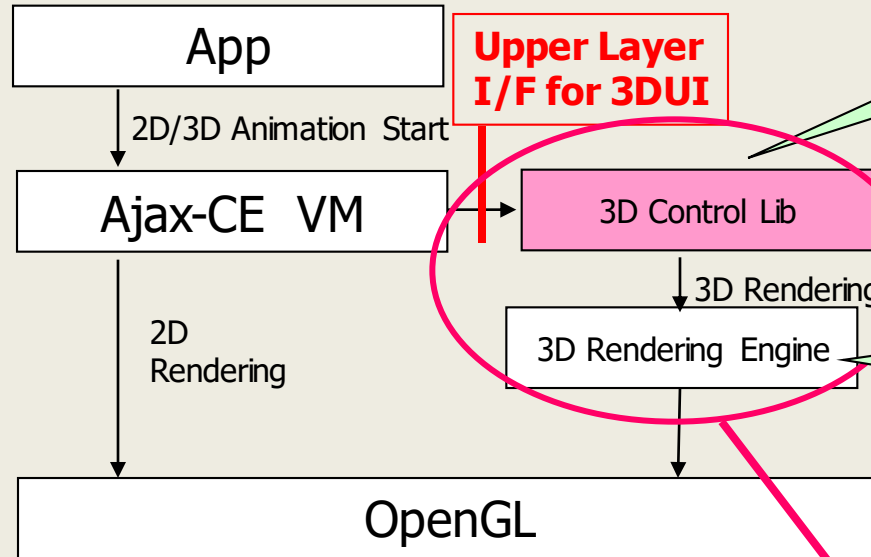
4-1. GT3T0A Block Diagram



Software development: C++



Futuristic 3D UI Design
2011CES Audi Demo



Extract the 3D Rendering Controller to Smoothly Shift to the use of HTML5

RGS, **OGRE**, Irrlicht Engine, Crystal Space, etc...

PRDCV's job !!



```
double vertices[] = {
    35.878300, 116.563200, 4.016700, // 0
    31.711100, 116.563200, 4.016700, // 1
    38.111100, 116.563200, 4.016700, // 2
    34.045500, 95.141100, -3.019600, // 3
    29.400300, 103.248500, -23.622200, // 4
}
```

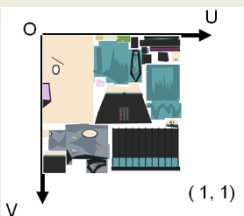
Points

```
double indices[] = {
    8, 9, 10, // 0
    11, 12, 13, // 1
    14, 15, 16, // 2
    17, 18, 19, // 3
    20, 21, 22, // 4
}
```

Triangles

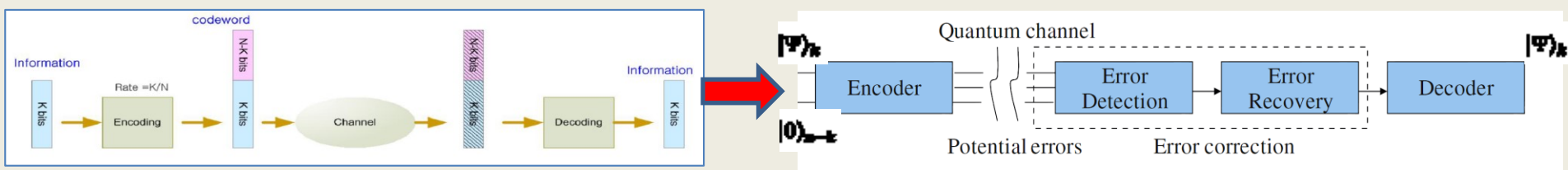
```
double UVs[] = {
    0.984840, 0.090560, // 0
    0.984840, 0.090560, // 1
    0.984840, 0.090560, // 2
    0.778350, 0.111650, // 3
    0.778350, 0.090560, // 4
}
```

Points in 2D coordinate system



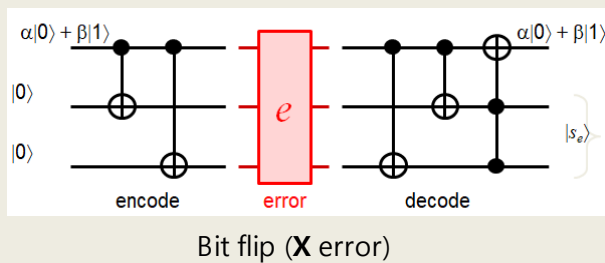
Quantum error correction code

Error correction

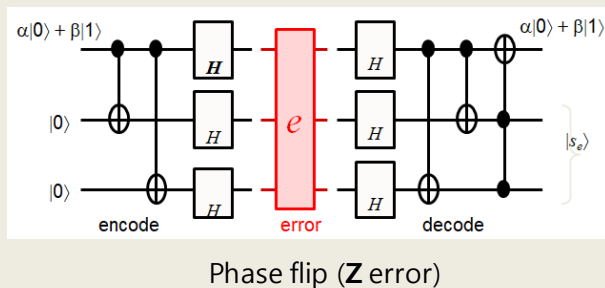


Potential errors: X, Z, Y

Repetition code

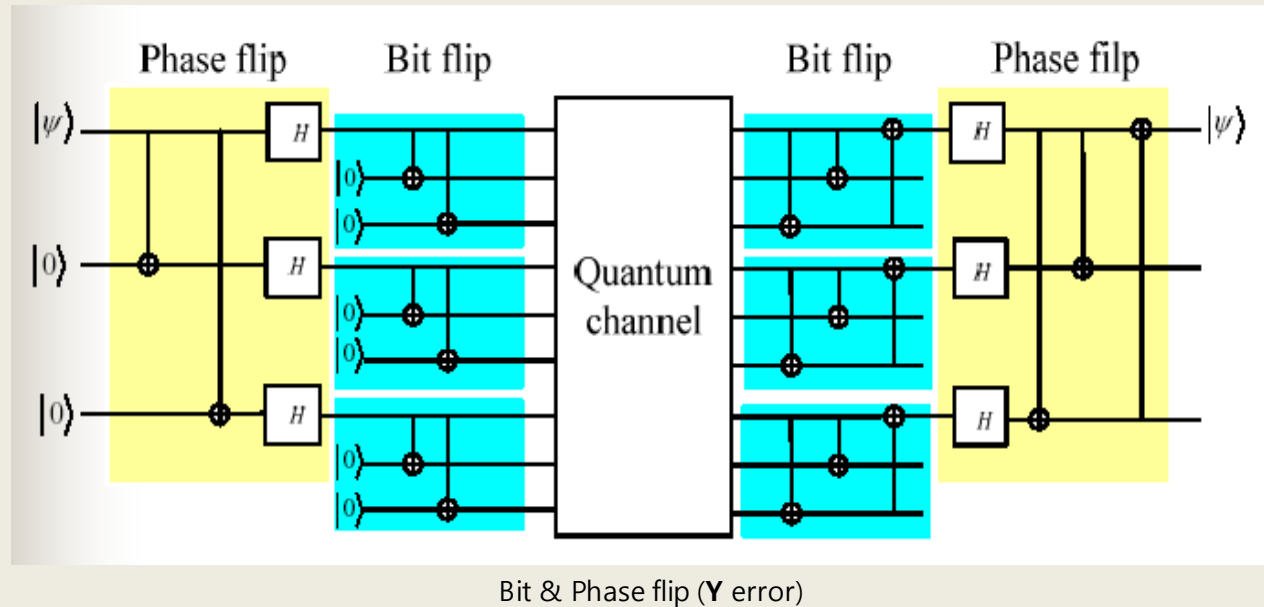


Bit flip (X error)



Phase flip (Z error)

Shor code



Bit & Phase flip (Y error)

Quantum error correction code

Framework for simulation

Based on MATLAB

```

Current Folder
+ Name
DUC-Circuit.pptx
DUC-QEC-ver-0528.pptx
BitFlip.m
BitFlip_V2.m
gate2CTE.m
gateCNE.m
hadamard.m
Identify.m
PhaseFlip.m
PhaseFlip_V2.m
qubit0.m
qubit1.m
ShorError.m
ShorError_V2.m

Editor - D:\0_Quantum information research\Qunova computing\my presentation\qec\...
+6 ShorError_V2.m gate2CTE.m gateCNE.m hadamard.m
1 function H = hadamard(n)
2 if n==1
3 H=[1 1; 1 -1]/sqrt(2);
4 else
5 H=kron(hadamard(n-1),hadamard(1));
6 end

Command Window
>> hadamard(2)

ans =

0.5000    0.5000    0.5000    0.5000
0.5000   -0.5000    0.5000   -0.5000
0.5000    0.5000   -0.5000   -0.5000
0.5000   -0.5000   -0.5000    0.5000

tem = kron(eye(2^3),-i*Ygate);
Error4 = kron(tem,eye(2^5));
tem = kron(eye(2^4),-i*Ygate);
Error5 = kron(tem,eye(2^4));
tem = kron(eye(2^5),-i*Ygate);
Error6 = kron(tem,eye(2^3));
tem = kron(eye(2^6),-i*Ygate);
Error7 = kron(tem,eye(2^2));
tem = kron(eye(2^7),-i*Ygate);
Error8 = kron(tem,eye(2^1));
Error9 = kron(eye(2^8),-i*Ygate);
%-----
%Setup qubit input:
q_zero = qubit0(1);
q_one = qubit1(1);
s_q1 = (3/5)*q_zero+(4/5)*q_one;
s_q2 = qubit0(8);
%-----
%Init
Phi1 = kron(s_q1,s_q2);
Phi2 = gate1*Phi1;
Phi3 = gate2*Phi2;
Phi4 = gate3*Phi3;
Phi5 = gate4*Phi4;
%Applying Error
Phi6 = Error0*Phi5;
%-----
Phi7 = gate5*Phi6;
Phi8 = gate6*Phi7;
Phi9 = gate7*Phi8;
Phi10 = gate8*Phi9;
Phi11 = gate9*Phi10;
Phi12 = gate10*Phi11;
%-----
s_final = [];
[m,n] = size(Phi12);

Error4 512x512 double
Error5 512x512 double
Error6 512x512 double
Error7 512x512 double
Error8 512x512 double
Error9 512x512 double
gate1 512x512 double
gate10 512x512 double
gate2 512x512 double
gate2CN 8x8 double
gate2CT 8x8 double
gate3 512x512 double
gate4 512x512 double
gate5 512x512 double
gate6 512x512 double
gate7 512x512 double
gate8 512x512 double
gate9 512x512 double
H [0.7071,0.7071;0.7071,-0.7071]
n 1
Phi1 512x1 double
Phi10 512x1 double
Phi11 512x1 double
Phi12 512x1 double
Phi2 512x1 double
Phi3 512x1 double
Phi4 512x1 double
Phi5 512x1 double
Phi6 512x1 double
Phi7 512x1 double
Phi8 512x1 double
Phi9 512x1 double
q_one [1;0]
q_zero [0;1]
s_final [0.6000;0.8000]
s_q1 [0.6000;0.8000]
s_q2 256x1 double
tem 256x256 double
tem1 64x64 double
tem2 64x64 double
Ygate [0.0000 + 0.0000i -0.0000 - 1.0000i]

```

Based on MAGMA

```

% Shor code:
F<i> := ComplexField(4);
H1 := HilbertSpace(F, 9);
f := 3/5 * H1![0,0,0,0,0,0,0,0,0] + 4/5 * H1![1,0,0,0,0,0,0,0,0];
% Encoder:
ControlledNot(~f, {1}, 4);
ControlledNot(~f, {1}, 7);
f1 := BitFlip(f, 1);
f2 := PhaseFlip(f, 1);
f := 1/SquareRoot(2)*f1 + 1/SquareRoot(2)*f2;
f1 := BitFlip(f, 4);
f2 := PhaseFlip(f, 4);
f := 1/SquareRoot(2)*f1 + 1/SquareRoot(2)*f2;
f1 := BitFlip(f, 7);
f2 := PhaseFlip(f, 7);
f := 1/SquareRoot(2)*f1 + 1/SquareRoot(2)*f2;
ControlledNot(~f, {1}, 2);
ControlledNot(~f, {1}, 3);
ControlledNot(~f, {4}, 5);
ControlledNot(~f, {4}, 6);
ControlledNot(~f, {7}, 8);
ControlledNot(~f, {7}, 8);
% Errors:
PhaseFlip(~f, 3);
BitFlip(~f, 3);
% Decoder:
ControlledNot(~f, {1}, 2);
ControlledNot(~f, {1}, 3);
ControlledNot(~f, {4}, 5);
ControlledNot(~f, {4}, 6);
ControlledNot(~f, {7}, 8);
ControlledNot(~f, {7}, 8);
ControlledNot(~f, {2,3}, 1);
ControlledNot(~f, {5,6}, 4);
ControlledNot(~f, {8,9}, 7);
f1 := BitFlip(f, 1);
f2 := PhaseFlip(f, 1);
f := 1/SquareRoot(2)*f1 + 1/SquareRoot(2)*f2;
f1 := BitFlip(f, 4);
f2 := PhaseFlip(f, 4);
f := 1/SquareRoot(2)*f1 + 1/SquareRoot(2)*f2;
f1 := BitFlip(f, 7);
f2 := PhaseFlip(f, 7);
f := 1/SquareRoot(2)*f1 + 1/SquareRoot(2)*f2;
ControlledNot(~f, {1}, 4);
ControlledNot(~f, {1}, 7);
ControlledNot(~f, {4,7}, 1);
% Output:
f;
%http://magma.maths.usyd.edu.au/calc/
--> 0.5999|001100100> + 0.8000|101100100>

```

Quantum error correction code

■ Stabilizer + Coding theory → Quantum stabilizer codes

○ Representing a state as its **group of stabilizers**

1. I stabilizes everything.

2. $-I$ stabilizes nothing.

3. X stabilizes $|+\rangle$: $X|+\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = |+\rangle$

4. $-X$ stabilizes $|-\rangle$: $-X|-\rangle = -\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix} = |-\rangle$

5. Y stabilizes $|+i\rangle$: $Y|+i\rangle = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} \end{bmatrix} = |+i\rangle$

6. $-Y$ stabilizes $|-i\rangle$: $-Y|-i\rangle = -\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{i}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{i}{\sqrt{2}} \end{bmatrix} = |-i\rangle$

7. Z stabilizes $|0\rangle$: $Z|0\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$

8. $-Z$ stabilizes $|1\rangle$: $-Z|1\rangle = -\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$

Critical result from group theory: for any N-qubit stabilized state, only N elements needed to specify group.

1. $|0\rangle$ is stabilized by $\{I, Z\}$, Z is generator

2. $|1\rangle$ is stabilized by $\{I, -Z\}$, $-Z$ is generator

3. $|+\rangle$ is stabilized by $\{I, X\}$, X is generator

4. $|-\rangle$ is stabilized by $\{I, -X\}$, $-X$ is generator

5. $|+i\rangle$ is stabilized by $\{I, Y\}$, Y is generator

6. $|-i\rangle$ is stabilized by $\{I, -Y\}$, $-Y$ is generator

7. $|0\rangle \otimes |0\rangle$ is stabilized by $\{I \otimes I, I \otimes Z, Z \otimes I, Z \otimes Z\}$, $\{I \otimes Z, Z \otimes I\}$ is generator

8. $|+\rangle \otimes |0\rangle$ is stabilized by $\{I \otimes I, I \otimes Z, X \otimes I, X \otimes Z\}$, $\{I \otimes Z, X \otimes I\}$ is generator

9. $\frac{|00\rangle + |11\rangle}{\sqrt{2}}$ is stabilized by $\{I \otimes I, X \otimes X, -Y \otimes Y, X \otimes Z\}$, $\{X \otimes X, Z \otimes Z\}$ is generator

$$\mathbf{E}|\Psi\rangle = \mathbf{E}\mathbf{M}_i|\Psi\rangle = \begin{cases} \mathbf{M}_i\mathbf{E}|\Psi\rangle & \text{Error undetected} \\ -\mathbf{M}_i\mathbf{E}|\Psi\rangle & \text{Error detected} \end{cases}$$

$[[n, k, d]]$

$$\sim \begin{pmatrix} Z & Z & X & I & X \\ X & Z & Z & X & I \\ I & X & Z & Z & X \\ X & I & X & Z & Z \end{pmatrix}$$

e.g. $H = \left(\begin{array}{ccccc|ccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \end{array} \right)$

Pauli operator	$GF(4)$ elements
I	0
X	1
Y	$\omega^2 = \omega + 1$
Z	ω
Multiplication operator	Addition
Commutative	Trace-inner product

Symplectic inner product

$$\mathbf{M}_i\mathbf{M}_j = \mathbf{M}_j\mathbf{M}_i$$

$$H(i, :) \odot H(j, :) = 0$$

$$\mathbf{H}_X \cdot \mathbf{H}_Z^T + \mathbf{H}_Z \cdot \mathbf{H}_X^T = 0$$

Quantum error correction code

My main contribution:

symplectic product $(u, v) \odot (\pi^x(u), \pi^{-x}(v))$ is zero

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$\left\{ \begin{array}{l} \begin{bmatrix} \hat{\mathbf{I}} & \hat{\mathbf{A}}_1 & \hat{\mathbf{A}}_2 & \hat{\mathbf{B}} & \hat{\mathbf{C}}_1 & \hat{\mathbf{C}}_2 \\ 0 & 0 & 0 & \mathbf{D} & \mathbf{I} & \mathbf{E} \end{bmatrix} \end{array} \right\} \begin{array}{l} r \\ n-k-r \end{array}$$

$$\begin{cases} \bar{\mathbf{X}} = \begin{bmatrix} 0 & \mathbf{E}^T & \mathbf{I} & (\mathbf{E}^T \mathbf{C}_1 + \mathbf{C}_2^T) & 0 & 0 \end{bmatrix} \\ \bar{\mathbf{Z}} = \begin{bmatrix} 0 & 0 & 0 & \mathbf{A}_2^T & 0 & \mathbf{I} \end{bmatrix} \end{cases}$$

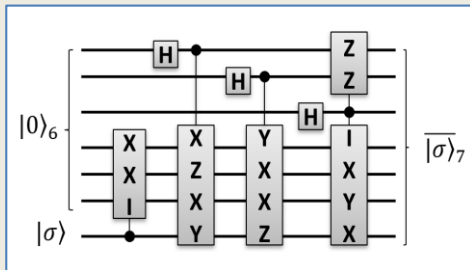
$$|\overline{c_1 c_2 \dots c_k}\rangle = \frac{1}{\sqrt{2^m}} \times \left(\prod_{i=1}^m (\mathbf{I} + \mathbf{g}_i) \right) \times \bar{\mathbf{X}}_1^{c_1} \times \bar{\mathbf{X}}_2^{c_2} \times \dots \times \bar{\mathbf{X}}_k^{c_k} |00\dots 0\rangle_n,$$

$$\begin{aligned} |\overline{\sigma_1 \sigma_2 \dots \sigma_k}\rangle &= \frac{1}{\sqrt{2^{n-k}}} \times \left(\prod_{i=1}^{n-k} (\mathbf{I} + \mathbf{g}_i) \right) \times \bar{\mathbf{X}}_1^{\sigma_1} \times \bar{\mathbf{X}}_2^{\sigma_2} \times \dots \times \bar{\mathbf{X}}_k^{\sigma_k} |00\dots 0\rangle_n \\ &= \left(\prod_{i=1}^r T_i^{\sigma_i} \mathbf{H}_i \right) \times \left(\prod_{i=1}^k \mathbf{U}_i \right) |00\dots 0 \sigma_1 \sigma_2 \dots \sigma_k\rangle_n. \end{aligned}$$

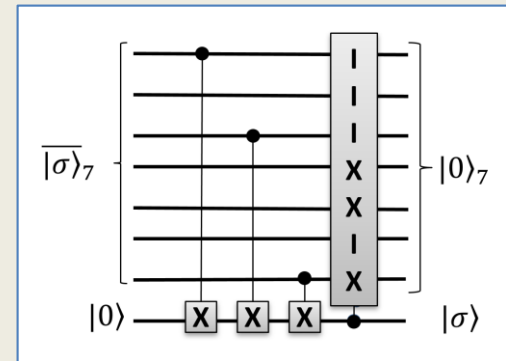
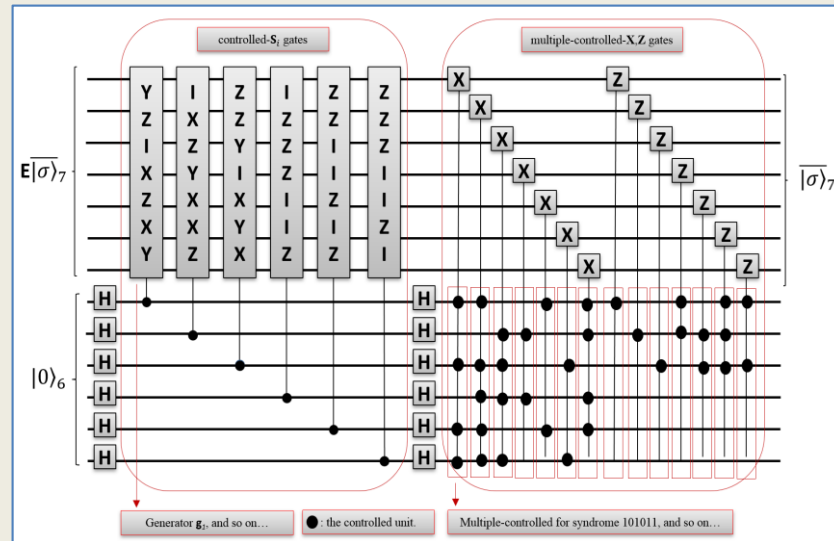
$$\mathbf{g}_i |\psi\rangle = (-1)^{l_i} |\psi\rangle \Leftrightarrow \mathbf{g}_i \mathbf{E} = (-1)^{l_i} \mathbf{E} \mathbf{g}_i$$

$$|\psi_{in}\rangle = |\overline{\sigma_1 \sigma_2 \dots \sigma_k}\rangle_n \otimes |00\dots 0\rangle_k$$

$$|\psi_f\rangle = \mathbf{U}_{\text{decode}} |\overline{\sigma_1 \sigma_2 \dots \sigma_k}\rangle_n \otimes |00\dots 0\rangle_k = |\overline{00\dots 0}\rangle_n \otimes |\sigma_1 \sigma_2 \dots \sigma_k\rangle_k.$$



E	Syndrome	E	Syndrome	E	Syndrome
XXXXXX	101011	ZXXXXX	100000	YXXXXX	001011
XXXXXX	101111	IZXXXX	010000	IYXXXX	111111
XXXXXX	011101	IIZXXX	001000	IIYXXX	010101
XXXXXX	010100	IIIZXX	110000	IIYYXX	100100
XXXXXX	100010	IIIZX	011000	IIYYX	111010
XXXXXX	001001	IIIZ	111000	IIYY	110001
XXXXXX	110110	IIIZ	101000	IIYY	011110



Quantum communication: Quantum walk

A discrete-time QWs over the line involves in two Hilbert spaces. The total space of the walk is given as

$$H = H_P \otimes H_C, \quad (1)$$

where H_P is the position space, which is spanned by $\{|n\rangle$ where $n \in \mathbb{Z}\}$, and H_C is the coin space, which is spanned by $\{|0\rangle, |1\rangle\}$.

$$\begin{aligned} \mathbf{s}^T &= \left[\sum_{n=-\infty}^{+\infty} |n+1\rangle\langle n| \otimes |0\rangle\langle 0| + \sum_{n=-\infty}^{+\infty} |n-1\rangle\langle n| \otimes |1\rangle\langle 1| \right]^T \\ &= \sum_{n=-\infty}^{+\infty} |n\rangle\langle n+1| \otimes |0\rangle\langle 0| + \sum_{n=-\infty}^{+\infty} |n\rangle\langle n-1| \otimes |1\rangle\langle 1|. \end{aligned}$$

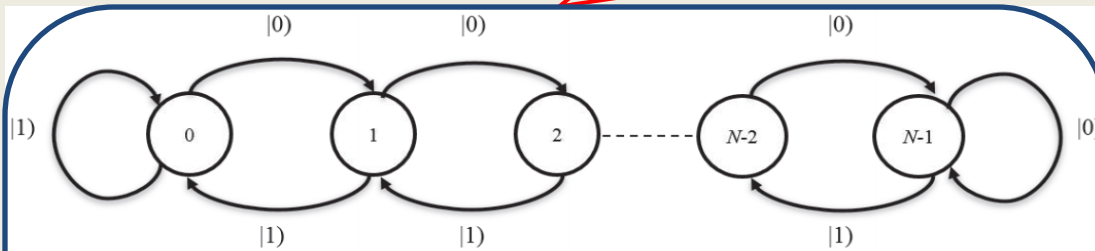


Fig. 2. The QW over the N -line.

$$\begin{aligned} \mathbf{S}_1 &= \sum_{t=0}^{N-2} |t+1\rangle\langle t| \otimes |0\rangle\langle 0| + |N-1\rangle\langle N-1| \otimes |0\rangle\langle 0| \\ &+ \sum_{t=1}^{N-1} |t-1\rangle\langle t| \otimes |1\rangle\langle 1| + |0\rangle\langle 0| \otimes |1\rangle\langle 1|. \end{aligned}$$

$$\mathbf{S}_1 * \mathbf{S}_1^T = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \cdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix} \neq \mathbf{I}.$$

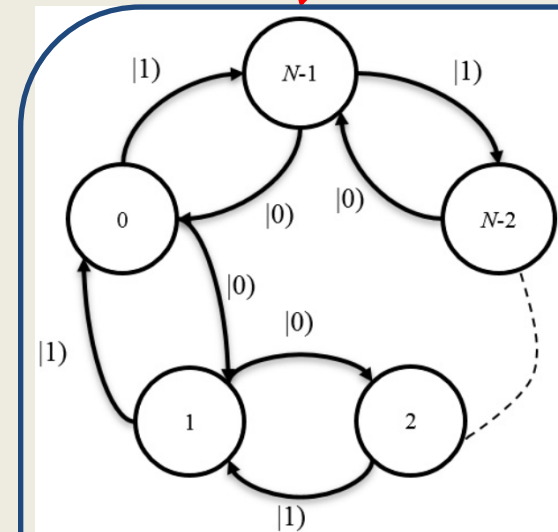
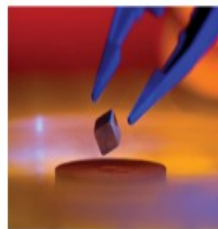


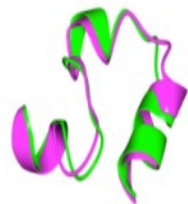
Fig. 3. The QW over N -cycle.

$$\begin{aligned} \mathbf{S}_2 &= \sum_{t=0}^{N-2} |t \oplus 1\rangle\langle t| \otimes |0\rangle\langle 0| + \sum_{t=1}^{N-1} |t \ominus 1\rangle\langle t| \otimes |1\rangle\langle 1|, \\ \mathbf{S}_2 * \mathbf{S}_2^\dagger &= \mathbf{S}_2 * \mathbf{S}_2^T = \mathbf{I}, \\ \mathbf{S}_2^\dagger * \mathbf{S}_2 &= \mathbf{S}_2^T * \mathbf{S}_2 = \mathbf{I}. \end{aligned}$$

Plan for working in Qunova computing



Energy
Room-temperature
superconductivity



Health
Quantum chemistry



Internet Security

A computation is a physical process. It may be performed by a piece of electronics or on an abacus, or in your brain, but it is a process that takes place in nature and as such it is subject to the laws of physics.

Quantum computers are machines that rely on characteristically **quantum phenomena**, such as **quantum interference** and **quantum entanglement** in order to perform computation.

– Artur Ekert

Overarching goal

Solve intractable problems with **massive speedup** in computation...
...using the **superposition** and **entanglement**, two of the cornerstones quantum mechanics

Drug discovery process

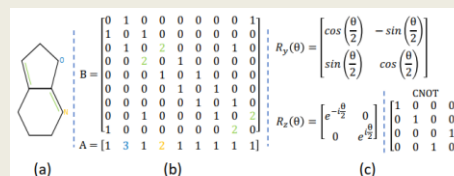
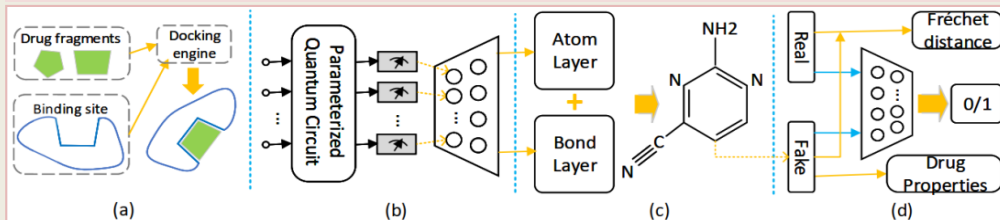


FIGURE 2. (a-b) A sample molecular graph from QM9 denoted by its corresponding atom vector **A** and bond matrix **B**; (c) all quantum gates used in this study.

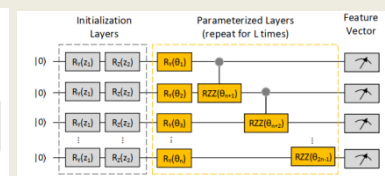
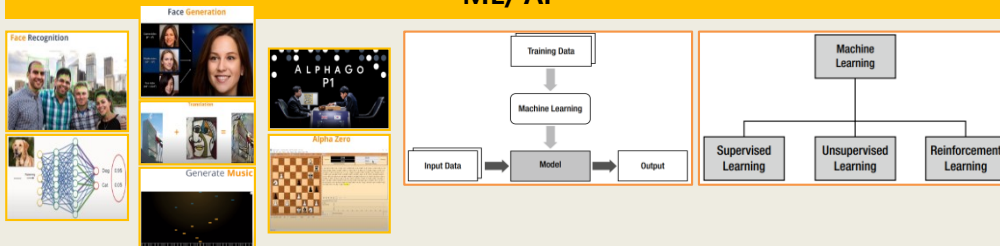


FIGURE 3. Parameterized quantum circuit to obtain feature vector of N dimensions. The circuit is composed of initialization layers, repeatable parameterized layers and measurement layer. Each RZZ gate is composed of two CNOT gates and one parametric RZ gate.

ML/ AI



-Using quantum computing with **parameterized circuit** to boost and generate **novel drug patterns**.

Quantum computing

