Introduction to Quantum Computing

Report *Qubit code / measurements*

Maria Dima
University of Bucharest – Romania

Contents

Contents

Ondulatory behaviour

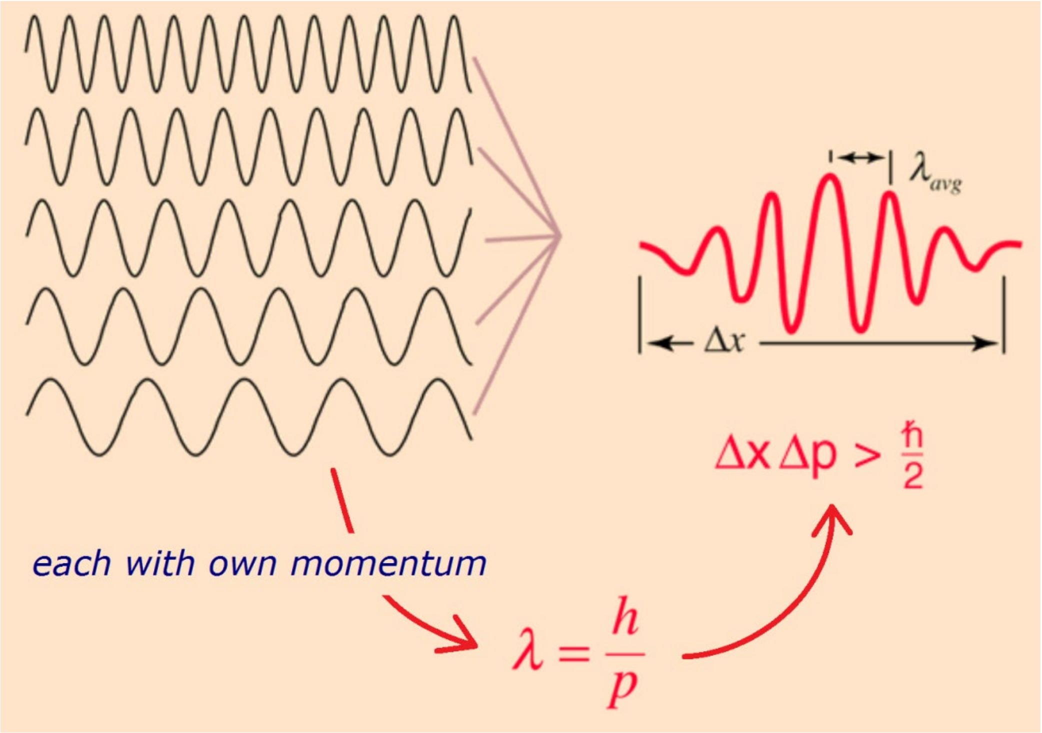
particles have wavelength : $\lambda = h / p$

… and a wavefunction : $|\psi\rangle$ = *Hilbert-space vec*

overlap of state ϕ *onto* ψ *:*

prob% $= |\langle \phi | \psi \rangle|^2$

Superposition of states



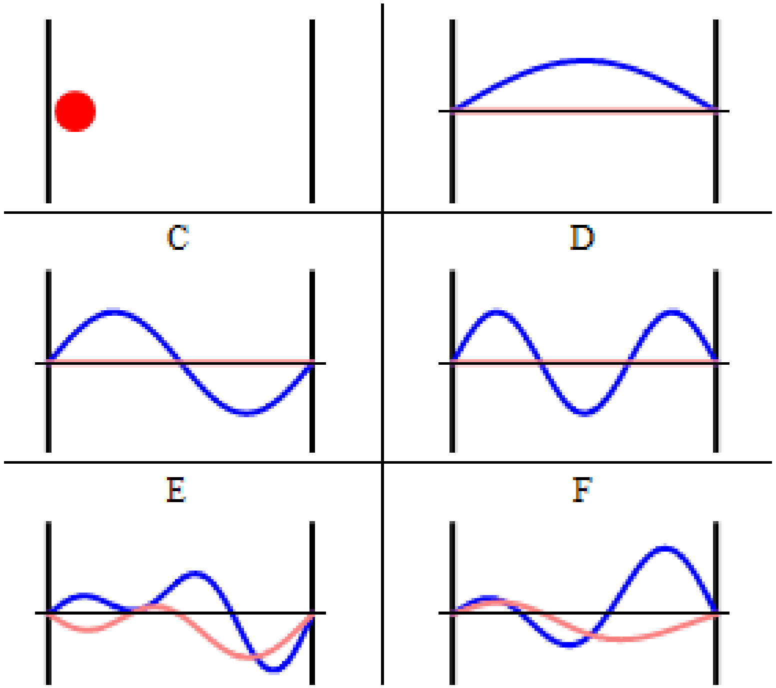
- *of uncertain momentum and location*

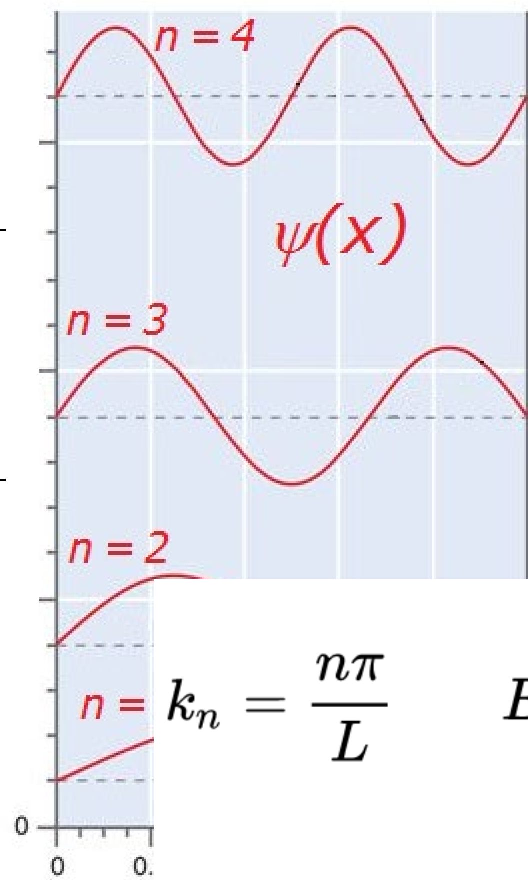
- *Heisenberg uncertainty*

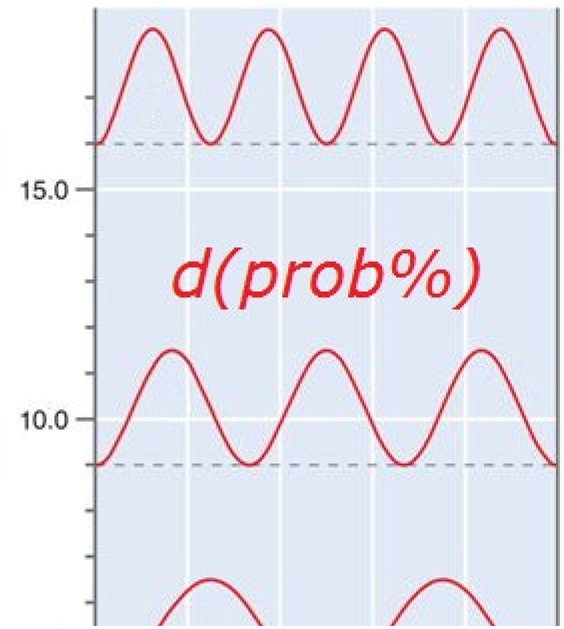
Quantisation

Schrödinger equation

$$i\hbar\frac{\partial}{\partial t}\psi(x,t) = -\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2}\psi(x,t) + V(x)\psi(x,t)$$







$$k_n = \frac{n\pi}{L} \qquad E_n = \hbar\omega_n = n^2\frac{\pi^2\hbar^2}{2mL^2}$$

E_0

Stern-Gerlach experiment

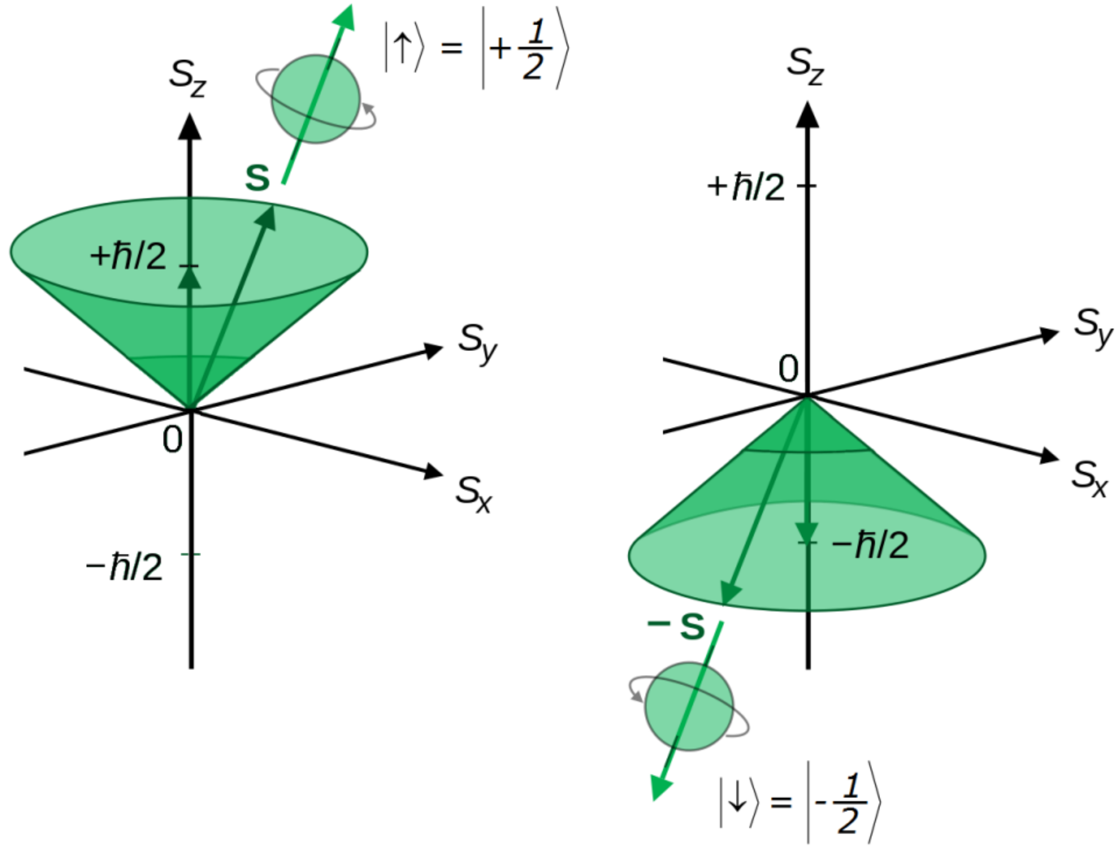
- *electron has intrinsic spin*
- *that is quantised ↑ or ↓*

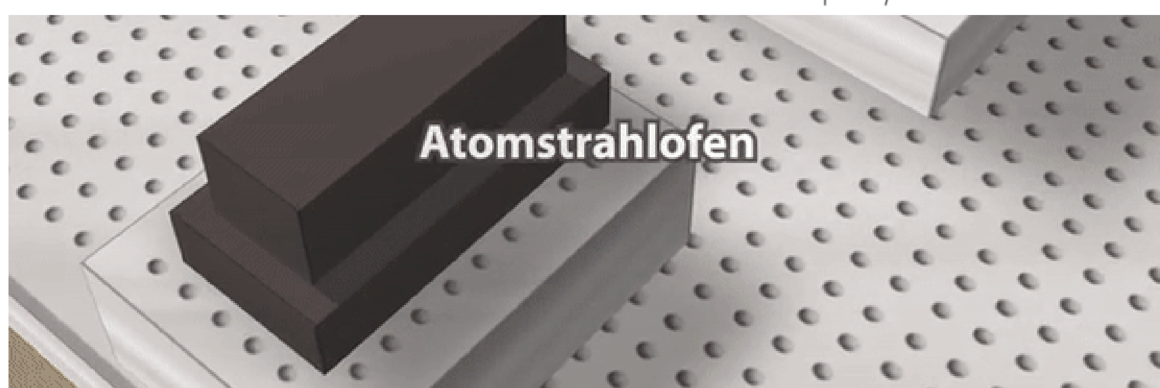
$$H = -\vec{\boldsymbol{\mu}} \cdot \vec{\mathbf{B}} = -\mu\vec{\boldsymbol{\sigma}} \cdot \vec{\mathbf{B}}$$

$$\vec{\sigma} \times \vec{\sigma} \;=\; 2i\,\vec{\sigma}$$

- $|\leftarrow\rangle + |\rightarrow\rangle = \sqrt{2}\,|\uparrow\rangle$

  pure state in one base is superposition in another

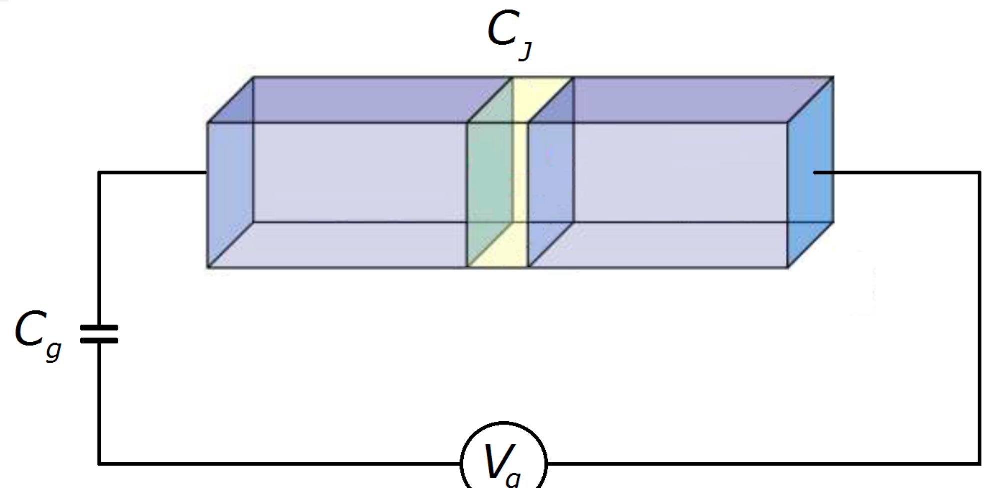




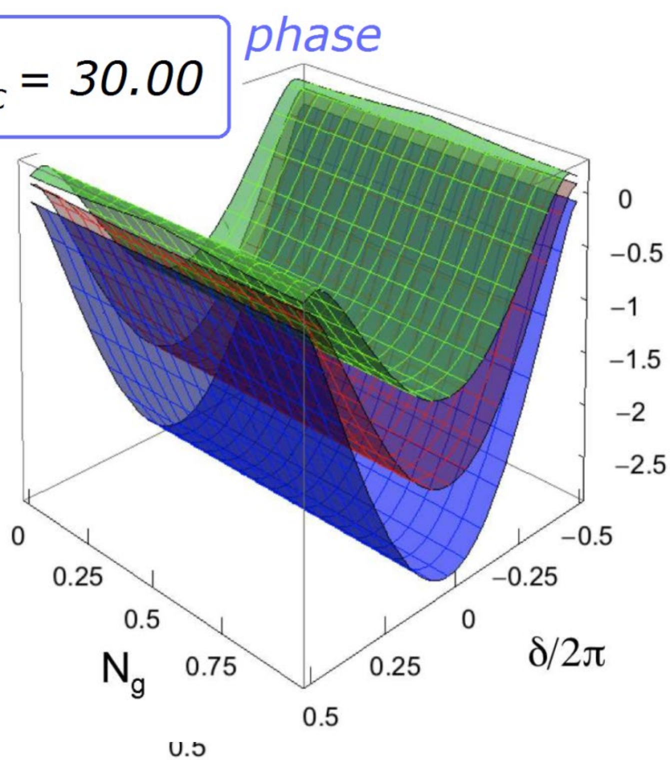
Cooper-pair Box

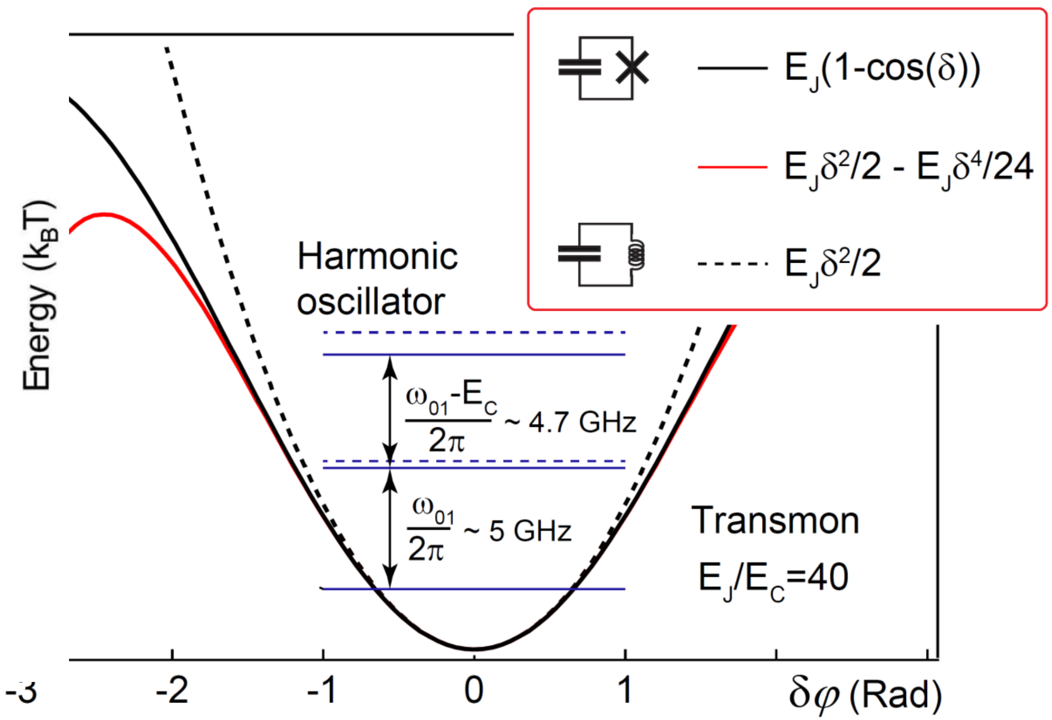
- *2 superconductors*
- *ca. 1 nm insulator*

$$H = 4E_C(n - n_g)^2 - E_J \cos\varphi$$



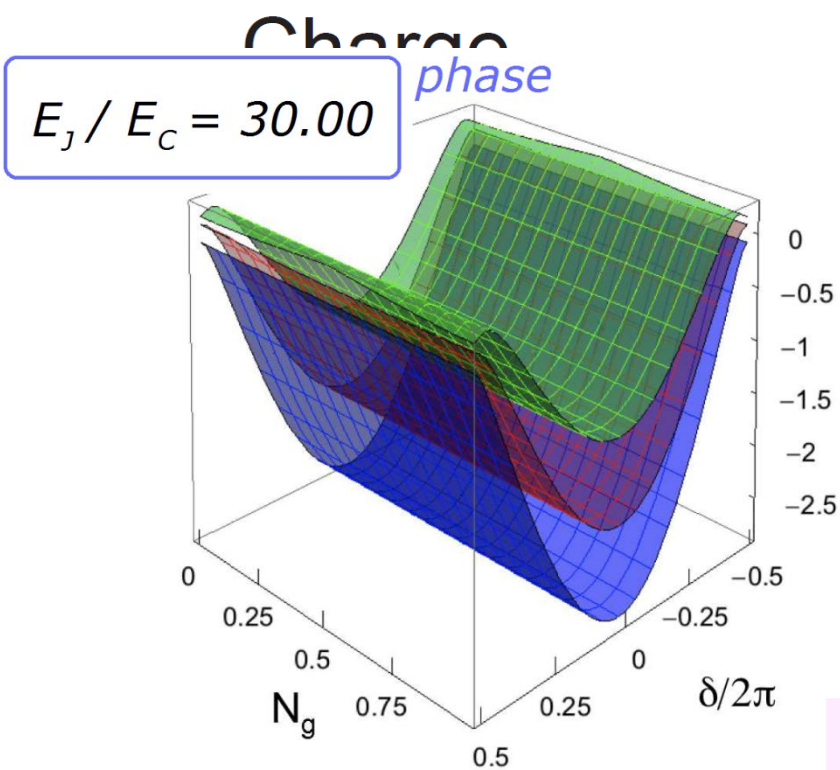
$E_J / E_C = 30.00$

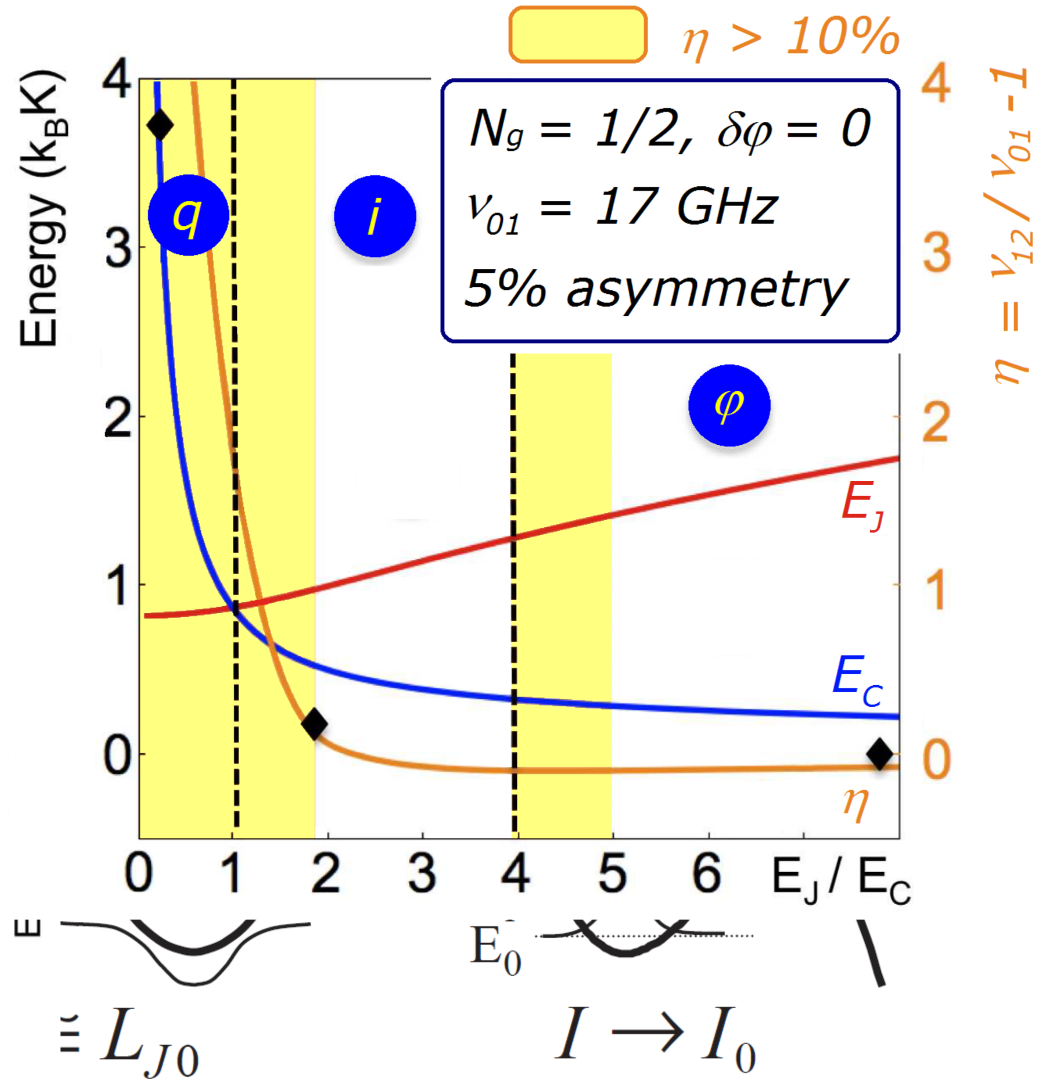




Transmon qubit

- *anharmonicity engineered*
- *immune to V_g variations*
- *phase-state qubit*



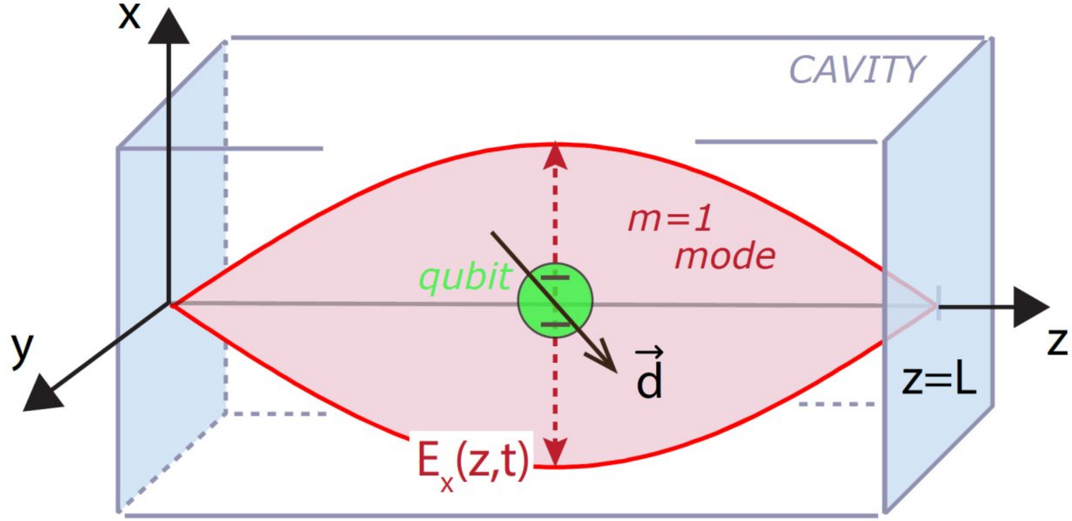


transm.-line shunted plasma oscillation qubit

Interaction w/ qubits

Microwave cavity

- fundamental mode
- interaction w/ qubit dipole



$$H_{int} = -d \cdot E_x$$

$$= -d_x \mathcal{E}_0 \, (\hat{a} + \hat{a}^\dagger)(\sigma_+ + \sigma_-)$$

DRESSED states

ground state

excited

cavity

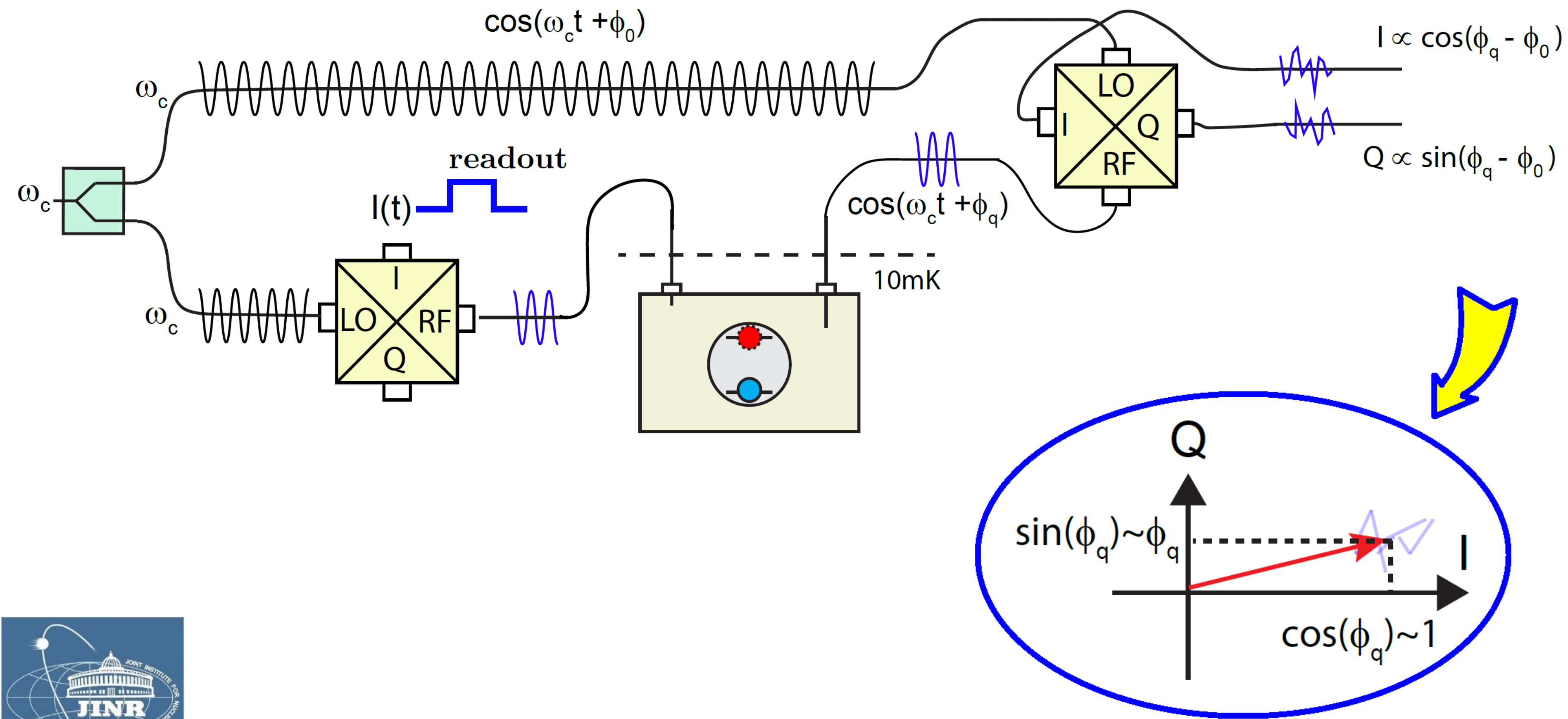
$$|0,-\rangle \;=\; |g,0\rangle$$

$$|n,-\rangle \;=\; \cos(\theta_n)|g,n+1\rangle - \sin(\theta_n)|e,n\rangle$$

$$|n,+\rangle \;=\; \sin(\theta_n)|g,n+1\rangle + \cos(\theta_n)|e,n\rangle$$

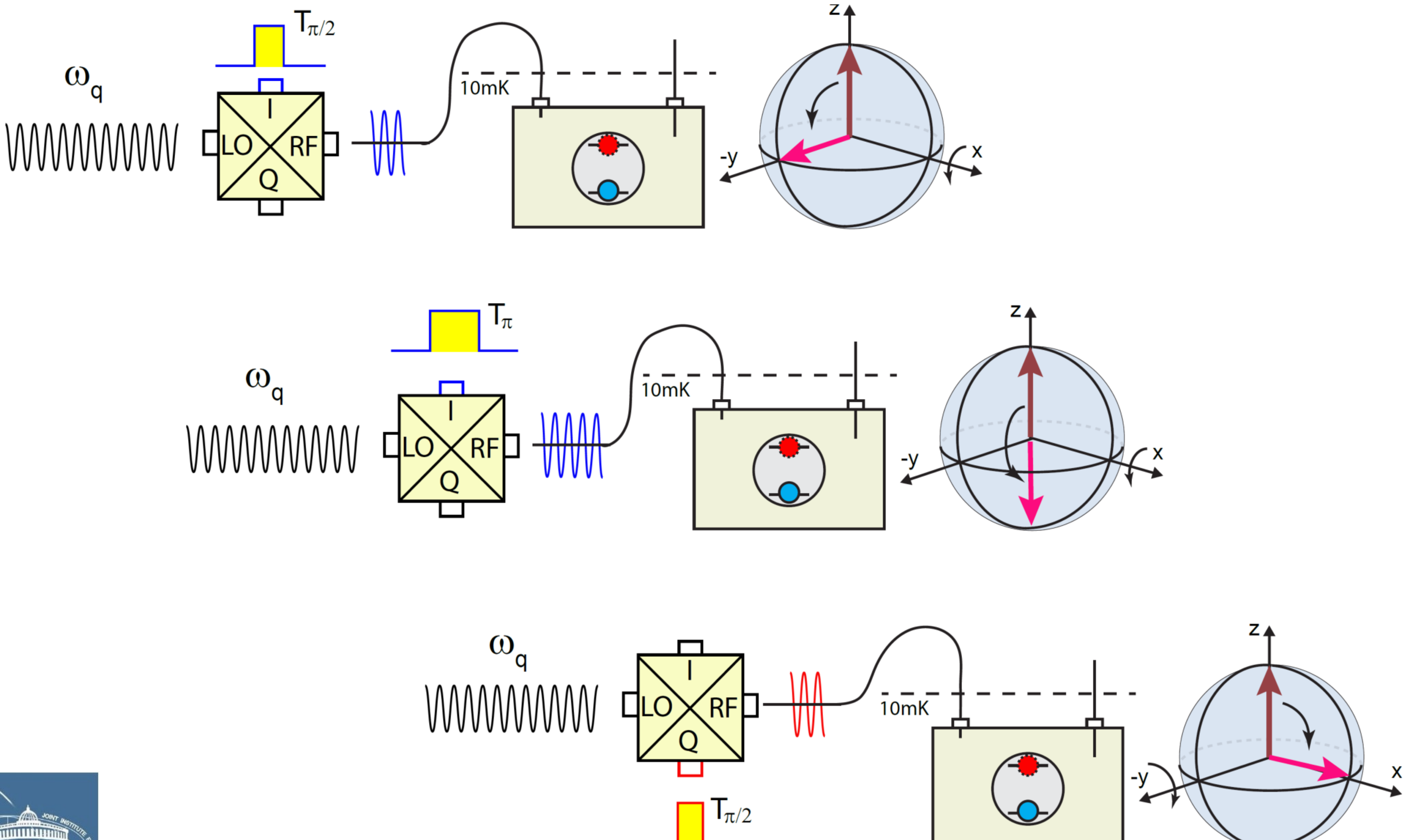
Readout pulse

- *homodyne measurement*
- *dressed-state frequency*



Qubit manipulation

Manipulation pulses

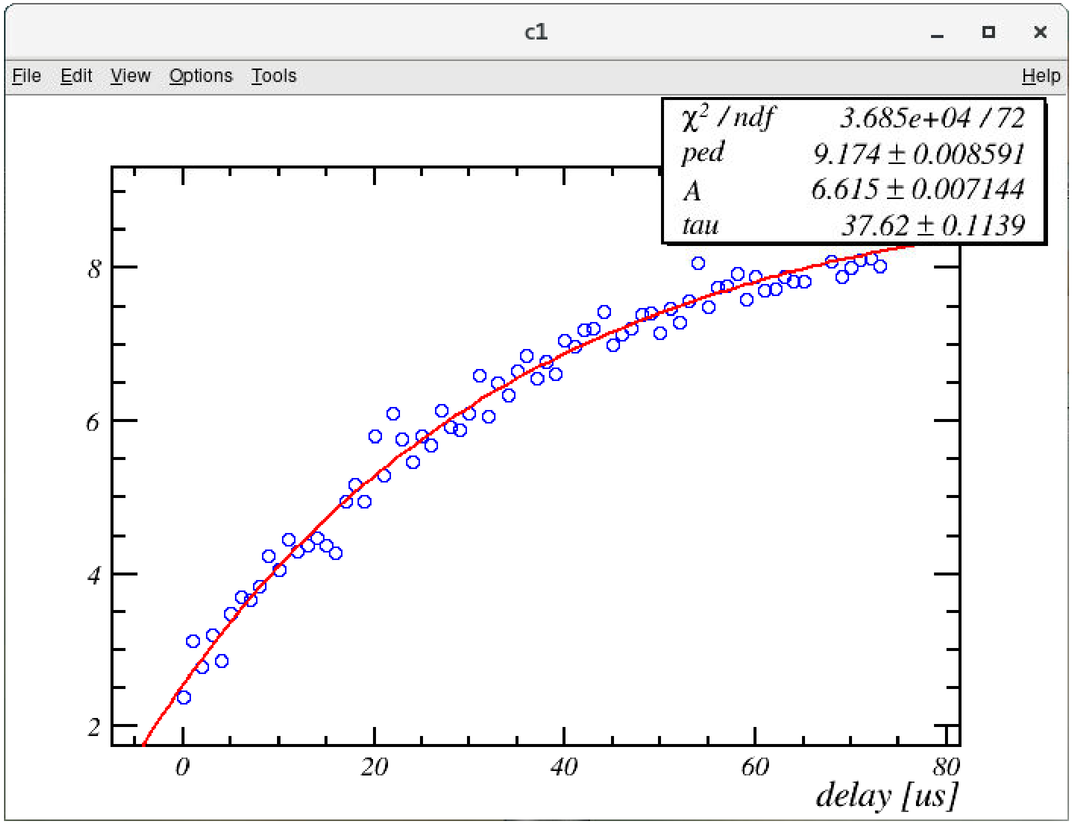


Contents

ROOT package

- *I downloaded from CERN the ROOT-5.34 (Windows)*
- *I learned how to write my own macro and do fits*

```
// ____________________ ROOT FITS ______________

void myfit() {

// TGraph gr  ("data.txt", "%lg %lg");
// TGraph grr ("test.txt", "%lg %*lg %lg")
// TGraph grrr("test.txt", "%lg %*lg %*lg %lg")

gStyle->SetOptFit   (1)
gStyle->SetLineWidth(2)


TGraphErrors* gr = new TGraphErrors("z4.txt")

Int_t N = gr->GetN()
Double_t x,y
  for (Int_t i=0; i<N; i++) {
    gr->GetPoint      (i,        x,       y)
    gr->SetPointError(i,     0.01,    0.01)
    gr->SetPoint      (i,   x/1.0,       y)


TF1 fit("fit", "([0]-[1]*exp(-x/[2]))", 0, 74)

    fit.SetParName  (0, "ped"  )
    fit.SetParName  (1, "A"    )
    fit.SetParName  (2, "tau"  )

    fit.SetParameter(0, 11.500 )
    fit.SetParameter(1,  9.500 )
    fit.SetParameter(2, 55.400 )

  gr->Fit("fit")
```

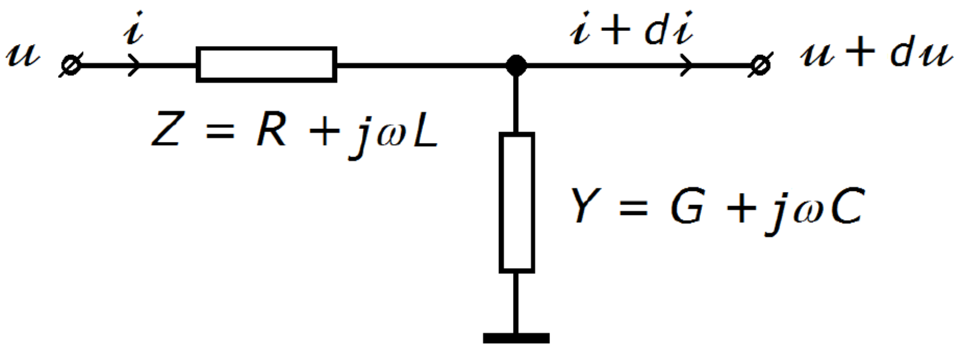



```
Terminal
File  Edit  View  Search  Terminal  Help
   3  tau          3.76223e+01   1.13854e-01   2.00376e-03   4.04917e-02
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas with name c1
root [1]
```

Contents

SU2 package

- *model dispersion of a square wave on a transmission line:*



$$-\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \partial_x \equiv \begin{pmatrix} 0 & L \\ C & 0 \end{pmatrix} \partial_t + \begin{pmatrix} 0 & R \\ G & 0 \end{pmatrix} \Bigg| \begin{pmatrix} u \\ i \end{pmatrix}$$

$$Z_0 = Y_0^{-1} = \sqrt{L/C}, \text{ line characteristic impedance}$$

$$\lambda_d^{-1} = (RY_0 - GZ_0)/2, \text{ dispersion length}$$

$$\lambda_a^{-1} = (RY_0 + GZ_0)/2, \text{ attenuation length}$$

$$c = 1/\sqrt{LC}, \text{ signal propagation speed}$$

- equation: $\partial_x + \sigma_1(\partial_{ct} + \lambda_a^{-1}) + j\sigma_2\lambda_d^{-1} = 0\,|_\psi$

$$\psi = e^{-ct/\lambda_a}\phi$$

$$\partial_x + \sigma_1\partial_{ct} + j\sigma_2\lambda_d^{-1} = 0\,|_\phi$$

- solution:

$$\phi = e^{-\gamma^2(1+\sigma_1\beta)\frac{j\sigma_2}{\lambda_d}(x-vt)}\,|_{\phi_0}$$

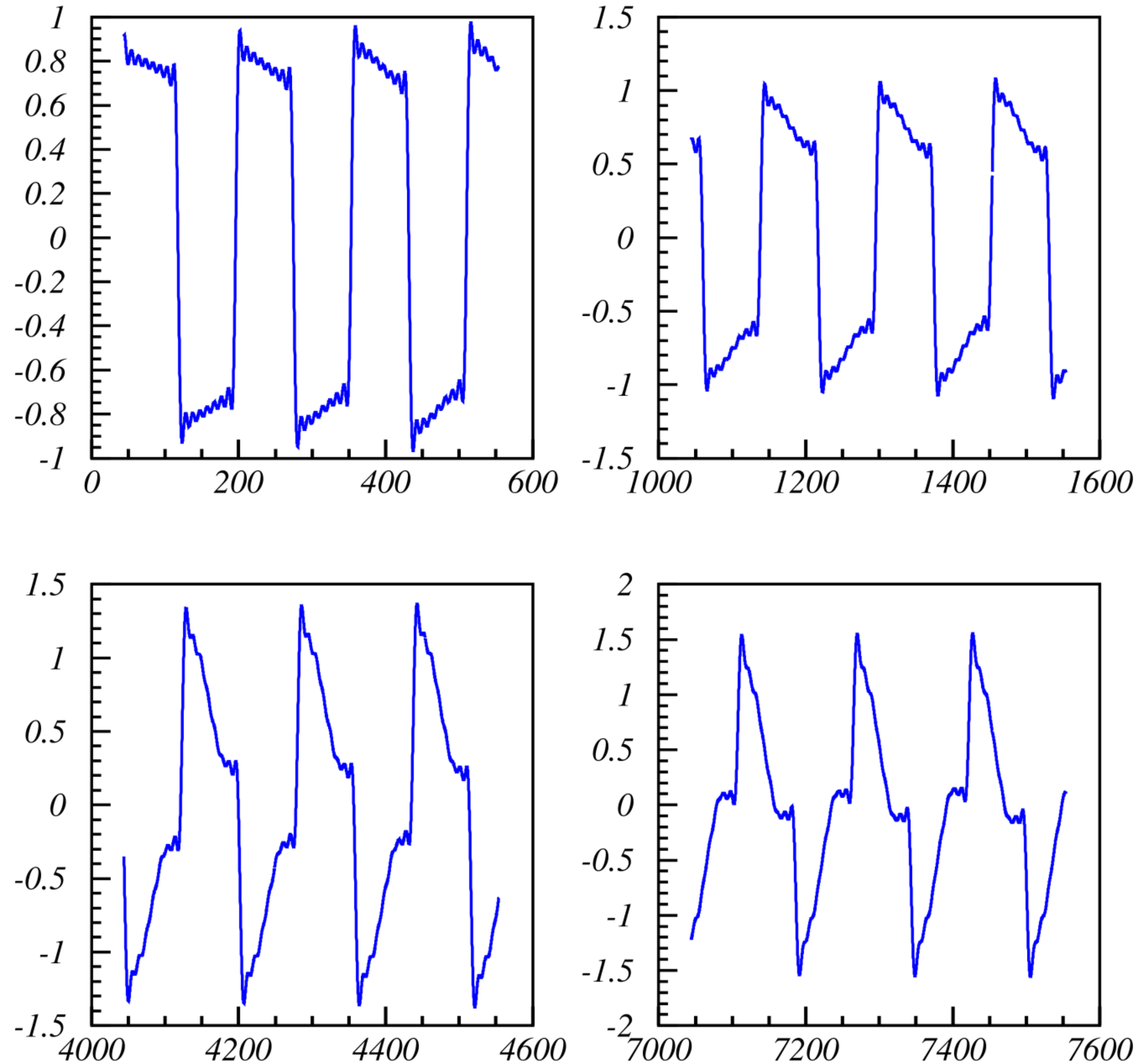
SU2 package

- I used the SU2 package to model the propagator:

```
auto propagator(real x, real t, real f)}
{
 real gamma = sqrt(1+f*f*Ld*Ld/c/c)
 real beta  = sqrt(gamma*gamma-1) / gamma      ;
 return e^(-(1+sx*beta)*(j*sy)*(x-beta*c*t)    ;
                          *gamma*gamma/Ld)     ;}
```

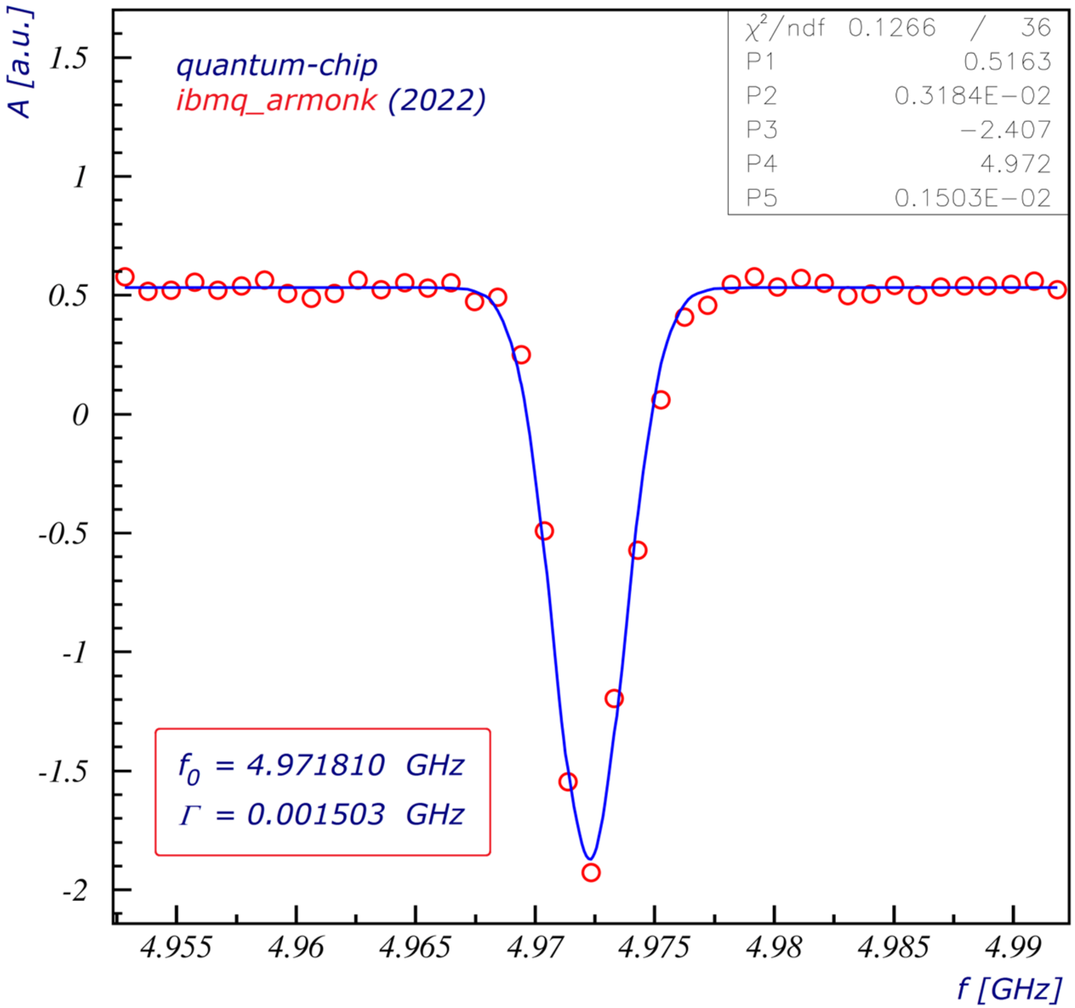
SU2 package

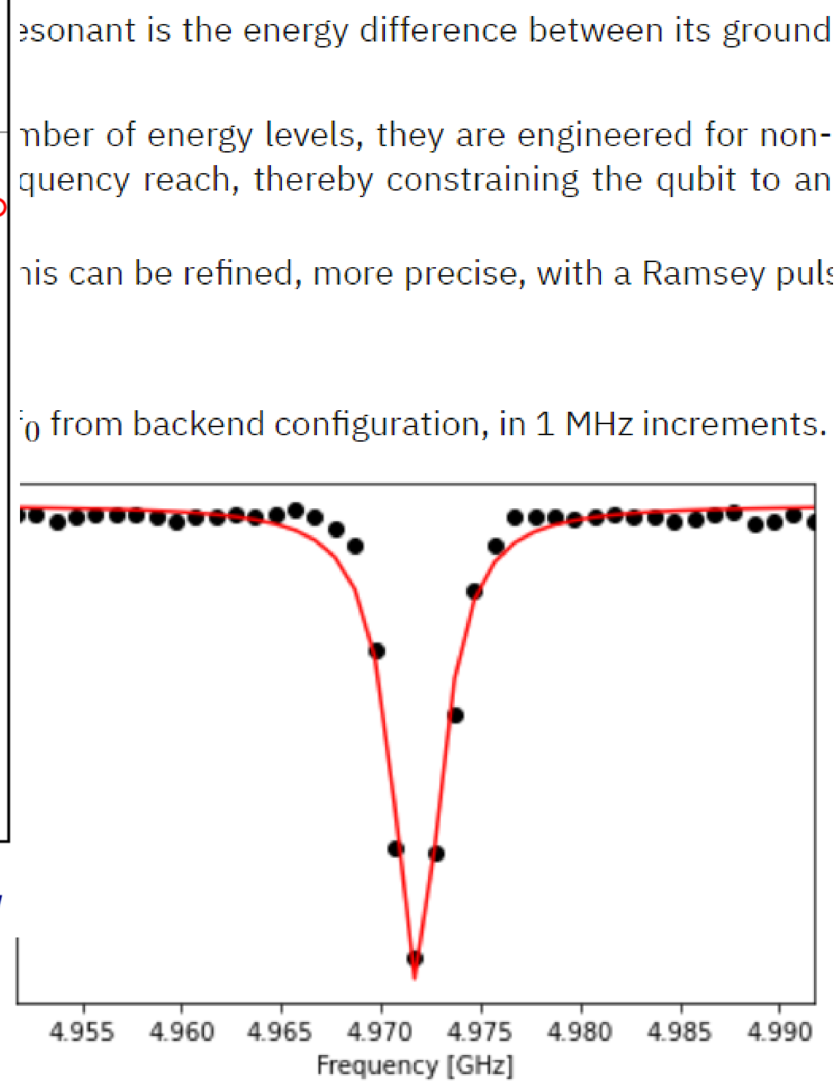
- I obtained a very nice solution of square wave dispersion:



Contents

Qubit resonance frequency





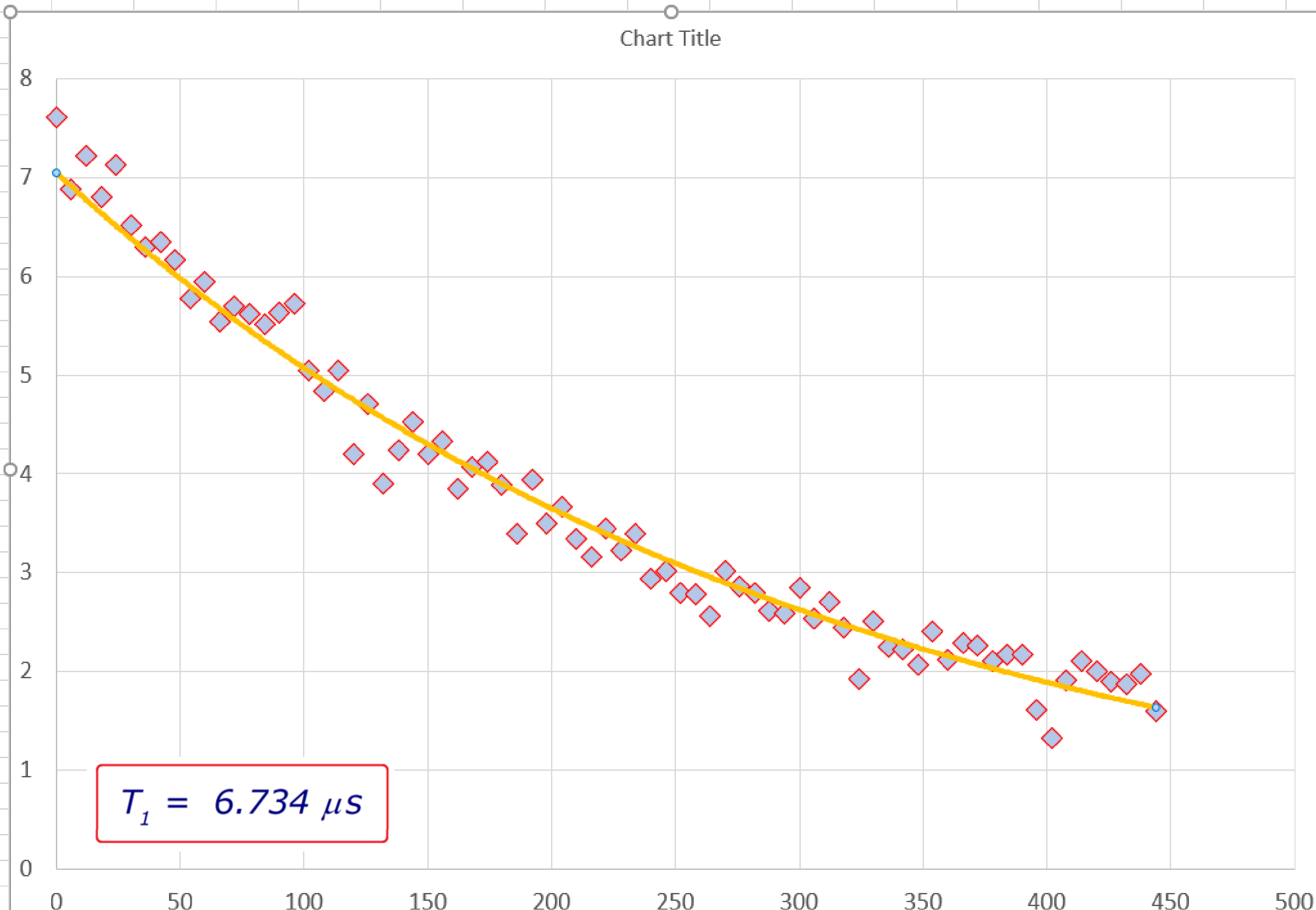
3.3 T_1 determination via Inversion Recovery

T_1 is
hurd

The
vary
(rela

C	D
246	3.020099
252	2.799042
258	2.787868
264	2.563377
270	3.014803
276	2.860762
282	2.792878
288	2.608236
294	2.580696
300	2.845292
306	2.53213
312	2.699927
318	2.438297
324	1.923568
330	2.513103
336	2.252348
342	2.227735
348	2.072595
354	2.407039
360	2.114896
366	2.290972
372	2.262727
378	2.10457
384	2.167686
390	2.176339
396	1.615931
402	1.322625
408	1.913311
414	2.103201
420	1.997242
426	1.900634
432	1.86954
438	1.976897
444	1.597172

E F G H I J K L M N O P Q R S T U



Bloch sphere

- 2 level system always equivalent to spin

- arbitrary wave-vector can be written as:

$$|\psi\rangle = \psi_\uparrow|\uparrow\rangle + \psi_\downarrow|\downarrow\rangle$$

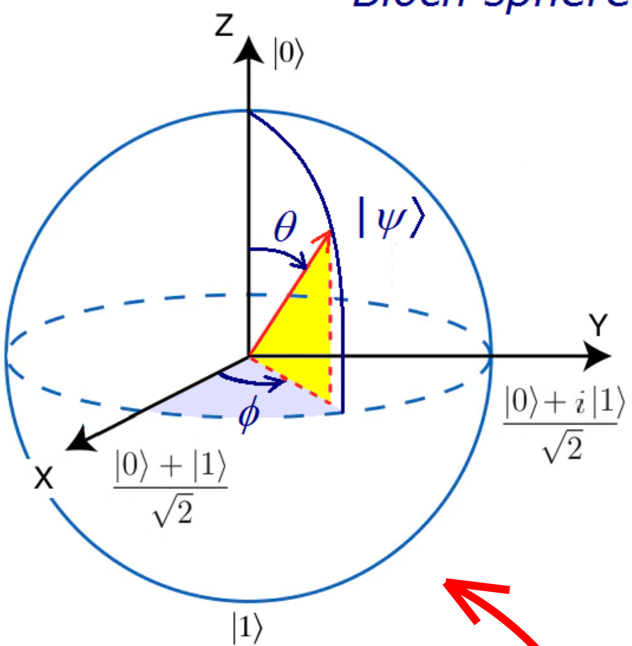
$$= e^{i\phi_\uparrow}\left(|\psi_\uparrow|\cdot|\uparrow\rangle + e^{i(\phi_\downarrow-\phi_\uparrow)}|\psi_\downarrow|\cdot|\downarrow\rangle\right)$$

$$= e^{i\phi_\uparrow}\sqrt{|\psi_\uparrow|^2+|\psi_\downarrow|^2}\left(\frac{|\psi_\uparrow|}{\sqrt{|\psi_\uparrow|^2+|\psi_\downarrow|^2}}|\uparrow\rangle + \frac{|\psi_\downarrow|}{\sqrt{|\psi_\uparrow|^2+|\psi_\downarrow|^2}}e^{i(\phi_\downarrow-\phi_\uparrow)}|\downarrow\rangle\right)$$

$$= e^{i\phi_\uparrow}\sqrt{|\psi_\uparrow|^2+|\psi_\downarrow|^2}\left(cos\frac{\theta}{2}|\uparrow\rangle + sin\frac{\theta}{2}e^{i(\phi_\downarrow-\phi_\uparrow)}|\downarrow\rangle\right)$$

represented on the Bloch sphere

Bloch sphere



Quantum logical gates

GATE	CIRCUIT REPRESENTATION	MATRIX REPRESENTATION	TRUTH TABLE	BLOCH SPHERE
H gate: rotates the qubit state by π radians (180º) about an axis diagonal in the x-z plane. This is equivalent to an X-gate followed by a $\frac{\pi}{2}$ rotation about the y-axis.		$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$	Input / Output $\lvert 0\rangle$ → $\frac{\lvert 0\rangle + \lvert 1\rangle}{\sqrt{2}}$ $\lvert 1\rangle$ → $\frac{\lvert 0\rangle - \lvert 1\rangle}{\sqrt{2}}$	

$$2\mathrm{U}_3(\theta,\phi,\lambda) = cos\frac{\theta}{2}\left[\left(1+e^{i(\lambda+\phi)}\right)\cdot\mathbf{1} + \left(1-e^{i(\lambda+\phi)}\right)\cdot\sigma_z\right] + sin\frac{\theta}{2}\left[e^{-i\lambda}\sigma_+ + e^{i\phi}\sigma_-\right]$$

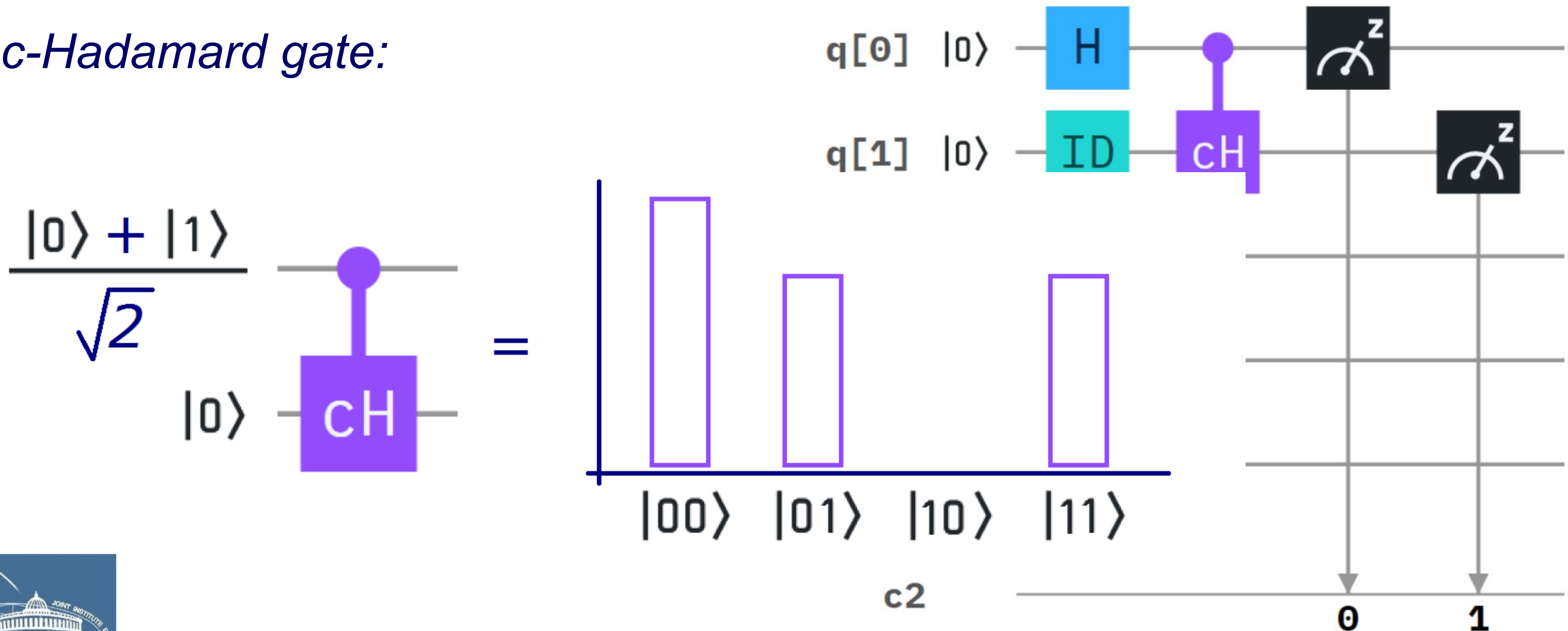
controlled-**U** gates

if `q[0]` = $|1\rangle$ operation **U** is performed on `q[1]`

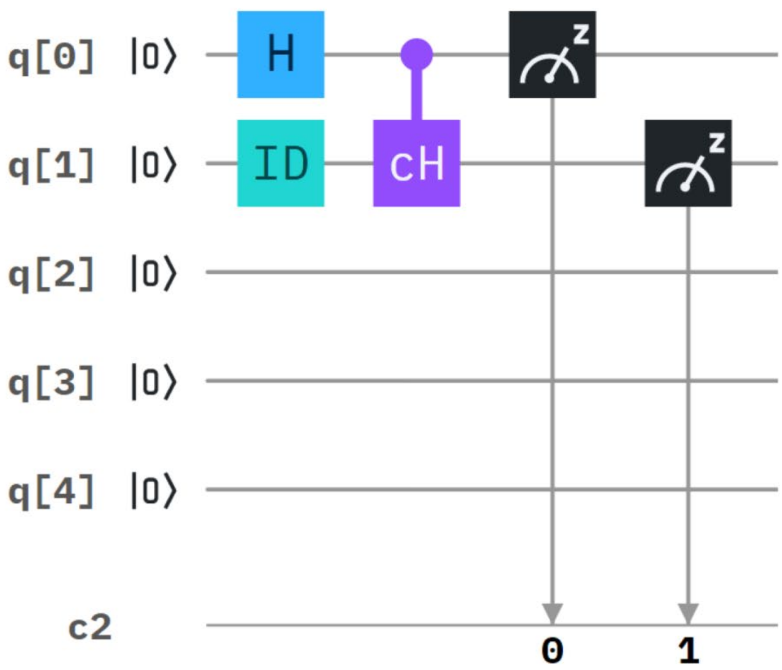
else `ID`

Entanglement

- 2 qubit states: $|\uparrow\uparrow\rangle, |\uparrow\downarrow\rangle, |\downarrow\uparrow\rangle$ and $|\downarrow\downarrow\rangle$
- entangled states: $|\psi\rangle = \dfrac{|\downarrow,\uparrow\rangle \pm |\uparrow,\downarrow\rangle}{\sqrt{2!}}$
- c-Hadamard gate:



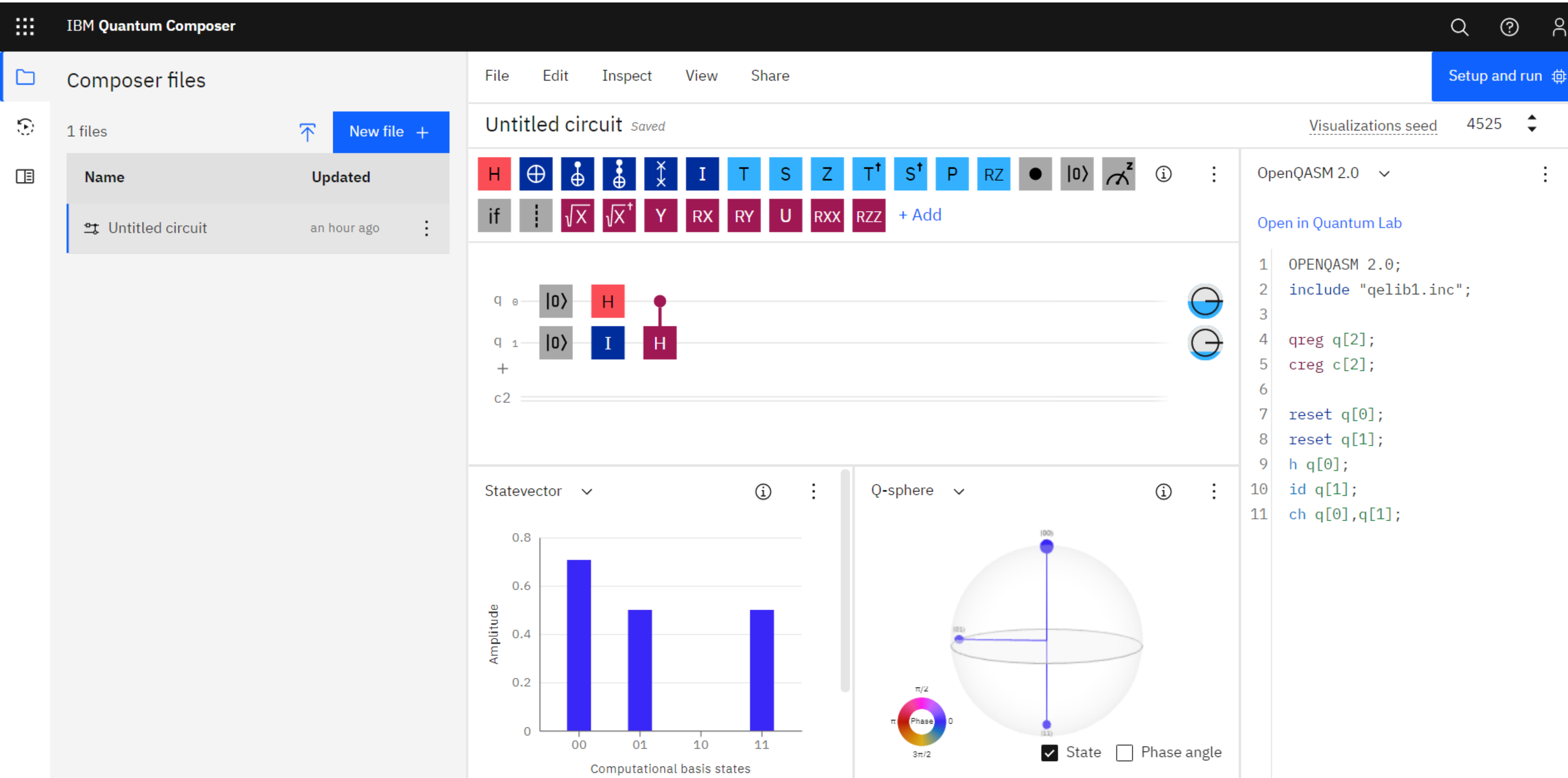
Circuit composer



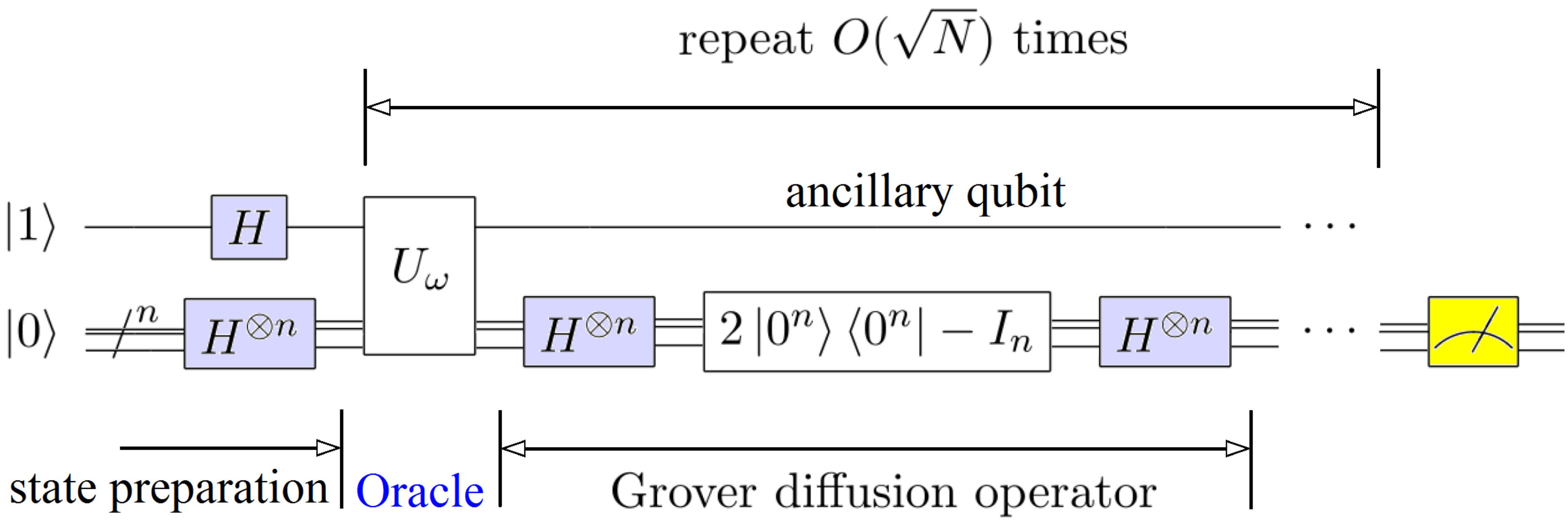
Circuit editor

```
OPENQASM 2.0;
include "qelib1.inc";

qreg q[5];
creg c[2];

h q[0];
id q[1];
ch q[0],q[1];
measure q[0] -> c[0];
measure q[1] -> c[1];
```

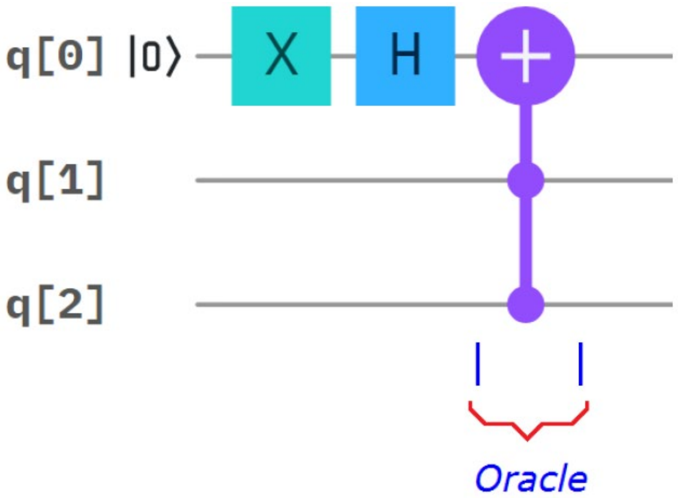
Create account



Grover algorithm



Detect the $|1,1\rangle$ state



$$\text{cc-NOT}(|q_0\rangle; q_1, q_2)$$
$$= \text{NOT}(|q_0\rangle) \quad \dots\dots \quad \text{for } (q_1, q_2) = (1, 1)$$
$$= \quad \text{ID}(|q_0\rangle) \quad \dots\dots \quad \text{for } (q_1, q_2) = (0, 0)$$
$$(0, 1)$$
$$(1, 0)$$

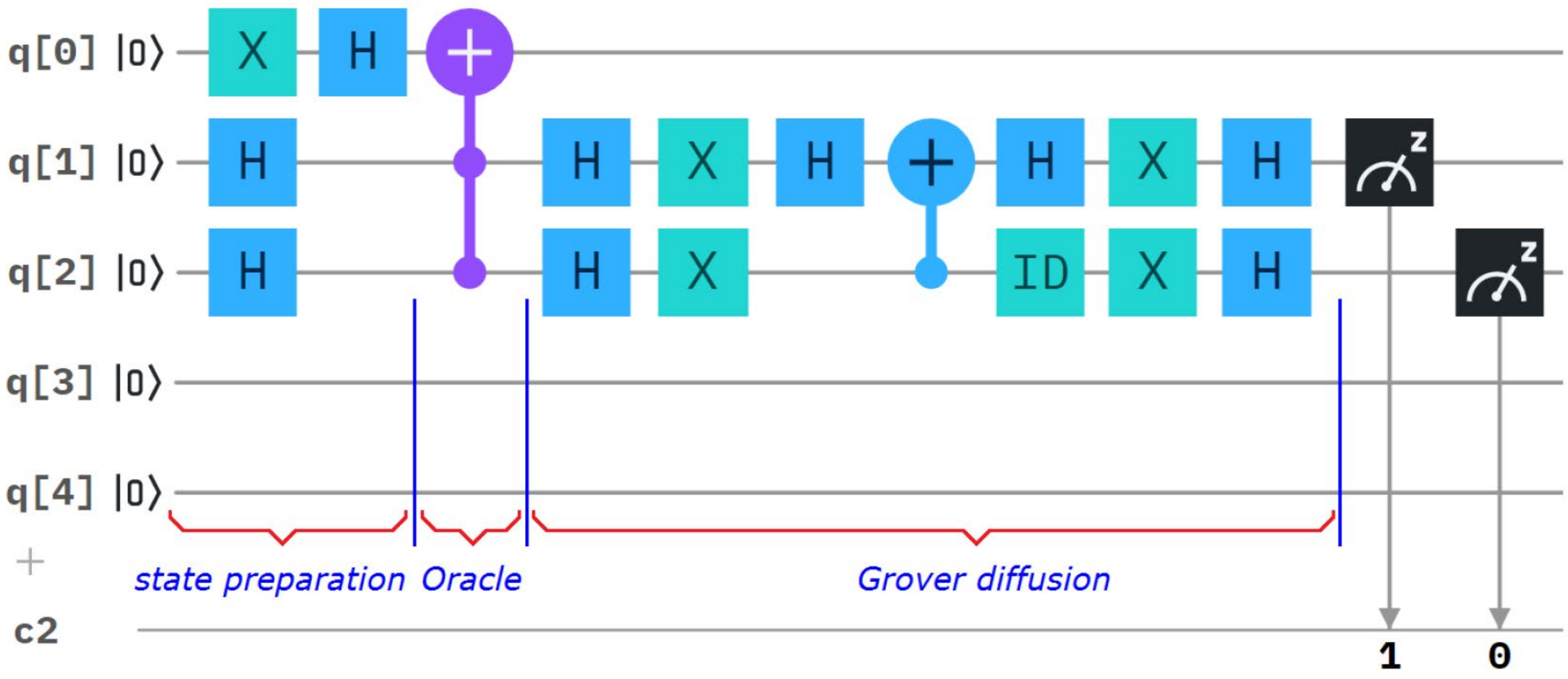
$$\text{NOT}(|0\rangle - |1\rangle) = |1\rangle - |0\rangle = -(|0\rangle - |1\rangle)$$

$$\text{ID}(|0\rangle - |1\rangle) = |0\rangle - |1\rangle = +(|0\rangle - |1\rangle)$$

Signals with a " - " the target state

IBM Q-Experience

Circuit composer

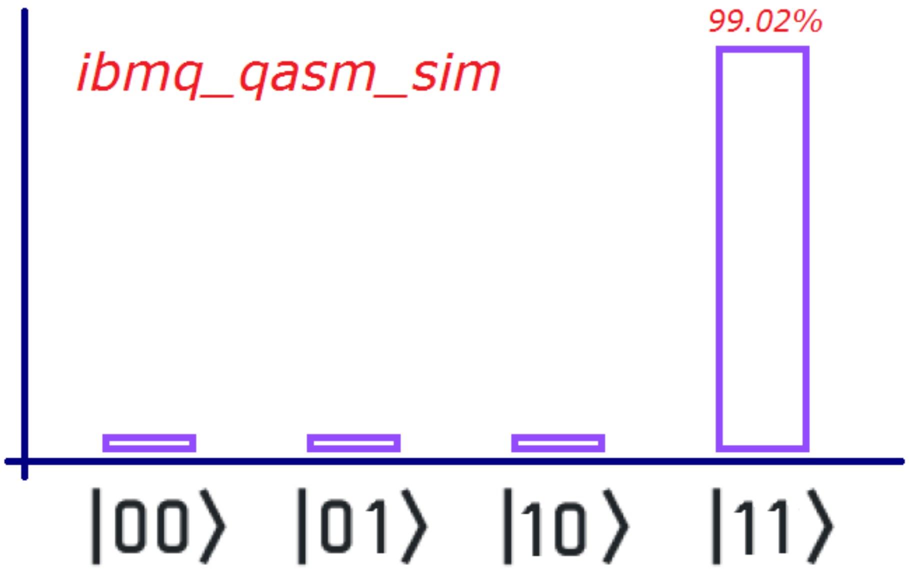


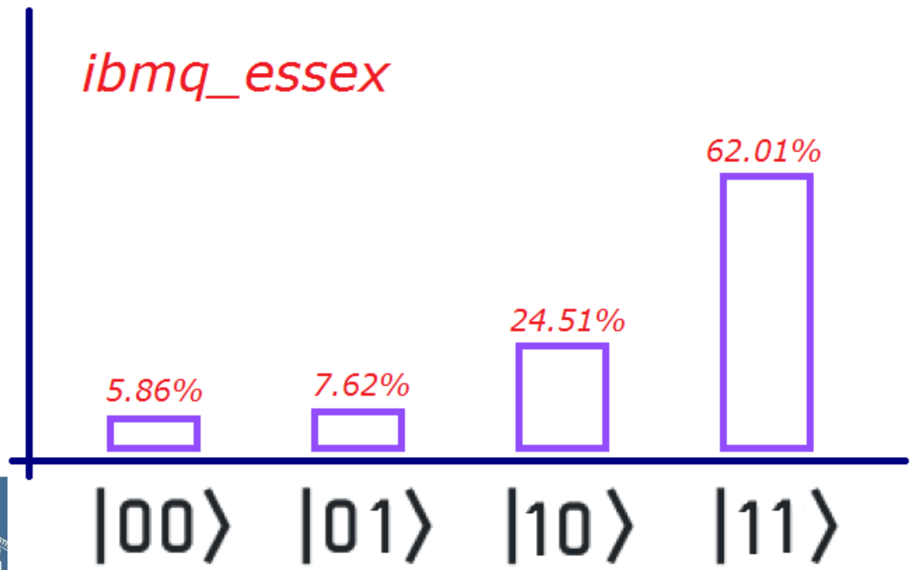
QASM-2

Circuit editor

```
OPENQASM 2.0;
include "qelib1.inc";

qreg q[5];
creg c[2];

x q[0];
h q[1];
h q[2];
h q[0];
ccx q[1],q[2],q[0];
h q[1];
h q[2];
x q[1];
x q[2];
h q[1];
cx q[2],q[1];
h q[1];
id q[2];
x q[1];
x q[2];
h q[1];
h q[2];
measure q[1] -> c[1];
measure q[2] -> c[0];
```

Results


ibmq_qasm_sim
99.02%
|00⟩
|01⟩
|10⟩
|11⟩


ibmq_essex
5.86%
7.62%
24.51%
62.01%
|00⟩
|01⟩
|10⟩
|11⟩

Conclusions

Personal opinions

- I learned about the quantum physics fundamentals of qubits and did some interesting hands-on determinations (f_0, T_1, T_2) of the ibmq_armonk qubit system on IBM's Q-Experience site

- We had access to the supercomputing cluster HybriLIT of JINR, which was very cool – for an SU2 simulation package in C++

- I learned to use the ROOT package from CERN to process and do fits on data

- We learned how to process multiple-entry quantum gate output and walked through the Grover quantum search algorithm – and after implemented and ran it on IBM's Q-Experience site

- The professors were very good and friendly, I highly recommend this student training programme !