March 3, 2015

# Notes on the 12 qubit PPS

Dawei  Lu

Notes about the problems in the 12 qubit PPS preparation, including Matlab codes and Experiments.

**DEC 12, 2014**

Calculating the state to state GRAPE on Ordi2. In pulsefinder folder. paramsfile is 'twqubit_subS2S.m', and the output file is 'twqubit_7zto12z'.

The GRAPE is to evolve ZZZZZZZIIIII to ZZZZZZZZZZZZ. As the couplings between nearest-neighbored C and H are about 150Hz. I set the GRAPE

```
1  % Number of timesteps
2  params.plength = 400;
3
4  % Length of each time step
5  params.timestep = 10e-6;
6
7  params.subsystem{1} = [1 2 3 9 10 11];
8  params.subsystem{2} = [4 5 6 7 8 12];
9  params.subsys_weight = [6 6];
10
11 % Input and goal states for state to state
12 params.rhoin = mkstate('+1ZZZZZZZIIIII',1);
13 params.rhogoal = mkstate('+1ZZZZZZZZZZZZ',1);
14
15 % Allow Zfreedom or not
16 params.Zfreedomflag = 1;
```

The fidelity keeps 0 all the time. Guess the reason is 'Zfreedom'. Set 'params.Zfreedomflag = 0;'. However, still 0.

Annie said maybe due to the length. Her SWAP gate requires 8ms, so I changed 'params.plength = 800;'. But for with or without Zfreedom, fidelity is still 0.

Check if some of my GRAPE settings are wrong. try to repeat Annie's SWAP gate calculation.

```
1  % Number of timesteps
2  params.plength = 800;
3
4  % Length of each time step
5  params.timestep = 10e-6;
6
7  params.subsystem{1} = [1 2 3 9 10 11];
8  params.subsystem{2} = [4 5 6 7 8 12];
9  params.subsys_weight = [6 6];
10
11 % Input and goal states for state to state
12 params.rhoin = mkstate('+1IIIIIIIZIIII+1IIIIIIIIZIII+1IIIIIIIIIZII+1IIIIIIIIIIZI
      +1IIIIIIIIIIIZ',1);
13 params.rhogoal = mkstate('+1IIIIIIIZIIII+1IIIIIIIIZIII+1IIIIIIIIIZII+1
      IIIIIIIIIIZI+1IIIIIZIIIIZ',1);
14
15 % Allow Zfreedom or not
16 params.Zfreedomflag = 0;
```

The outputfile is 'twqubit_SWAPC7H5'. And the fidelity is already over 98%. Then I changed 'params.Zfreedomflag = 1;', and the fidelity is over 95% after 30 iterations. Much slower than the no Zfreedom case. Maybe due to different initial guesses.

Generate all $\pi/2$ and $\pi$ pulses for the 7 Carbons, with the Calibration $= 25$KHz. $\pi/2$ pulses are 1ms length and 100 steps, and $\pi$ pulses are 2ms length and 200 steps. Generating Code in 'twqubit_shape.m'

```
1  for ii = 1:7
2  loadfile = ['twqubit_C', num2str(ii), '180', '.mat'];
3  eval(['load ', loadfile]);
4  filename1 = ['twqubit_C', num2str(ii), '180_C_25000.txt'];
5  filename2 = ['twqubit_C', num2str(ii), '180_H_25000.txt'];
6  make_bruker_shape(pulses{1}, 25000, filename1,1);
7  make_bruker_shape(pulses{1}, 25000, filename2,2);
8  end
```

The pulses are saved in Ordi2 '\pulsefinder\12 Qubit\' with the names such as 'twqubit_C590_C_25000.txt'.

I checked all the fidelities of the $\pi/2$ pulses in the folder '\pulseexam_12qubit\C_rotations\check_grape.m'. The code is

```
1   load Para.mat
2   load twpauliX_full.mat
3   load twpauliY_full.mat
4
5   %% Check all 90 rotations
6   %% Parameters for the GRAPE pulse
7   for spin_number = 1:7
8   Name1 = ['twqubit_C', num2str(spin_number), '90_C_25000.txt'];
9   Name2 = ['twqubit_C', num2str(spin_number), '90_H_25000.txt'];
10  Amplitude = 25000;
11  Time = 1e-3;
12  Length = 100;
13  dt = Time/Length;
14  FirstLine = 19; % the first line which contains the information of power and
        phase
15
16  Output1 = 'test1';
17  Output2 = 'test2';
18
19  [power1,phase1]=dataout(Name1,Output1,FirstLine,Length);
20  [power2,phase2]=dataout(Name2,Output2,FirstLine,Length);
21  %% Check
22  X_C = 0; Y_C = 0;
23  for jj = 1:7
24      X_C = X_C + KIx{jj};
25      Y_C = Y_C + KIy{jj};
26  end
27
28  X_H = 0; Y_H = 0;
29  for jj = 8:12
30      X_H = X_H + KIx{jj};
31      Y_H = Y_H + KIy{jj};
32  end
33
34
35  U = eye(2^12);
36  U = U*expm(-i*H*4e-6);
37  for ii = 1:Length
```

```
38       Hext = 2*pi*(Amplitude*power1(ii)/100)*(X_C*cos(phase1(ii)/360*2*pi)-Y_C*sin
             (phase1(ii)/360*2*pi))+2*pi*(Amplitude*power2(ii)/100)*(X_H*cos(phase2(ii
             )/360*2*pi)-Y_H*sin(phase2(ii)/360*2*pi));
39       U = expm(-i*(Hext+H)*dt)*U;
40  end
41  U = U*expm(-i*H*4e-6);
42
43  Utar = expm(-i*KIx{spin_number}*pi/2);
44
45  % Fidelity = ['Fidelity_C', num2str(spin_number), '90'];
46  % eval(['Fidelity_C', num2str(spin_number), '90 = abs(trace(U*Utar'))/2^12']);
47  Fidelity = abs(trace(U*Utar'))/2^12
48
49  savefile = ['twqubit_C', num2str(spin_number), '90_Ufid.mat'];
50  save (savefile, 'U', 'Fidelity');
51
52  end
```

Unitaries and Fidelities of the pulses will both be saved in 'twqubit_C590_Ufid.mat', so they can be called for further calculations in the PPS simulation. Wait for the results.

**DEC 16, 2014**

Combine pulses in the PPS preparation into big shape files, which should be easy for calibrations and pulsefixing.

The code is in the SVN server for Matlab named '\Twqubit\pulse_combine.m'.

First read all the powers and phases for the $\pi/2$ and $\pi$ rotations.

```matlab
for spin_number = 1:7
        Name1 = ['twqubit_C', num2str(spin_number), '90_C_25000.txt'];
        Name2 = ['twqubit_C', num2str(spin_number), '90_H_25000.txt'];
        [power1,phase1]=dataout(Name1,Output1,FirstLine,Length_90);
        [power2,phase2]=dataout(Name2,Output2,FirstLine,Length_90);
        eval(['power_C', num2str(spin_number), '90_C = power1;']); eval(['phase_C
            ', num2str(spin_number), '90_C = phase1;']);
        eval(['power_H', num2str(spin_number), '90_H = power2;']); eval(['phase_H
            ', num2str(spin_number), '90_H = phase2;']);
end

for spin_number = 1:7
        Name1 = ['twqubit_C', num2str(spin_number), '180_C_25000.txt'];
        Name2 = ['twqubit_C', num2str(spin_number), '180_H_25000.txt'];
        [power1,phase1]=dataout(Name1,Output1,FirstLine,Length_180);
        [power2,phase2]=dataout(Name2,Output2,FirstLine,Length_180);
        eval(['power_C', num2str(spin_number), '180_C = power1;']); eval(['
            phase_C', num2str(spin_number), '180_C = phase1;']);
        eval(['power_H', num2str(spin_number), '180_H = power2;']); eval(['
            phase_H', num2str(spin_number), '180_H = phase2;']);
end
```

Then combine them with the free evolutions. Here I set the time step dt = 10us.

```matlab
%% From Z7 to Z24567
step_27 = round(1/(4*Para(2,7))/dt);
step_67_27 = round((1/(4*Para(6,7))-1/(4*Para(2,7)))/dt);
step_47_67 = round((1/(4*Para(4,7))-1/(4*Para(6,7)))/dt);
step_57_47 = round((1/(4*Para(5,7))-1/(4*Para(4,7)))/dt);
step_57 = round((1/(4*Para(5,7)))/dt);

power_encoding1_C = [power_C790_C; zeros(step_27,1);power_C2180_C; zeros(
    step_67_27,1); power_C6180_C; zeros(step_47_67,1);power_C4180_C; zeros(
    step_57_47,1);...
                                    power_C5180_C; power_C7180_C; zeros(step_57,1)
                                        ;power_C790_C]*Calibration/Calibration_old;
phase_encoding1_C = [phase_C790_C; zeros(step_27,1);phase_C2180_C; zeros(
    step_67_27,1); phase_C6180_C; zeros(step_47_67,1);phase_C4180_C; zeros(
    step_57_47,1);...
                                    phase_C5180_C; phase_C7180_C; zeros(step_57,1)
                                        ;mod((phase_C790_C+90),360)];
power_encoding1_H = [power_H790_H; zeros(step_27,1);power_H2180_H; zeros(
    step_67_27,1); power_H6180_H; zeros(step_47_67,1);power_H4180_H; zeros(
    step_57_47,1);...
                                    power_H5180_H; power_H7180_H; zeros(step_57,1)
                                        ;power_H790_H]*Calibration/Calibration_old;
phase_encoding1_H = [phase_H790_H; zeros(step_27,1);phase_H2180_H; zeros(
    step_67_27,1); phase_H6180_H; zeros(step_47_67,1);phase_H4180_H; zeros(
    step_57_47,1);...
```

```matlab
15                                                    phase_H5180_H; phase_H7180_H; zeros(step_57,1)
                                                      ;mod((phase_H790_H+90),360)];
16
17  total_time_encoding1 = length(power_encoding1_C)*dt;
18
19  outputfile = 'twqubit_encoding1_C';
20  shpfile = fopen(outputfile,'w');
21      fprintf(shpfile,'##TITLE= %s\n',outputfile);
22      fprintf(shpfile,'##JCAMP-DX= 5.00 Bruker JCAMP library\n');
23      fprintf(shpfile,'##DATA TYPE= Shape Data\n');
24      fprintf(shpfile,'##ORIGIN= Dawei''s GRAPE Pulses \n');
25      fprintf(shpfile,'##OWNER= Dawei\n');
26      fprintf(shpfile,'##DATE= %s\n',date);
27      time = clock;
28      fprintf(shpfile,'##TIME= %d:%d\n',fix(time(4)),fix(time(5)));
29      fprintf(shpfile,'##MINX= %7.6e\n',min(power_encoding1_C));
30      fprintf(shpfile,'##MAXX= %7.6e\n',max(power_encoding1_C));
31      fprintf(shpfile,'##MINY= %7.6e\n',min(phase_encoding1_C));
32      fprintf(shpfile,'##MAXY= %7.6e\n',max(phase_encoding1_C));
33      fprintf(shpfile,'##$SHAPE_EXMODE= None\n');
34      fprintf(shpfile,'##$SHAPE_TOTROT= %7.6e\n',90);
35      fprintf(shpfile,'##$SHAPE_BWFAC= %7.6e\n',1);
36      fprintf(shpfile,'##$SHAPE_INTEGFAC= %7.6e\n',1);
37      fprintf(shpfile,'##$SHAPE_MODE= 1\n');
38      fprintf(shpfile, '##PULSE_WIDTH= %d\n',total_time_encoding1);
39      fprintf(shpfile, '##Calibration_Power= %d\n',Calibration);
40      fprintf(shpfile,'##NPOINTS= %d\n',length(power_encoding1_C));
41      fprintf(shpfile,'##XYPOINTS= (XY..XY)\n');
42
43  for ii = 1:length(power_encoding1_C)
44      fprintf(shpfile,'  %7.6e,  %7.6e\n',power_encoding1_C(ii),phase_encoding1_C(
          ii));
45  end
46
47      fprintf(shpfile,'##END=\n');
48
49  outputfile = 'twqubit_encoding1_H';
50  shpfile = fopen(outputfile,'w');
51      fprintf(shpfile,'##TITLE= %s\n',outputfile);
52      fprintf(shpfile,'##JCAMP-DX= 5.00 Bruker JCAMP library\n');
53      fprintf(shpfile,'##DATA TYPE= Shape Data\n');
54      fprintf(shpfile,'##ORIGIN= Dawei''s GRAPE Pulses \n');
55      fprintf(shpfile,'##OWNER= Dawei\n');
56      fprintf(shpfile,'##DATE= %s\n',date);
57      time = clock;
58      fprintf(shpfile,'##TIME= %d:%d\n',fix(time(4)),fix(time(5)));
59      fprintf(shpfile,'##MINX= %7.6e\n',min(power_encoding1_H));
60      fprintf(shpfile,'##MAXX= %7.6e\n',max(power_encoding1_H));
61      fprintf(shpfile,'##MINY= %7.6e\n',min(phase_encoding1_H));
62      fprintf(shpfile,'##MAXY= %7.6e\n',max(phase_encoding1_H));
63      fprintf(shpfile,'##$SHAPE_EXMODE= None\n');
64      fprintf(shpfile,'##$SHAPE_TOTROT= %7.6e\n',90);
65      fprintf(shpfile,'##$SHAPE_BWFAC= %7.6e\n',1);
66      fprintf(shpfile,'##$SHAPE_INTEGFAC= %7.6e\n',1);
67      fprintf(shpfile,'##$SHAPE_MODE= 1\n');
68      fprintf(shpfile, '##PULSE_WIDTH= %d\n',total_time_encoding1);
```

```
69      fprintf(shpfile,  '##Calibration_Power= %d\n',Calibration);
70      fprintf(shpfile,'##NPOINTS= %d\n',length(power_encoding1_H));
71      fprintf(shpfile,'##XYPOINTS= (XY..XY)\n');
72
73  for ii = 1:length(power_encoding1_H)
74      fprintf(shpfile,'  %7.6e,  %7.6e\n',power_encoding1_H(ii),phase_encoding1_H(
            ii));
75  end
76
77      fprintf(shpfile,'##END=\n');
```

The two output files are 'twqubit_encoding1_C' and 'twqubit_encoding1_H'. The calibrations are 25000Hz.

**DEC 17, 2014**

All fidelities of $\pi/2$ pulses are done! The folder is '\pulseexam_12qubit\C_rotations\'. Use 'check_power.m' to check the maximal powers for C and H channel.

| Rotation | Fidelity | File | MaxPower C | MaxPower H |
|---|---|---|---|---|
| $R_x^1(\pi/2)$ | 0.9838 | twqubit_C190_Ufid.mat | 56.0%, 14000Hz | 22.3%, 5557Hz |
| $R_x^2(\pi/2)$ | 0.9758 | twqubit_C290_Ufid.mat | 41.7%, 10422Hz | 23.5%, 5878Hz |
| $R_x^3(\pi/2)$ | 0.9647 | twqubit_C390_Ufid.mat | 31.9%, 7979.0Hz | 22.3%, 5568Hz |
| $R_x^4(\pi/2)$ | 0.9801 | twqubit_C490_Ufid.mat | 31.6%, 7892.0Hz | 23.8%, 5954Hz |
| $R_x^5(\pi/2)$ | 0.9936 | twqubit_C590_Ufid.mat | 56.1%, 14033Hz | 30.7%, 7678Hz |
| $R_x^6(\pi/2)$ | 0.9683 | twqubit_C690_Ufid.mat | 57.3%, 14333Hz | 34.4%, 8595Hz |
| $R_x^7(\pi/2)$ | 0.9857 | twqubit_C790_Ufid.mat | 43.7%, 10925Hz | 24.8%, 6207Hz |
| $R_x^1(\pi)$ | 0.9699 | twqubit_C1180_Ufid.mat | 62.6%, 15655Hz | 34.9%, 8726Hz |
| $R_x^2(\pi)$ | 0.9537 | twqubit_C2180_Ufid.mat | 51.1%, 12783Hz | 32.4%, 8094Hz |
| $R_x^3(\pi)$ | 0.9330 | twqubit_C3180_Ufid.mat | 37.4%, 9350.0Hz | 24.0%, 5997Hz |
| $R_x^4(\pi)$ | 0.9639 | twqubit_C4180_Ufid.mat | 45.1%, 11268Hz | 20.4%, 5108Hz |
| $R_x^5(\pi)$ | 0.9904 | twqubit_C5180_Ufid.mat | 67.6%, 16895Hz | 31.1%, 7782Hz |
| $R_x^6(\pi)$ | 0.9393 | twqubit_C6180_Ufid.mat | 71.8%, 17948Hz | 33.6%, 8396Hz |
| $R_x^7(\pi)$ | 0.9743 | twqubit_C7180_Ufid.mat | 51.0%, 12759Hz | 32.1%, 8022Hz |

For $\pi$ pulses, the maximal power of C5 exceeds 100% so it cannot be used. Check if we can generate $\pi$ pulses by combining two $\pi/2$ pulses. A potential problem is when calculating the GRAPE, we have considered the 4us free evolutions in the beginning and in the end. If we combine, we will have an unwanted 8us free evolution in the middle of the new $\pi$ pulse.

Use 'combine90to180' to check the $\pi$ pulse fidelity. They are very bad actually. All of them are just 0.75 0.76 in fidelity.

So I run 'check_grape.m' to check the fidelities of the $\pi$ pulses. Only from C1 to C4, as C5 has exceeds the power limit 25000Hz.

**DEC 22, 2014**

Got 4 GRAPE pulses for encoding. The folder is '\pulseexam_12qubit\C_rotations\'. The fidelities are in calculation on Ordi2.

| Rotation | Fidelity | File | MaxPower C | MaxPower H |
|---|---|---|---|---|
| $R_x^{5,7}(\pi)$ | 0.9667 | twqubit_C57180_Ufid.mat | 32.3%, 8072.5Hz | 24.2%, 6049Hz |
| $R_x^{2,3}(\pi)$ | 0.8908 | twqubit_C23180_Ufid.mat | 32.4%, 8101.5Hz | 22.8%, 5701Hz |
| $R_x^{2,3,4,7}(\pi/2)$ | 0.9156 | twqubit_C234790_Ufid.mat | 37.4%, 9358.3Hz | 28.9%, 7213Hz |
| $R_x^{1,5,6}(\pi)$ | 0.9055 | twqubit_C156180_Ufid.mat | 32.2%, 8039.7Hz | 20.3%, 5086Hz |

Have to recalculate many $\pi$ pulses.

**DEC 23, 2014**

Got $\pi$ pulse on C6. Combine two $\pi/2$ pulses as the initial guess, with the fidelity 0.75, and then search the optimal $\pi$ pulse. The convergence speed is very fast, which means initial guess is indeed very important in 12 qubits.

Now C2, C3, C5, C7 $\pi$ pulses are in calculation, with the initial guess.

**FEB 05, 2015**

C2, C3, C5, C7 $\pi$ pulses are all finished, and the update is in the Section Dec 17, 2014.

Also submitted the last pulse for the Encoding. From the PPS.m file in folder 'Twqubit'

```
%Phase Correction
U7 = R(gop(2,X),90)*R(gop(2,-Z),360*(Para(2,2)-20696)*1/2/148.5)*...
R(gop(3,-Y),90)*R(gop(3,-Z),360*(Para(3,3)-20696)*1/2/148.5)*...
R(gop(4,X),90)*R(gop(4,-Z),360*(Para(4,4)-20696)*1/2/148.5)*...
R(gop(7,X),90)*R(gop(7,-Z),360*(Para(7,7)-20696)*1/2/148.5);
```

And the operator is in 'twqubit_sub_234790_and_phasecorrection.m'

```
params.Uwant = expm(-1i*(90*pi/180)/2*full(mkstate('+1IXIIIIIIIIII-1IIYIIIIIIIII
    +1IIIXIIIIIIIII+1IIIIIIXIIIII',0)))*...
expm(-1i*((8778.95-20696)*1/2/148.5*360*pi/180)/2*full(mkstate('-1IZIIIIIIIIII'
    ,0)))*...
expm(-1i*((6245.16675-20696)*1/2/148.5*360*pi/180)/2*full(mkstate('-1
    IIZIIIIIIIII',0)))*...
expm(-1i*((10333.55-20696)*1/2/148.5*360*pi/180)/2*full(mkstate('-1IIIZIIIIIIIII'
    ,0)))*...
expm(-1i*((11928.21998-20696)*1/2/148.5*360*pi/180)/2*full(mkstate('-1
    IIIIIIZIIIII',0)));
```

**FEB 06, 2015**

The pulse 'twqubit_sub_234790_and_phasecorrection.m' which is the last piece of the encoding was found!

All $\pi$ pulses are found. The lowest is 0.9330 for C3 and the highest is 0.9904 for C5. Now is calculating the fidelity of the last piece in Encoding 'twqubit_C234790withPC_Ufid.mat'. Will combine all of the pulses in Encoding and check again after this calculation.

The last piece 'twqubit_C234790withPC_Ufid.mat' has been checked. The fidelity is 0.9164.

**FEB 12, 2015**

When combining all pulses into a large shape file, one has to know how to change a shape for X rotation to Y rotation. It should be a $\pi/2$ phase difference for every segment in the shape. I am going to check it.

The checking uses 4-qubit Crotonic in the folder 'F:\matlab\pulseexam_7qubit\4 qubit pulse check'. The target unitary is $R_x^1(\pi/2)$, and the pulse is 'Croton_90x1.txt' with length 1ms, 500 segments and amplitude 6000Hz.

When compared with $R_x^1(\pi/2)$, the fidelity is 0.9996. Then I changed the target to $R_y^1(\pi/2)$ with the GRAPE pulse unchanged. The fidelity goes to 0.4998 which is reasonable.

In order to produce a $R_y^1(\pi/2)$ from the original X rotation pulse, I added 90 to all phases in all segments. However, the fidelity goes to almost 0. Again all phases are reduced by 90, and this time the fidelity is 0.9996, which is what we want!

Conclusion: If you want to realize a Y rotation based on a X rotation pulse, just change the phase to phase-90 in each segment, and mod by 360 for the spectrometer.

```
phase = phase - 90;
phase = mod(phase, 360);
```

I wrote a program 'grape_phase' to generate the new phase in the folder 'F:\matlab\pulseexam_7qubit\'. Used in this manner 'phase_new = grape_phase( phase, initial_phase, end_phase)', where the initial_phase and end_phase can be X, Y, -X, or -Y.

The way to get the new operator is through the equation $R_z(\theta) = X R_y(\theta) \bar{X}$. If you know the unitary $U_x$ of the X rotation pulse, and when you are realizing Y pulse from that one, the new unitary $U_y$ is thus

$$U_y = R_z(\pi/2) U_x R_z(-\pi/2); \tag{1}$$

Generated the first encoding part, which will evolve Z7 to Z24567. The files 'twqubit_encoding1_C' and 'twqubit_encoding1_H' are in Ordi2 '\pulseexam_12qubit'. The total length is 32.98ms. Next I have to check whether the final state after this pulse will be Z24567 or not. So 'check_encoding.m' is written. The directory is Ordi2 '\pulseexam_12qubit'. This function will load all necessary .mat files to get the unitaries and calculate the final state based on these unitaries from Z7.

The final fidelity is 0.9832 (the same for with or without gradient) for Z24567. Two files 'U_encoding1.mat' and 'rho_encoding1.mat'. Now go on to the second Encoding part.

The fidelity for the second Encoding part is -0.9692 (the same for with or without gradient) for Z1234567. Two files 'twqubit_encoding2_C' and 'twqubit_encoding2_H' are in Ordi2 '\pulseexam_12qubit'. The total length is 21.29ms. Then the last piece in Encoding!

The fidelity for the third Encoding part is -0.9160 (the same for with or without gradient) for Z123456789101112. Two files 'twqubit_encoding3_C' and 'twqubit_encoding3_H' are in Ordi2 '\pulseexam_12qubit'. The total length is 7.36ms.

**FEB 18, 2015**

Running the two pulses. One is 'twqubit_all90.m' which is used for phase cycling, and the other one is 'twqubit_all90butC7.m' used for the polarization crush in the beginning. The fidelities are obtained!

| Rotation | Length | Fidelity | File | MaxPower C | MaxPower H |
|---|---|---|---|---|---|
| $R_x^{1-12}(\pi/2)$ | 1ms | 0.8738 | twqubit_all90_Ufid.mat | 27.8%, 6956.6Hz | 30.4%, 7594Hz |
| $R_x^{1-6,8-12}(\pi/2)$ | 1ms | 0.8849 | twqubit_all90butC7_Ufid.mat | 24.5%, 6134.9Hz | 25.0%, 6239Hz |

**FEB 23, 2015**

Running all 6 pulses for the decoding part. Got 2, 3, 5 and6! But the first pulse 'twqubit_C2347andH180.m' has to be recalculated.

Then check all the fidelities for pulse 2, 3, 5 and 6 first.

**MAR 3, 2015**

All 6 pulses in decoding have been checked. To my surprise some fidelities are pretty, such as the fidelity of $R_x^{2,3,4,5,6}(\pi)$ is only 0.8058. Anyway, it is better to check the state fidelity right away.

| Rotation | Length | Fidelity | File | MaxPower C | MaxPowe |
|---|---|---|---|---|---|
| $R_x^{2,3,4,7-12}(\pi)$ | 2ms | 0.8424 | twqubit_C2347andH180_Ufid.mat | 61.6%, 15400Hz | 52.2%, 130 |
| $R_x^{1,3,4,6}(\pi/2)R_{-y}^{8-12}(\pi/2)$ | 1ms | 0.9080 | twqubit_C134690andH90_Ufid.mat | 24.8%, 6203.2Hz | 22.1%, 55 |
| $R_x^{2,3,4,5,6}(\pi)$ | 2ms | 0.8058 | twqubit_C23456180_Ufid.mat | 37.8%, 9438.2Hz | 23.0%, 57 |
| $R_{-y}^{1,3,4,6}(\pi/2)R_x^2(\pi/2)R_{-z}^1((w_1-O_1)*(1/2J23-1/2J12))$ | 1ms | 0.8881 | twqubit_C1234690withPC_Ufid.mat | 21.2%, 5310.9Hz | 22.9%, 57 |
| $R_x^{2,7}(\pi)$ | 2ms | 0.9306 | twqubit_C27180_Ufid.mat | 29.1%, 7285.3Hz | 21.7%, 54 |
| $R_{-y}^2(\pi/2)R_x^5(\pi/2)$ | 1ms | 0.9728 | twqubit_C2Y5X90_Ufid.mat | 59.2%, 14795Hz | 40.9%, 102 |

**ALL PULSES FOR 12 QUBITS**

The saving folder is '\pulseexam_12qubit\C_rotations\'.

$\pi/2$ and $\pi$ rotations on every single spin.

| Rotation | Length | Fidelity | File | MaxPower C | MaxPower H |
|---|---|---|---|---|---|
| $R_x^1(\pi/2)$ | 1ms | 0.9838 | twqubit_C190_Ufid.mat | 56.0%, 14000Hz | 22.3%, 5557Hz |
| $R_x^2(\pi/2)$ | 1ms | 0.9758 | twqubit_C290_Ufid.mat | 41.7%, 10422Hz | 23.5%, 5878Hz |
| $R_x^3(\pi/2)$ | 1ms | 0.9647 | twqubit_C390_Ufid.mat | 31.9%, 7979.0Hz | 22.3%, 5568Hz |
| $R_x^4(\pi/2)$ | 1ms | 0.9801 | twqubit_C490_Ufid.mat | 31.6%, 7892.0Hz | 23.8%, 5954Hz |
| $R_x^5(\pi/2)$ | 1ms | 0.9936 | twqubit_C590_Ufid.mat | 56.1%, 14033Hz | 30.7%, 7678Hz |
| $R_x^6(\pi/2)$ | 1ms | 0.9683 | twqubit_C690_Ufid.mat | 57.3%, 14333Hz | 34.4%, 8595Hz |
| $R_x^7(\pi/2)$ | 1ms | 0.9857 | twqubit_C790_Ufid.mat | 43.7%, 10925Hz | 24.8%, 6207Hz |
| $R_x^1(\pi)$ | 2ms | 0.9699 | twqubit_C1180_Ufid.mat | 62.6%, 15655Hz | 34.9%, 8726Hz |
| $R_x^2(\pi)$ | 2ms | 0.9537 | twqubit_C2180_Ufid.mat | 51.1%, 12783Hz | 32.4%, 8094Hz |
| $R_x^3(\pi)$ | 2ms | 0.9330 | twqubit_C3180_Ufid.mat | 37.4%, 9350.0Hz | 24.0%, 5997Hz |
| $R_x^4(\pi)$ | 2ms | 0.9639 | twqubit_C4180_Ufid.mat | 45.1%, 11268Hz | 20.4%, 5108Hz |
| $R_x^5(\pi)$ | 2ms | 0.9904 | twqubit_C5180_Ufid.mat | 67.6%, 16895Hz | 31.1%, 7782Hz |
| $R_x^6(\pi)$ | 2ms | 0.9393 | twqubit_C6180_Ufid.mat | 71.8%, 17948Hz | 33.6%, 8396Hz |
| $R_x^7(\pi)$ | 2ms | 0.9743 | twqubit_C7180_Ufid.mat | 51.0%, 12759Hz | 32.1%, 8022Hz |

Pulses for the encoding part of PPS preparation.

| Rotation | Length | Fidelity | File | MaxPower C | MaxPower H |
|---|---|---|---|---|---|
| $R_x^{5,7}(\pi)$ | 2ms | 0.9667 | twqubit_C57180_Ufid.mat | 32.3%, 8072.5Hz | 24.2%, 6049Hz |
| $R_x^{2,3}(\pi)$ | 2ms | 0.8908 | twqubit_C23180_Ufid.mat | 32.4%, 8101.5Hz | 22.8%, 5701Hz |
| $R_x^{2,3,4,7}(\pi/2)$ | 1ms | 0.9156 | twqubit_C234790_Ufid.mat | 37.4%, 9358.3Hz | 28.9%, 7213Hz |
| $R_x^{1,5,6}(\pi)$ | 2ms | 0.9055 | twqubit_C156180_Ufid.mat | 32.2%, 8039.7Hz | 20.3%, 5086Hz |
| $R_x^{2,4,7}(\pi/2)R_{-y}^3(\pi/2)R_{-z}^{i=2,3,4,7}((w_i - O_1) * 1/2J_{CH})$ | 1ms | 0.9164 | twqubit_C234790withPC_Ufid.mat | 47.4%, 11841Hz | 25.3%, 6336Hz |

All 3 Encoding pulses in the saving folder '\pulseexam_12qubit\'. The fidelities in the following table are state fidelities

| Files | Length | Target State | State Fidelity |
|---|---|---|---|
| twqubit_encoding1_C, H | 32.98ms | IZIZZZZIIIII | 0.9832 |
| twqubit_encoding2_C, H | 21.29ms | ZZZZZZZIIIII | -0.9692 |
| twqubit_encoding3_C, H | 7.36ms | ZZZZZZZZZZZZ | -0.9160 |

Pulses for polarization crush and phase cycling.

| Rotation | Length | Fidelity | File | MaxPower C | MaxPower H |
|---|---|---|---|---|---|
| $R_x^{1-12}(\pi/2)$ | 1ms | 0.8738 | twqubit_all90_Ufid.mat | 27.8%, 6956.6Hz | 30.4%, 7594Hz |
| $R_x^{1-6,8-12}(\pi/2)$ | 1ms | 0.8849 | twqubit_all90butC7_Ufid.mat | 24.5%, 6134.9Hz | 25.0%, 6239Hz |

Pulses for the decoding part of PPS preparation.

| Rotation | Length | Fidelity | File | MaxPower C | MaxPow |
|---|---|---|---|---|---|
| $R_x^{2,3,4,7-12}(\pi)$ | 2ms | 0.8424 | twqubit_C2347andH180_Ufid.mat | 61.6%, 15400Hz | 52.2%, 130 |
| $R_x^{1,3,4,6}(\pi/2)R_{-y}^{8-12}(\pi/2)$ | 1ms | 0.9080 | twqubit_C134690andH90_Ufid.mat | 24.8%, 6203.2Hz | 22.1%, 55 |
| $R_x^{2,3,4,5,6}(\pi)$ | 2ms | 0.8058 | twqubit_C23456180_Ufid.mat | 37.8%, 9438.2Hz | 23.0%, 57 |
| $R_{-y}^{1,3,4,6}(\pi/2)R_x^2(\pi/2)R_{-z}^1((w_1-O_1)*(1/2J23-1/2J12))$ | 1ms | 0.8881 | twqubit_C1234690withPC_Ufid.mat | 21.2%, 5310.9Hz | 22.9%, 57 |
| $R_x^{2,7}(\pi)$ | 2ms | 0.9306 | twqubit_C27180_Ufid.mat | 29.1%, 7285.3Hz | 21.7%, 54 |
| $R_{-y}^2(\pi/2)R_x^5(\pi/2)$ | 1ms | 0.9728 | twqubit_C2Y5X90_Ufid.mat | 59.2%, 14795Hz | 40.9%, 102 |