Corresponding Manuscript: Conditional Operation of Hole Spin Qubits above 1 K — [Notebook: S5_Two_Tone_varied_pump-time_J38.7MHZ_v040925_v3.nb]

Authors: M.J. Carballido | ORCID: https://orcid.org/0000-0001-7385-8284
T. Patlatiuk

Project mirror: https://github.com/carmig00/Publications-OpenAccess-Code-Conditional2Q

How to run: Evaluate top-to-bottom (Evaluation → Evaluate Notebook).

Outputs: HDF5 (.h5) written to ./exports.

Figure mapping: This notebook reproduces the supplementary information's 2-tone experiments which were expanded by an additional dimension. That is, the duration of the pump tone, at fixed frequency f1 to f4, was additionally varied.

Comment/Note:

-The fast qubit (high f_Rabi), at lower Larmor frequency, is the one located physically on the right and is represented by pink/purple color tones, here qubit 1.

- The slow qubit (low f_Rabi), at higher Larmor frequency, is the one located physically on the left and is represented by orange/yellow color tones, here qubit 2.

```
(* f_Rabi,MHz *)

(* We assume this to be the bare Rabi frequency of the two ideally isolated
 spin qubits. Sometimes also referred to as bare Rabi frequency,for J=0 *)
fR1t = 24;
fR2t = 13;

(* Experimentally chosen pulse durations to induce a Pi/2 flip for qubit 1 and qubit 2,
see supplementary material for details on the calibration *)
(* units in \mus *)
Tp1 = 20×10^-3; (* corresponding to 20 ns *)
Tp2 = 30×10^-3; (* corresponding to 30 ns *)

(* Effective exchnage interaction J, MHz*)
Jval = 38.7;
(*Larmor frequencies,
2*Pi*MHz. These values were settled on after running the 2-tone simulations.*)
ωz1 = 2 Pi 2075;
ωz2 = 2 Pi 2270;

(* Empirical frequency choices for f1-f4, see supplementary information. *)
(* 2*Pi*MHz *)
ωxexp1 = 2 Pi 2057;
ωxexp2 = 2 Pi 2094;
ωxexp3 = 2 Pi 2257;
ωxexp4 = 2 Pi 2284;
```

```
(* Theta is the SO-angle. In an accurate and complete microscopic description,
his accounts for the angle by which the spin is rotated due to SOI,
i.e. the angle in the rotation matrix R. To avoid over-
 fitting this angle is set to 0 as we consider an effective J,
J_eff between the two qubits *)
θso = 0;
(* Phases of the drives. *)
φ1 = 0;
φ2 = 0;


(* Populations:
 These are the states that are blocked in PSB and which we initialize in. *)
ψ0a = {1, 0, 0, 0};
ψ0b = {0, 0, 0, 1};

(*NOTE: block not used here. Originally for testing time scans.*)
resTg = 100;
(*initial time*)
Tg0 = 0;
(*final time, 66.6ns*)
Tgf = 3 * 1 / 30;
Tgarr = Table[t, {t, Tg0, Tgf, (Tgf - Tg0)/(resTg - 1)}];



(* SO-vector *)
{nx, ny, nz} = {1, 0, 0};

(*Effective exchange interaction*)
(*Relevant component parallel with respect to external magnetic field B.*)
Jpar[J_] = 2 Pi J (nz^2 + (1 - nz^2) Cos[θso]);
(*λx00=2 Pi 30; (*Russ/Burkard resonance condition for driving*)*)


(*2-Qubit Hamiltonian*)
HQ[ωz1in_, ωz2in_] = KroneckerProduct[PauliMatrix[3] ωz1in/2, PauliMatrix[0]] +
   KroneckerProduct[PauliMatrix[0], PauliMatrix[3] ωz2in/2];
Rso = {
   {nx^2 + (1 - nx^2) Cos[θso],
    nx ny - nx ny Cos[θso] - nz Sin[θso], nx nz - nx nz Cos[θso] + ny Sin[θso]},
```

```
  {nx ny - nx ny Cos[θso] + nz Sin[θso], ny^2 + (1 - ny^2) Cos[θso],
   ny nz - ny nz Cos[θso] - nx Sin[θso]}, {nx nz - nx nz Cos[θso] - ny Sin[θso],
   2 Sin[θso/2] (nx Cos[θso/2] + ny nz Sin[θso/2]), nz^2 + (1 - nz^2) Cos[θso]}
 };


(*Exchange Hamiltonian*)
HJ[J_] =
  J/4 Sum[KroneckerProduct[PauliMatrix[i], (Rso.Array[PauliMatrix, 3])[[i]]], {i, 1, 3}];



(*turn on time of pulses*)
dt0 = 0.001;
(*wait time between pulses*)
Tw = 0.001;
(*driving amplitudes,here normalised to 2 Pi and equal to each other.One could choose to
 e.g.drive one qubit harder than the other one.(this could lead to over-fitting)*)
λx10 = 2 Pi; (*Drive amplitude Q1*)
λx20 = 2 Pi; (*Drive amplitude Q2*)

(*first pulse for a time Tflip1*)
λx1[t_] = λx10 (1/2 + 1/2 Erf[(Tflip1 - t) / dt0]);
(*second pulse after a time Tflip1+Tw*)
λx2[t_] = λx20 (1/2 + 1/2 Erf[(t - Tflip1 - Tw) / dt0]);

(* Drive Hamiltonian *)
Hλ[t_] = λx1[t] Cos[ωx1 t + φ1]
    (fR1 KroneckerProduct[{nx, ny, nz}.Array[PauliMatrix, 3], PauliMatrix[0]] + fR2
       KroneckerProduct[PauliMatrix[0], {nx, ny, nz}.Array[PauliMatrix, 3]]) + λx2[t] Cos[
     ωx2 t + φ2] (fR1 KroneckerProduct[{nx, ny, nz}.Array[PauliMatrix, 3], PauliMatrix[0]] +
      fR2 KroneckerProduct[PauliMatrix[0], {nx, ny, nz}.Array[PauliMatrix, 3]]);


(*Total Hamiltonian*)
Htot[t_, J_, ωz1in_, ωz2in_] = HQ[ωz1in, ωz2in] + HJ[J] + Hλ[t];
ρSf =
  Function[{ψ0, ψro, Tflip1in, Tflip2in, ωz1in, ωz2in, ωx1in, ωx2in, fR1in, fR2in, Jin}, sub =
    {Tflip1 → Tflip1in, Tflip2 → Tflip2in, ωx1 → ωx1in, ωx2 → ωx2in, fR1 → fR1in, fR2 → fR2in};
   Tfin = Tflip1 + Tw + Tflip2 /. sub;
   ψfin[t_] = {a[t], b[t], c[t], d[t]} /. NDSolve[
```

```
        {Htot[t, Jin, ωz1in, ωz2in].{a[t], b[t], c[t], d[t]} ==
           ⅈ D[{a[t], b[t], c[t], d[t]}, t] /. sub,
         a[0] == ψ0⟦1⟧, b[0] == ψ0⟦2⟧, c[0] == ψ0⟦3⟧, d[0] == ψ0⟦4⟧},
        {a[t], b[t], c[t], d[t]}, {t, 0, Tfin},
        Method → "PDEDiscretization" → {"MethodOfLines",
           "SpatialDiscretization" → "FiniteElement", MaxCellMeasure → 0.0005 / fR1}]⟦1⟧;
    (*readout signal*)
    Sum[Abs[(ψro⟦i⟧.ψfin[Tfin])]^2, {i, 1, Dimensions[ψro]⟦1⟧}];
```

```
(*Two-tone spectroscopy 2D, averaged over different Larmor ferquencies*)

(*To simulate noise, we averaged over different Larmor
 frequencies sampled from a random, normal distributed set.*)
(*sigma of normal distributuion 2*Pi*MHz *)
σωz1 = 25 × 2 Pi;
σωz2 = 25 × 2 Pi;
(* Resolution of the array of random Larmor frequencies MHz*)
resωz1 = 150; (* 150 *)
resωz2 = 150; (* 150 *)
(* Arrays of random Larmor freqs. MHz*)
ωz1arr = Abs[RandomVariate[NormalDistribution[ωz1, σωz1], {resωz1}]];
ωz2arr = Abs[RandomVariate[NormalDistribution[ωz2, σωz2], {resωz2}]];

(* 2Pi rotation of the slover qubit to define upper time bound.*)
t1max = 1 / fR2t;
rest1 = 40; (* 40 *)
t1arr = Table[t1val, {t1val, 0, t1max, t1max / (rest1 - 1)}];

(*Resolution of frequency scanned (x-axis) MHz*)
resω = 80; (* 160 *)
(*initial freq*)
ωin = 2 π 1800;
(*final freq*)
ωfin = 2 π 2600;
(*array of freqs. scanned over, MHz*)
ωarr = Table[t, {t, ωin, ωfin, (ωfin - ωin) / (resω - 1)}];


(* We performed the experiment with the fixed frequency
 guesses introduced in the supplementary information (and above),
denoted as ωxexp1 to ωxexp4 above. We further use ωz1 and ωz2 that we extracted as fit
  parameters from the previous two-tone scans. The values for the Larmor frequencies
  ωz1 and ωz2 found herein are the same ones as used in the two-tone scans.*)
```

```
(* We read out in the 0,1,0,0 and *)
(* 0,0,1,0 states (anti-parallel) which are not blocked in PSB *)
(*NOTE: For the probe tones, we apply the Pi pulse Tp1 for
 f1 and f2 and Tp2 for f3 and f4. The pump tones follow the array*)

resF1vs2D =
   ParallelTable[(ρSf[ψ0a, {{0, 1, 0, 0}, {0, 0, 1, 0}}, t1arr⟦k⟧, Tp1, ωz1arr⟦j⟧, ωz2arr⟦j⟧,
        ωxexp1, ωarr⟦i⟧, fR1t, fR2t, Jpar[Jval]] + ρSf[ψ0b, {{0, 1, 0, 0}, {0, 0, 1, 0}},
        t1arr⟦k⟧, Tp1, ωz1arr⟦j⟧, ωz2arr⟦j⟧, ωxexp1, ωarr⟦i⟧, fR1t, fR2t, Jpar[Jval]]) / 2,
     {j, 1, resωz1}, {k, 1, rest1}, {i, 1, resω}]; // AbsoluteTiming
resF2vs2D =
   ParallelTable[(ρSf[ψ0a, {{0, 1, 0, 0}, {0, 0, 1, 0}}, t1arr⟦k⟧, Tp1, ωz1arr⟦j⟧, ωz2arr⟦j⟧,
        ωxexp2, ωarr⟦i⟧, fR1t, fR2t, Jpar[Jval]] + ρSf[ψ0b, {{0, 1, 0, 0}, {0, 0, 1, 0}},
        t1arr⟦k⟧, Tp1, ωz1arr⟦j⟧, ωz2arr⟦j⟧, ωxexp2, ωarr⟦i⟧, fR1t, fR2t, Jpar[Jval]]) / 2,
     {j, 1, resωz1}, {k, 1, rest1}, {i, 1, resω}]; // AbsoluteTiming
resF3vs2D =
   ParallelTable[(ρSf[ψ0a, {{0, 1, 0, 0}, {0, 0, 1, 0}}, t1arr⟦k⟧, Tp2, ωz1arr⟦j⟧, ωz2arr⟦j⟧,
        ωxexp3, ωarr⟦i⟧, fR1t, fR2t, Jpar[Jval]] + ρSf[ψ0b, {{0, 1, 0, 0}, {0, 0, 1, 0}},
        t1arr⟦k⟧, Tp2, ωz1arr⟦j⟧, ωz2arr⟦j⟧, ωxexp3, ωarr⟦i⟧, fR1t, fR2t, Jpar[Jval]]) / 2,
     {j, 1, resωz1}, {k, 1, rest1}, {i, 1, resω}]; // AbsoluteTiming
resF4vs2D =
   ParallelTable[(ρSf[ψ0a, {{0, 1, 0, 0}, {0, 0, 1, 0}}, t1arr⟦k⟧, Tp2, ωz1arr⟦j⟧, ωz2arr⟦j⟧,
        ωxexp4, ωarr⟦i⟧, fR1t, fR2t, Jpar[Jval]] + ρSf[ψ0b, {{0, 1, 0, 0}, {0, 0, 1, 0}},
        t1arr⟦k⟧, Tp2, ωz1arr⟦j⟧, ωz2arr⟦j⟧, ωxexp4, ωarr⟦i⟧, fR1t, fR2t, Jpar[Jval]]) / 2,
     {j, 1, resωz1}, {k, 1, rest1}, {i, 1, resω}]; // AbsoluteTiming

Export["C:\\Users\\export\\two_tone_2D_average_Larmor_sigma" <>
   TextString[σωz1] <> "MHz_J" <> TextString[Jval] <> "MHz_v170925_v1.h5",
  {resF1vs2D, resF2vs2D, resF3vs2D, resF4vs2D, ωz1arr / (2 Pi), ωz2arr / (2 Pi),
   ωarr / (2 Pi), t1arr}, {"Datasets", {"resF1vs2D", "resF2vs2D",
    "resF3vs2D", "resF4vs2D", "fR1arr", "fR2arr", "farr", "t1arr"}}];
```

Out[∘]=
```
{3510.87, Null}
```

Out[∘]=
```
{2862.44, Null}
```

Out[∘]=
```
{3272.62, Null}
```

Out[∘]=
```
{3256., Null}
```