Multiqubit_v2 Script Directory

Prepared by Ryan Marchildon. Last updated on 09/01/2012.

Essential Scripts:

These are files which need to be kept in the Multiqubit_v2 folder. They contain parameters, instrument creation instances, and function definitions which are routinely accessed by the other scripts.

- setup2.py
- setup3.py
- mixer_offset_meas.py

setup2.py – Several scripts will require this setup file to be run before they can be launched. Running the setup once will store all necessary values/functions etc. in qtlab's memory until either they are overwritten or qtlab is exited. This setup configuration is specifically intended for a particular single-qubit control-board layout → **please refer to the file "1qb Script Series – Control Circuit Setup.pptx" found in the "Multiqubit_v2 Script Documentation" folder.**

This setup file contains:

- The mixer offset values (for both control and readout)
- Imported classes and packages (such as numpy, instr_group, etc)
- Creation of the "AWGs" handle
- Creation of the following instruments (and their GPIB/VICP addresses)
  - LeCroy Scope
  - PSG_Qubit1 (Agilent_E8257D # 1)
  - PSG_Cavity (Agilent_E8257D # 2)
  - yoko_main (Yokogawa voltage controller for main flux bias coil)
  - yoko_mini_1 (Yokogawa voltage controller for mini flux bias coil 1)
- Definitions of sampling times, switch times, the readout reference amplitude (nominally a multiplier of 1) and the readout delay (an offset to ensure that signal propagation times don't result in the readout signal overlapping the control signals)
- Definitions of the sign() function and the slow yokogawa sweep functions

Setup3.py – Several scripts will require this setup file to be run before they can be launched. Running the setup once will store all necessary values/functions etc. in qtlab's memory until either they are overwritten or qtlab is exited. This setup configuration is specifically intended for a particular two-qubit control-board layout → **please refer to the file "2qb Script Series – Control Circuit Setup.pptx" found in the "Multiqubit_v2 Script Documentation" folder.**

This setup file contains:

- The mixer offset values (for both control and readout)
- Imported classes and packages (such as numpy, instr_group, etc)
- Creation of the "AWGs" handle
- Creation of the following instruments (and their GPIB/VICP addresses)
    - LeCroy Scope
    - PSG_Qubit1 (Agilent_E8257D # 1)
    - PSG_Qubit2 (Agilent_E8257D # 2)
    - PSG_Cavity (PhaseMatrix)
    - yoko_main (Yokogawa voltage controller for main flux bias coil)
    - yoko_mini_1 (Yokogawa voltage controller for mini flux bias coil 1)
- Definitions of sampling times, switch times, the readout reference amplitude (nominally a multiplier of 1) and the readout delay (an offset to ensure that signal propagation times don't result in the readout signal overlapping the control signals)
- Definitions of the sign() function and the slow yokogawa sweep functions

mixer_offset_meas.py – This script contains the "measure_mixer_offset" functions called during measurement scripts to normalize the homodyne readout. It is called by other scripts (i.e. not run through qtlab) and is not intended to be modified.

Utilities:

These are files which control particular instruments outside of the normal 'experimental measurement' setting. They are used either as remote controls, for calibration, or for debugging.

- phasematrix_quickcontrol.py
- fsv_scan_power.py
- tek1_set_steady_output.py
- controlboard_mixer_calibrations.py
- readout_mixer_calibrations.py
- QcontrolGUI.py
- retune.py (alpha version)
- test_readout_[…].py

phasematrix_quickcontrol.py – This script acts as a quick remote control for the PhaseMatrix synthesizer, with options to set its output frequency, toggle the frequency reference, and turn the output on or off

.

fsv_scan_power.py – This serves as a launching point for quickly retrieving power data from the R&S FSV (Spectrum Analyser). It can be used for debugging should further changes be made to the "quick_power_scan()" function located in the instrument driver [RS_FSV30.py]

tek1_set_steady_output.py – Used to set a steady voltage level on the AWG (TEK1) so that the resultant mixer output signal can directly (and quickly) be measured as a power on the FSV. Used for mapping ctrl[#]_I/Q_amp parameters to a power level INTO the qubit fastlines. All switches (including readout switch) are kept in the HIGH state.

controlboard_mixer_calibrations.py – This script modulates the TEK AWG with offset triangular waves on the I and Q channels (Ch1 and Ch2 respectively). It is used for calibrating the readout IQ mixer (the offsets are free parameters used for calibration). As on option, a trace is captured from the FSV to permit remote viewing (the frequency span etc must still be set manually at the instrument).

Note that the pulse_input command will modify the waveform to make best use of the TEK resolution. Consequently, be wary that the offset/amplitude that you read off of the TEK display screen might not always correspond to the values you input in this script. It is always best to check this on the scope. If doubt exists, use the built-in waveforms on the TEK to define triangle and constant-level signals.

readout_mixer_calibrations.py – This script modulates the arbstudio AWG with offset triangular waves on the I and Q channels. It is used for calibrating the readout IQ mixer (the offsets are free parameters used for calibration). As an option, it can return a trace of the FSV to permit remote viewing (it only captures the trace; the frequency span etc. must still be set manually at the instrument).

QcontrolGUI.py – This unfinished script was being written as a graphical interface for quickly coding complicated pulse sequences. The idea was to automatically generate the python code (in a manner consistent with existing pulse conventions) corresponding to several user-defined events such as arbitrary Qubit rotations. The pulse sequence would be graphically displayed using quantum circuit symbolism, and the generated pulse sequence code could then be directly copied/pasted into a measurement script. Expansion was allowed for more complicated routines such as CNOT gates etc.

Note from creator: this script is still in the early stages, but excluding the visual quantum-circuit representations of pulse sequences it wouldn't take too much more to bring it to a point where output code can be generated. There is a GUI toolbox in Python that makes projects of this kind fairly straightforward ***– please see the wiki entry for more details****. –Ryan M*

retune.py (alpha version) – This is an as-of-yet incomplete script based on the multiqubit Mathematica file "2qb Coil-to-Frequency Map". It is intended to be a tool allowing the user to quickly retune qubits to their symmetry point based on a measured correspondence (during BSWEEPS) between Yokogawa voltages and effective flux through the qubit loops (please refer to the Mathematica file). The idea is that if the qubits suffer from flux drift (effectively an offset due to, for example, variations in the Earth's magnetic field), then one can quickly retune the system by simply performing pulsed spectroscopy at the present Yokogawa voltage settings and inputting the results into the retune script.

test_readout_[...].py – These scripts can be used to test the performance and linearity of the readout circuit when readout mixer is calibrated. Please see the script files themselves for details on their specific functionalities.

Single-Qubit (1qb) Script Group:

The "1qb" script group must be launched after **"setup2.py"** is run in the current session of qtlab. These scripts are intended for experimentation using the single-qubit instrument/control board configuration (please see document "1qb Script Series – Control Circuit Setup.pptx"). It can still be used to examine multiple qubits on the same device – simply change the fastline connected to the mixer output!

PART 1: *DEVICE & QUBIT CHARACTERIZATION* ----------

- 1qb_BSWEEP_sweep_yoko_sweep_PSGcavity.py
- 1qb_BSWEEP_sweep_yoko_sweep_PSGcavity_v2.py
- 1qb_BSWEEP_sweep_yoko_sweep_VNA.py
- 1qb_CWSPECTROSCOPY_sweep_CTRLamp_sweep_PSGQubit1.py
- 1qb_CWSPECTROSCOPY_sweep_PSGcavity_sweep_PSGQubit1.py
- 1qb_CWSPECTROSCOPY _sweep_PSGQubit1.py
- 1qb_CWSPECTROSCOPY_sweep_yoko_sweep_PSGQubit1.py
- 1qb_CWSPECTROSCOPY_sweep_yoko_sweep_PSGQubit1_with_VNA.py
- 1qb_HISTOGRAM.py
- 1qb_PIPULSE_sweep_PSGCavity.py
- 1qb_PULSEDSPECTROSCOPY_sweep_PSGQubit1.py

- 1qb_PULSEDSPECTROSCOPY_sweep_yoko_sweep_PSGQubit1.py
- 1qb_RABI_sweep_trabi.py
- 1qb_RABI_sweep_trabi_sweep_CTRLamp.py
- 1qb_RAMSEY
- 1qb_READOUToptimization_sweep_Pwr_sweep_PSGCavity.py
- 1qb_SPINECHO.py
- 1qb_T1_relaxation.py

1qb_BSWEEP_sweep_yoko_sweep_PSGcavity.py – This script sweeps the Cavity synthesizer frequency (using PSG_Cavity) AND the qubit bias (using a Yokogawa V-source) while: applying a constant voltage level to the readout mixer; keeping the readout switch open; and measuring I and Q averages from the LeCroy Oscilloscope. It is used to determine how the cavity resonance changes with magnetic flux, thereby allowing the location of the qubit branches to be determined.

Note that at this time the qubit drive lines (i.e. fastlines) must NOT be receiving a signal. Also, this script is coded specifically for sweeping the main coil – for sweeping the minicoils, please see v2.

1qb_BSWEEP_sweep_yoko_sweep_PSGcavity_v2.py – This script sweeps the Cavity synthesizer frequency (using PSG_Cavity) AND the qubit bias (using a selected Yokogawa V-source) while: applying a constant voltage level to the readout mixer; keeping the readout switch open; and measuring I and Q averages from the LeCroy Oscilloscope. It is used to determine how the cavity resonance changes with magnetic flux, thereby allowing the location of the qubit branches to be determined.

Note that at this time the qubit drive lines (i.e. fastlines) must NOT be receiving a signal. The user chooses between the main coil and mini coil 1 for the Yokogawa-controlled bias sweep.

1qb_BSWEEP_sweep_yoko_sweep_VNA.py – This script is used for measuring microwave transmission with the VNA while sweeping qubit bias (using the Yokogawas). It is used to get an initial rough estimate of the location of the qubit branch/symmetry point and corresponding cavity resonances.

Note that at this time the qubit drive lines (i.e. fastlines) must NOT be receiving a signal.

1qb_CWSPECTROSCOPY_sweep_CTRLamp_sweep_PSGQubit1.py – This script performs a continuous wave (CW) spectroscopy sweep that scans through a range of qubit drive frequencies at different drive powers. It can be used to investigate the power dependence of peak widths and amplitudes to, for example, determine saturation power or estimate Rabi frequency. All switches (including readout switch) are kept in the HIGH state.

1qb_CWSPECTROSCOPY_sweep_PSGcavity_sweep_PSGQubit1.py – This script is used to directly investigate the dispersive shift of the qubit using continuous wave (CW) spectroscopy. It sweeps both the cavity and the Qubit drive signals and is intended to be operated at a flux bias corresponding to the qubit symmetry point. All switches (including readout switch) are kept in the HIGH state.

Note: the effects of the dispersive shift should be evident by their appearance only when the PSG_Qubit drive is near the qubit resonance frequency (at the symmetry point).

1qb_CWSPECTROSCOPY_sweep_PSGQubit1.py – This script performs a continuous wave spectroscopy sweep (scanning over qubit drive frequencies) using the control circuit at a single coil voltage (flux bias) setpoint. All switches (including readout switch) are kept in the HIGH state.

1qb_CWSPECTROSCOPY_sweep_yoko_sweep_PSGQubit1.py – This script performs a continuous wave spectroscopy sweep (scanning over qubit drive frequencies) using the control circuit over a range of coil voltages (flux biases). All switches (including readout switch) are kept in the HIGH state at all times.

1qb_CWSPECTROSCOPY_sweep_yoko_sweep_PSGQubit1_with_VNA.py - This script performs continuous wave (CW) spectroscopy, sweeping both the qubit drive signal and Yokogawas, using the VNA to probe the cavity at a single selected frequency. It can be used to provide an initial analysis of the qubit's energy-level splitting near its symmetry point.

1qb_HISTOGRAM.py - This script is used to generate histograms to determine our |1> state preparation fidelity. It obtains sliced averages from the scope (where each data point corresponds to the average over a single marker window) for [A.] a qubit excitation with a Pi pulse and [B.] no qubit excitation (i.e. ground state).

1qb_PIPULSE_sweep_PSGCavity.py - This script puts the qubit in the excited state and sweeps PSG cavity to locate dispersively shifted branches near the qubit symmetry point.

1qb_PULSEDSPECTROSCOPY_sweep_PSGQubit1.py - This script performs a standard pulsed spectroscopy sweep (scanning over qubit drive frequency) at a single coil voltage. When the drive signal is in resonance with the qubit, a dip will be seen in the homodyne output voltage – this is due to the dispersive shift the qubit induces on the cavity resonance when it is in the excited state.

1qb_PULSEDSPECTROSCOPY_sweep_yoko_sweep_PSGQubit1.py - This script performs a standard pulsed spectroscopy sweep (scanning over qubit drive frequency) over a range of coil voltages. As the voltage is swept, the observed qubit resonance frequency will follow a parabolic pattern centered at the symmetry point (where the qubit frequency is at a minimum).

1qb_RABI_sweep_trabi.py - This script measures the Rabi oscillations of the qubit when subjected to an on-resonance drive frequency at its symmetry point. The oscillations will exponentially decay with an envelope characterized by the T1 (energy relaxation) time. The Rabi frequency, which is a function of both detuning and drive power, can be used to determine suitable settings for Pi and Half-Pi pulses on either quadrature.

1qb_RABI_sweep_trabi_sweep_CTRLamp.py - This script performs Rabi oscillation experiments for a range of control-board mixer voltages to characterize the relation between Rabi frequency and drive power.

1qb_RAMSEY.py - This script executes a standard Ramsey measurement (free induction decay) to measure T2 (dephasing) times. The output should depict a decaying sinusoidal function, where the oscillation frequency is approximately equal to the detuning from the qubit resonance, and the decay time is characterized by T2.

1qb_READOUToptimization_sweep_Pwr_sweep_PSGCavity.py - Use this script to optimize readout by sweeping cavity power and frequency while measuring the readout signal. One sweep is done with an excitation signal at the qubit frequency (@ symmetry point); the other is done without an excitation tone. The cavity driving conditions giving rise to largest readout signal difference between the two sweeps [i.e. sqrt((Qexc - Qgnd)^2 + (Iexc - Ignd)^2)] is optimal

(i.e. the settings at which the dispersive shift gives the greatest contrast between the |0> and |1> states).

1qb_SPINECHO.py - Performs a spin-echo Ramsey measurement that isolates the intrinsic coherence time via a spin echo pulse. The effect of the spin-echo will be to remove any steady-state dephasing contributions.

1qb_T1_relaxation.py - This script measures the T1 relaxation time of a single qubit by measuring the probability of the qb being in the excited state as a function of delay time (i.e. permitted relaxation time) between the end of the excitation (Pi-pulse) and the beginning of readout. The output signal should resemble a decaying exponential: A*(1 - exp(-t/T1))

Note that if the readout times are long, the results will reflect some relaxation occuring DURING readout.

PART 2: *QUANTUM TOMOGRAPHY & ADVANCED GATE ROUTINES*----------

- 1qb_QST_setup.py
- 1qb_QST_main.py
- 1qb_QPT_setup.py
- 1qb_QPT_main.py

1qb_QST_setup.py – This script defines the rotations and measurement operators necessary to perform Quantum-State Tomography on a single-qubit system. It must be run prior to launching any QST experiments. Pi and Half-Pi pulse amplitudes/durations must be defined for both I and Q quadratures for the qubit being tested. Please refer to "2qb_QST_TimingDiagram.pptx" found in the subfolder "Multiqubit_v2 Script Documentation" (contains all relevant 1qb timing information).

The measurement Pre-Rotations are listed below:

Operator

M1: I
M2: Rx(Pi/2)
M3: Ry(Pi/2)

*!*IMPORTANT CONVENTIONS - QUADRATURE CONTROL:
I = (+), Q = (0): -Rx (CCW)
I = (-), Q = (0): +Rx (CW)
I = (0), Q = (+): +Ry (CW)
I = (0), Q = (-): -Ry (CCW)

1qb_QST_main.py – Please refer to "2qb_QST_TimingDiagram.pptx" found in the subfolder "Multiqubit_v2 Script Documentation" (contains all relevant 1qb timing information). This script conducts automated quantum state tomography (QST) for a single-qubit system. It should be used as a template for particular experiments by saving it as a new file and modifying the sequence of gating/entanglement signals.

1qb_QPT_setup.py – This script defines the rotations, preparation states, and measurement operators necessary to perform Quantum Process Tomography on a single-qubit system.

Please refer to "1qb_QPT_TimingDiagram.pptx" found in the subfolder "Multiqubit_v2 Script Documentation".

Pi and Half-Pi pulse amplitudes/durations must be defined for both I and Q quadratures for the qubit being tested.

There are three main components to the control pulse sequence:

1. The state is initialized in either the |0>,|1>,|+> or |i> configuration, where we have:

|+> = 1/sqrt(2)*(|0> + |1>)

|i> = 1/sqrt(2)*(|0> + i|1>)

2. The system is evolved through whichever unitary process we wish to characterize (i.e. the quantum "black box")

3. Standard Quantum State Tomography (QST) is performed on the output to reconstruct physical density matrices. Recall that QST requires a number of pre-rotations to inspect the qubit along each measurement axis.

The measurement Pre-Rotations are listed below

Operator	PreRot	Form
M1:	I	BI<I> + BZ<Z>
M2:	X(Pi/2)	BI<I> + BZ<Y>
M3:	Y(Pi/2)	BI<I> - BZ<X>

IMPORTANT CONVENTIONS - QUADRATURE CONTROL:
I = (+), Q = (0): -Rx (CCW)
I = (-), Q = (0): +Rx (CW)
I = (0), Q = (+): +Ry (CW)
I = (0), Q = (-): -Ry (CCW)
An "X" gate (Bit-Flip) corresponds to Ry(Pi).

1qb_QPT_main.py - This script conducts automated quantum process tomography (QPT) for a single qubit. It should be used as a template for particular experiments by saving it as a new file and modifying the sequence of gating/entanglement signals. Please refer to "1qb_QPT_TimingDiagram.pptx" found in the subfolder "Multiqubit_v2 Script Documentation".

Two-Qubit (2qb) Script Group:

The "2qb" script group must be launched after **"setup3.py"** is run in the current session of qtlab. These scripts are intended for experimentation using the two-qubit instrument/control board configuration (please see document "2qb Script Series – Control Circuit Setup.pptx").

PART 1: *DEVICE & QUBIT CHARACTERIZATION*----------

- 2qb_BSWEEP_sweep_yoko_sweep_PSGcavity.py
- 2qb_BSWEEP_sweep_yoko_sweep_VNA.py
- 2qb_CWSPECTROSCOPY_sweep_CTRLamp_sweep_PSGQubit.py
- 2qb_CWSPECTROSCOPY_sweep_PSGcavity_sweep_PSGQubit.py
- 2qb_CWSPECTROSCOPY _sweep_PSGQubit.py
- 2qb_CWSPECTROSCOPY_sweep_yoko_sweep_PSGQubit.py
- 2qb_CWSPECTROSCOPY_sweep_yoko_sweep_PSGQubit_with_VNA.py
- 2qb_PULSEDSPECTROSCOPY_sweep_PSGQubit.py
- 2qb_PULSEDSPECTROSCOPY_sweep_yoko_sweep_PSGQubit.py
- 2qb_RABI_sweep_trabi.py
- 2qb_RABI_sweep_trabi_sweep_CTRLamp.py
- 2qb_RAMSEY
- 2qb_SPINECHO.py
- 2qb_T1_relaxation.py

2qb_BSWEEP_sweep_yoko_sweep_PSGcavity.py – This script sweeps the Cavity synthesizer frequency (using PSG_Cavity) AND the qubit bias (using a selected Yokogawa V-source) while: applying a constant voltage level to the readout mixer; keeping the readout switch open; and measuring I and Q averages from the LeCroy Oscilloscope. It is used to determine how the cavity

resonance changes with magnetic flux, thereby allowing the location of the qubit branches to be determined. The experiment can toggle between Qubit 1 and Qubit 2.

Note that at this time the qubit drive lines (i.e. fastlines) must NOT be receiving a signal. The user chooses between the main coil and mini coil 1 for the Yokogawa-controlled bias sweep.

2qb_BSWEEP_sweep_yoko_sweep_VNA.py – This script is used for measuring microwave transmission with the VNA while sweeping qubit bias (using the Yokogawas). It is used to get an initial rough estimate of the location of the qubit branch/symmetry point and corresponding cavity resonances. The experiment can toggle between Qubit 1 and Qubit 2.

Note that at this time the qubit drive lines (i.e. fastlines) must NOT be receiving a signal.

2qb_CWSPECTROSCOPY_sweep_CTRLamp_sweep_PSGQubit.py – This script performs a continuous wave (CW) spectroscopy sweep that scans through a range of qubit drive frequencies at different drive powers. It can be used to investigate the power dependence of peak widths and amplitudes to, for example, determine saturation power or estimate Rabi frequency. All switches (including readout switch) are kept in the HIGH state. The experiment can toggle between Qubit 1 and Qubit 2.

2qb_CWSPECTROSCOPY_sweep_PSGcavity_sweep_PSGQubit.py – This script is used to directly investigate the dispersive shift of the qubit using continuous wave (CW) spectroscopy. It sweeps both the cavity and the Qubit drive signals and is intended to be operated at a flux bias corresponding to the qubit symmetry point. All switches (including readout switch) are kept in the HIGH state. The experiment can toggle between Qubit 1 and Qubit 2.

Note: the effects of the dispersive shift should be evident by their appearance only when the PSG_Qubit drive is near the qubit resonance frequency (at the symmetry point).

2qb_CWSPECTROSCOPY_sweep_PSGQubit.py – This script performs a continuous wave spectroscopy sweep (scanning over qubit drive frequencies) using the control circuit at a single coil voltage (flux bias) setpoint. All switches (including readout switch) are kept in the HIGH state. The experiment can toggle between Qubit 1 and Qubit 2.

2qb_CWSPECTROSCOPY_sweep_yoko_sweep_PSGQubit.py – This script performs a continuous wave spectroscopy sweep (scanning over qubit drive frequencies) using the control circuit over a

range of coil voltages (flux biases). All switches (including readout switch) are kept in the HIGH state at all times. The experiment can toggle between Qubit 1 and Qubit 2.

2qb_CWSPECTROSCOPY_sweep_yoko_sweep_PSGQubit_with_VNA.py - This script performs continuous wave (CW) spectroscopy, sweeping both the qubit drive signal and Yokogawas, using the VNA to probe the cavity at a single selected frequency. It can be used to provide an initial analysis of the qubit's energy-level splitting near its symmetry point. The experiment can toggle between Qubit 1 and Qubit 2.

2qb_PULSEDSPECTROSCOPY_sweep_PSGQubit.py - This script performs a standard pulsed spectroscopy sweep (scanning over qubit drive frequency) at a single coil voltage. When the drive signal is in resonance with the qubit, a dip will be seen in the homodyne output voltage – this is due to the dispersive shift the qubit induces on the cavity resonance when it is in the excited state. The experiment can toggle between Qubit 1 and Qubit 2.

2qb_PULSEDSPECTROSCOPY_sweep_yoko_sweep_PSGQubit.py - This script performs a standard pulsed spectroscopy sweep (scanning over qubit drive frequency) over a range of coil voltages. As the voltage is swept, the observed qubit resonance frequency will follow a parabolic pattern centered at the symmetry point (where the qubit frequency is at a minimum). The experiment can toggle between Qubit 1 and Qubit 2.

2qb_RABI_sweep_trabi.py - This script measures the Rabi oscillations of the qubit when subjected to an on-resonance drive frequency at its symmetry point. The oscillations will exponentially decay with an envelope characterized by the T1 (energy relaxation) time. The Rabi frequency, which is a function of both detuning and drive power, can be used to determine suitable settings for Pi and Half-Pi pulses on either quadrature. The experiment can toggle between Qubit 1 and Qubit 2.

2qb_RABI_sweep_trabi_sweep_CTRLamp.py - This script performs Rabi oscillation experiments for a range of control-board mixer voltages to characterize the relation between Rabi frequency and drive power. The experiment can toggle between Qubit 1 and Qubit 2.

2qb_RAMSEY.py - This script executes a standard Ramsey measurement (free induction decay) to measure T2 (dephasing) times. The output should depict a decaying sinusoidal function,

where the oscillation frequency is approximately equal to the detuning from the qubit resonance, and the decay time is characterized by T2. The experiment can toggle between Qubit 1 and Qubit 2.

2qb_SPINECHO.py - Performs a spin-echo Ramsey measurement that isolates the intrinsic coherence time via a spin echo pulse. The effect of the spin-echo will be to remove any steady-state dephasing contributions. The experiment can toggle between Qubit 1 and Qubit 2.

2qb_T1_relaxation.py - This script measures the T1 relaxation time of a single qubit by measuring the probability of the qb being in the excited state as a function of delay time (i.e. permitted relaxation time) between the end of the excitation (Pi-pulse) and the beginning of readout. The output signal should resemble a decaying exponential: A*(1 - exp(-t/T1)). The experiment can toggle between Qubit 1 and Qubit 2.

Note that if the readout times are long, the results will reflect some relaxation occuring DURING readout.

PART 2: *QUANTUM TOMOGRAPHY & ADVANCED GATE ROUTINES*----------

- 2qb_QST_setup.py
- 2qb_QST_main.py
- 2qb_SELECTIVEDARKENING_v1.py (beta)

2qb_QST_setup.py – Please refer to "2qb_QST_TimingDiagram.pptx" found in the subfolder "Multiqubit_v2 Script Documentation". It must be launched prior to running a 2-qubit QST experiment. Note that Pi and Half-Pi pulses must be defined on the I and Q quadratures for both Qubits. This script defines the rotations and measurement operators necessary to perform Quantum-State Tomography on a 2-qubit system.

The measurement Pre-Rotations are listed below, where "(.)" is symbolic for the tensor product:

Operator QB1(.)QB2

M1: I(.)I
M2: Rx(Pi)(.)I
M3: I(.)Rx(Pi)
M4: Rx(Pi/2)(.)I
M5: Rx(Pi/2)(.)Rx(Pi/2)

M6: Rx(Pi/2)(.)Ry(Pi/2)
M7: Rx(Pi/2)(.)Rx(Pi)
M8: Ry(Pi/2)(.)I
M9: Ry(Pi/2)(.)Rx(Pi/2)
M10: Ry(Pi/2)(.)Ry(Pi/2)
M11: Ry(Pi/2)(.)Rx(Pi)
M12: I(.)Rx(Pi/2)
M13: Rx(Pi)(.)Rx(Pi/2)
M14: I(.)Ry(Pi/2)
M15: Rx(Pi)(.)Ry(Pi/2)

*!*IMPORTANT CONVENTIONS - QUADRATURE CONTROL:
I = (+), Q = (0): -Rx (CCW)
I = (-), Q = (0): +Rx (CW)
I = (0), Q = (+): +Ry (CW)
I = (0), Q = (-): -Ry (CCW)
An "X" gate (Bit-Flip) corresponds to Ry(Pi).

2qb_QST_main.py – Please refer to "2qb_QST_TimingDiagram.pptx" found in the subfolder "Multiqubit_v2 Script Documentation".This script conducts automated quantum state tomography (QST) for a two-qubit system. It shouldbe used as a template for particular experiments by saving it as a new file and modifying the sequence of gating/entanglement signals.

2qb_SELECTIVEDARKENING_v1.py (beta)- This script is meant to serve as a template for Selective Darkening experiments whereby a diagonal element of the Hamiltonian corresponding to one of the four transitions between 2qb basis states is tuned to zero. This is achieved by simultaneously driving both qubits with the same frequency but different relative amplitudes and phases, depending on which transition is to be suppressed. Both qubits must be tuned to their symmetry points. [Note - see "ratio_optimize.py", which was created for an early draft of this script. Depending on which parameters are being swept, it may (or may not) come in handy]

Note: delta_phase = (Qubit2 Control Signal Phase) - (Qubit1 Control Signal Phase)

-----Qubit 1-----
Transition 1A: |00> <--> |10>
Suppressed by: A2overA1 = h*(Qubit1_Freq - Qubit2_Freq)/J (required driving flux ratios), delta_phase = Pi (required phase difference between flux drives)

Transition 1B: |01> <--> |11>
Suppressed by: A2overA1 = h*(Qubit1_Freq - Qubit2_Freq)/J, delta_phase = 0

#-----Qubit 2-----
Transition 2A: |00> <--> |01>
Suppressed by: A2overA1 = J/(h*(Qubit1_Freq - Qubit2_Freq)), delta_phase = 0

Transition 2B: |10> <--> |11>
suppressed by: A2overA1 = J/(h*(Qubit1_Freq - Qubit2_Freq)), delta_phase = Pi