PyMeasRF: Automating RF Device Measurements Using Python

Jackson Anderson and Dana Weinstein
HybridMEMS Laboratory, Purdue University, West Lafayette, IN

HYBRIDMEMS

Package Goal and Objectives

Challenge: Test and Measurement (T&M) manufacturer's GUI solutions are often limited to specific use cases and locked behind software option licenses that can cost thousands extra. Alternatively, engineers may spend valuable time reading over individual equipment programmer manuals to write code for specific instruments using Standard Commands for Programmable Instruments (SCPI) syntax.

Goals:
1. Enable electrical characterization of devices that would not otherwise be feasible.
2. Create a Python package that abstracts away SCPI programming and provides in-line guidance through docstrings, speeding development time.

Package Structure and Limitations

PyMeasRF has three general layers:
1. **Physical Instruments**: Python bindings for instrument SCPI commands
2. **Compound Instruments**: Complex combinations of instruments to create simple interface for common tests. For example:
   - **pnaSMU**: S parameter measurements, iterating through biasing on arbitrary number of DC sources.
   - **smuMeas**: coordinated DC bias and current measurements (parametric analysis of semiconductor devices).
3. **Test Sequences**: Combinations of compound instrument tests and/or short scripts directly using physical instruments.

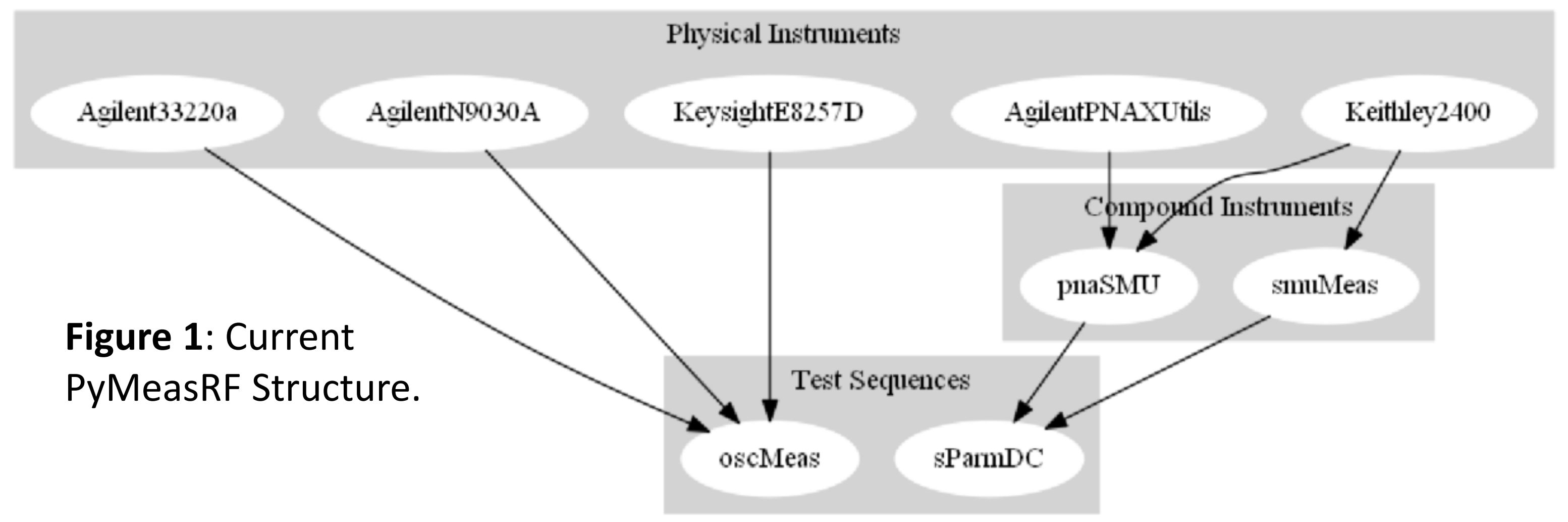


Figure 1: Current PyMeasRF Structure.

Development Challenges for Test and Measurement Equipment:
- **Instrument Complexity:**
  1. S parameter measurements have many parameters, many of which are set at time of user calibration – non-trivial to perform an instrument reset in Python to ensure a clean starting slate.
  2. Newer instruments have a variety of software options and software version numbers that change the allowable set of SCPI commands.
- **CI and Unit Testing:** How do package managers verify functionality of physical hardware they don't have? Virtual instruments?
- **Usability:** What is the best way to make future development accessible for engineers who may have little formal training?
- **Ecosystem [Dis]Unity:** Tens (if not hundreds) of packages exist that interface with lab equipment in various ways, all of which approach it slightly differently – no unified approach for building drivers (see LabPy/labpy-discussion Issue #23 on GitHub).

Case 1: Oscillator Injection Locking

Motivation: Under the right conditions, oscillators will lock on to nearby frequency signals. Understanding this behaviour is critical to harnessing it for neuromorphic computing [1], or for preventing it in highly scaled RF integrated circuits.

Initial investigations carried out on a single oscillator test board, with injection signal delivered via transistor current modulation.

Challenges:
- Measurement requires several pieces of test equipment, including a spectrum analyzer (oscillator output), power supply, and function generator (injected signal).
- Want to investigate injection locking behaviour as a function of injection frequency (N, multiple of oscillator fundamental frequency of 16 MHz), injection AC amplitude, and injection DC offset, requiring **3,894** spectrum Measurements.

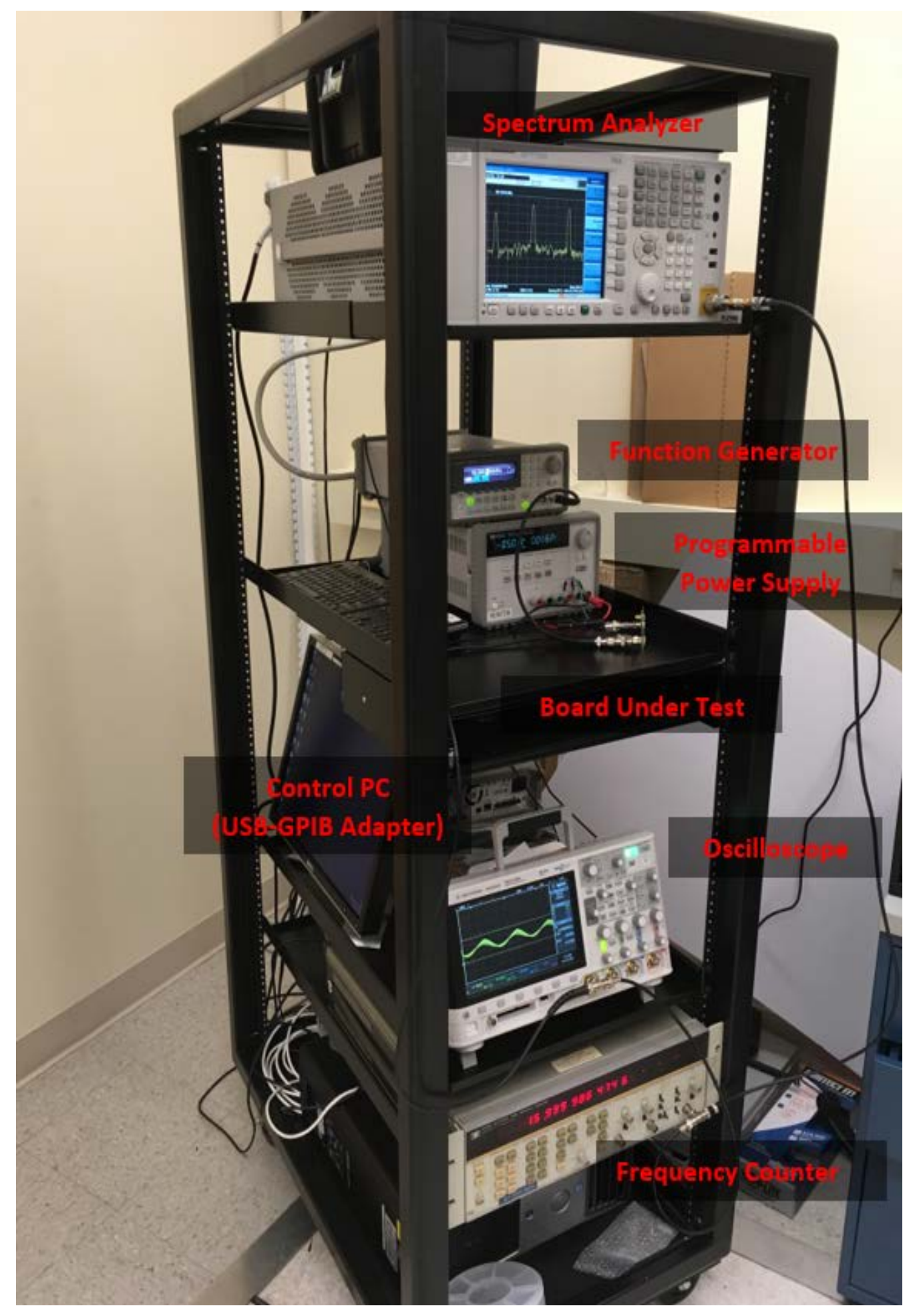
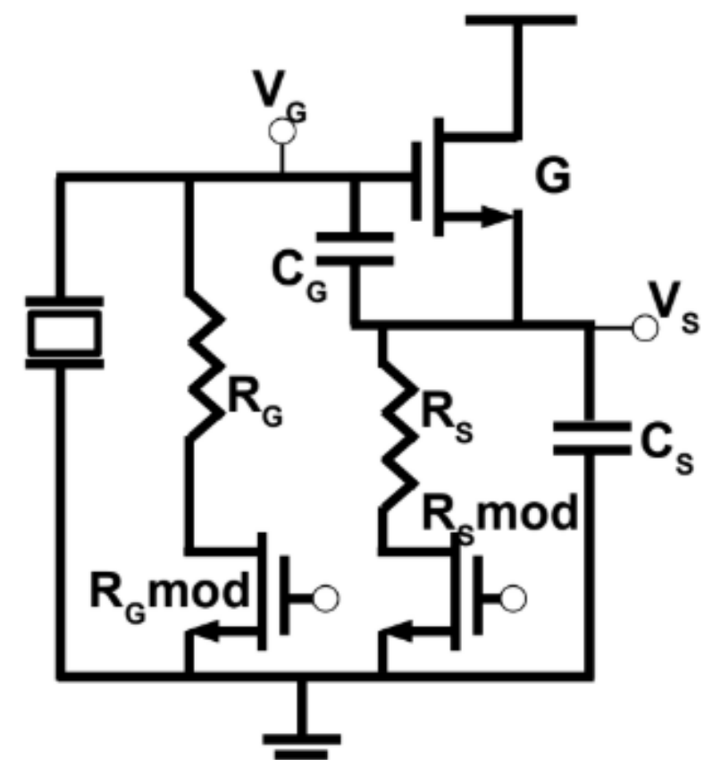
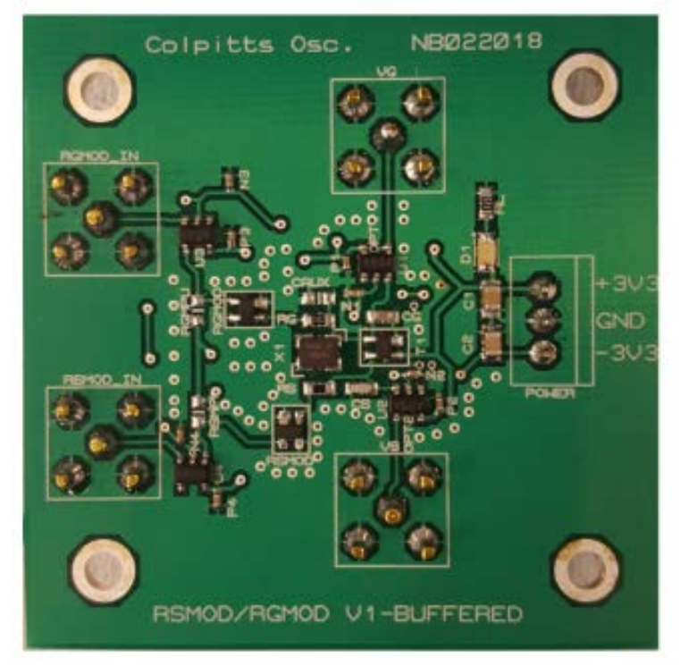


Figure 2: Oscillator circuit (below) and test setup (above).





Solution: Measurement automated in <100 lines of code using developed instrument bindings (oscMeas.py), less than the ~150 lines of analysis code (injectionLockPlot.py). Data obtained over the course of 51 hours.

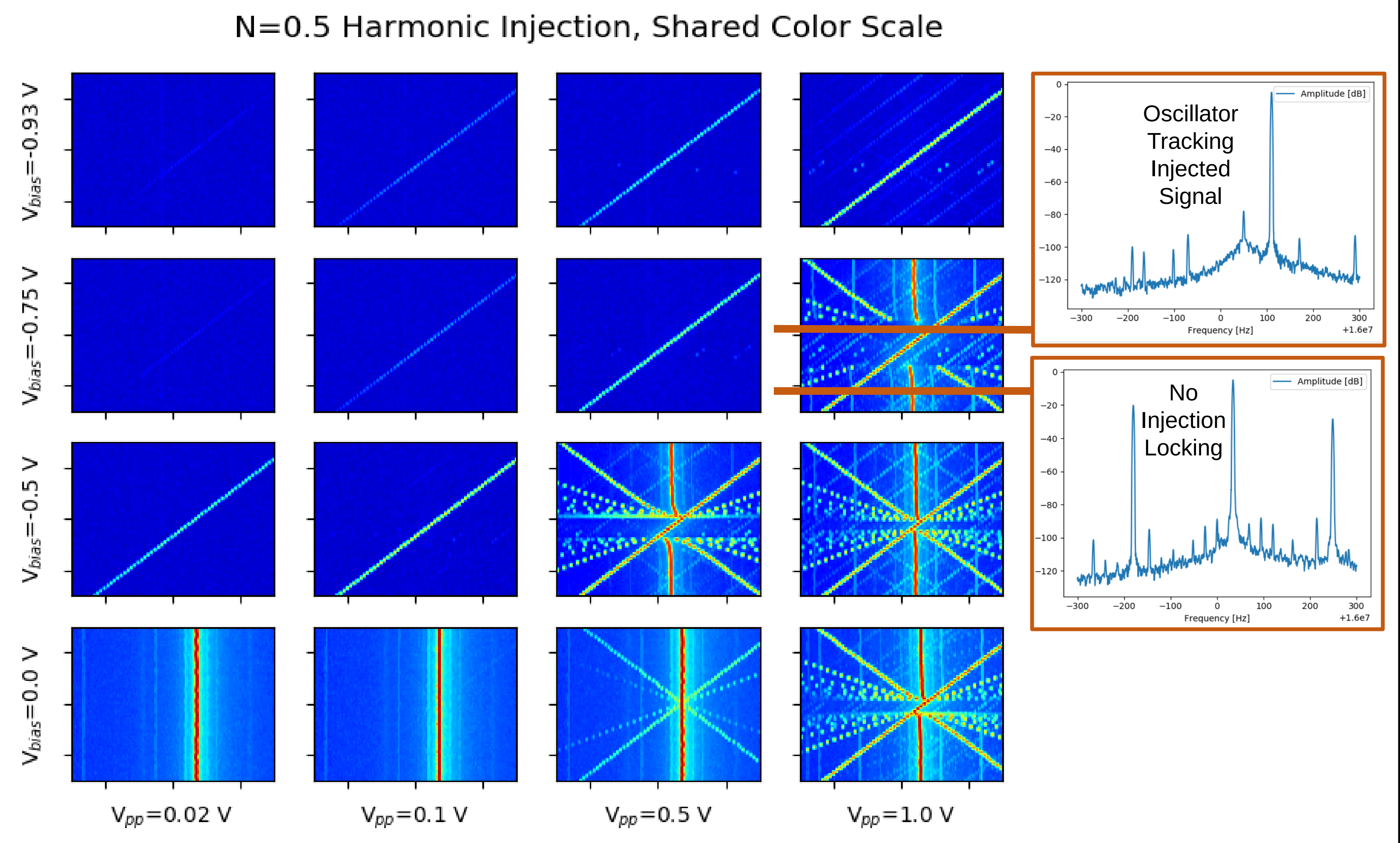
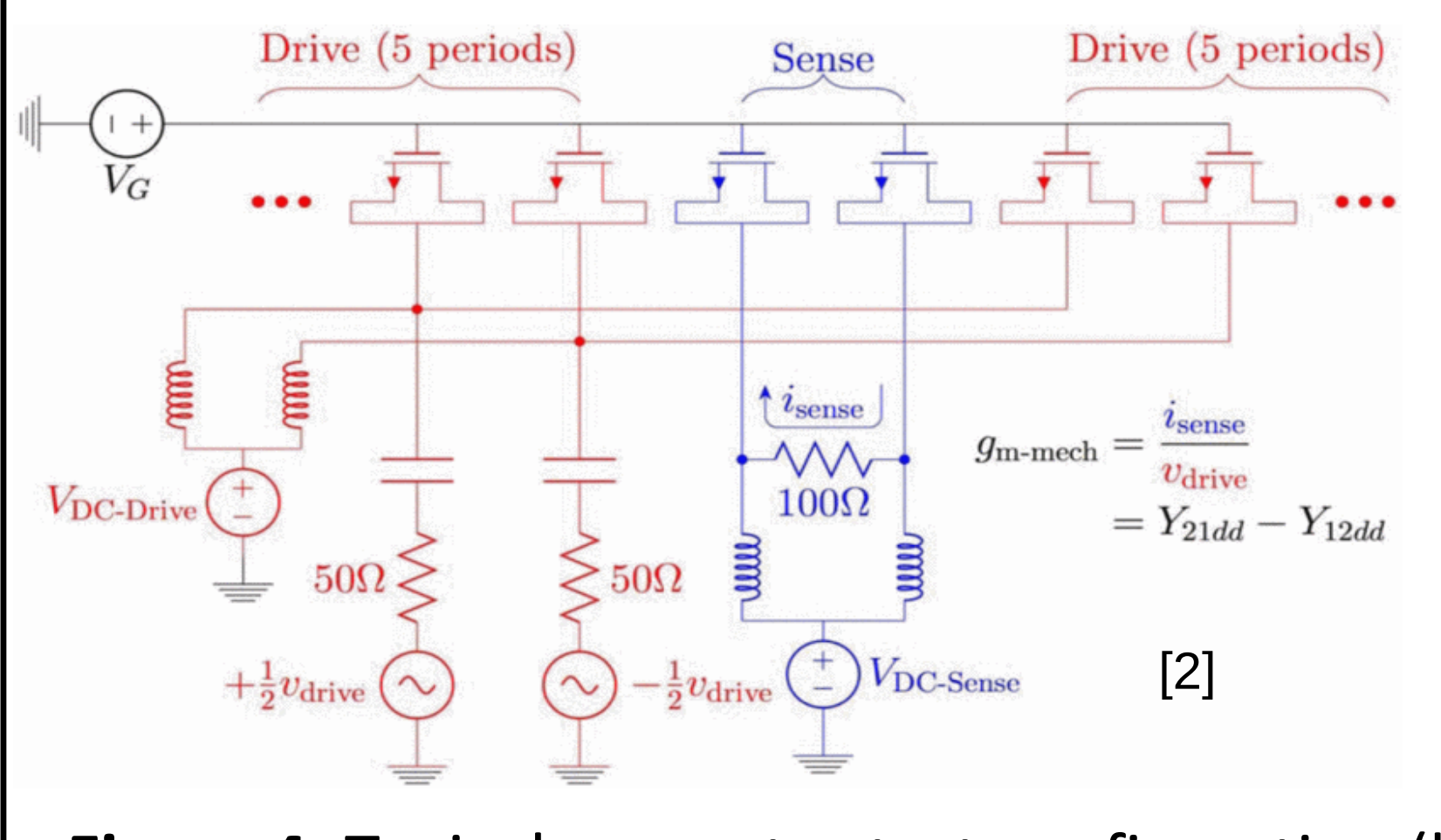
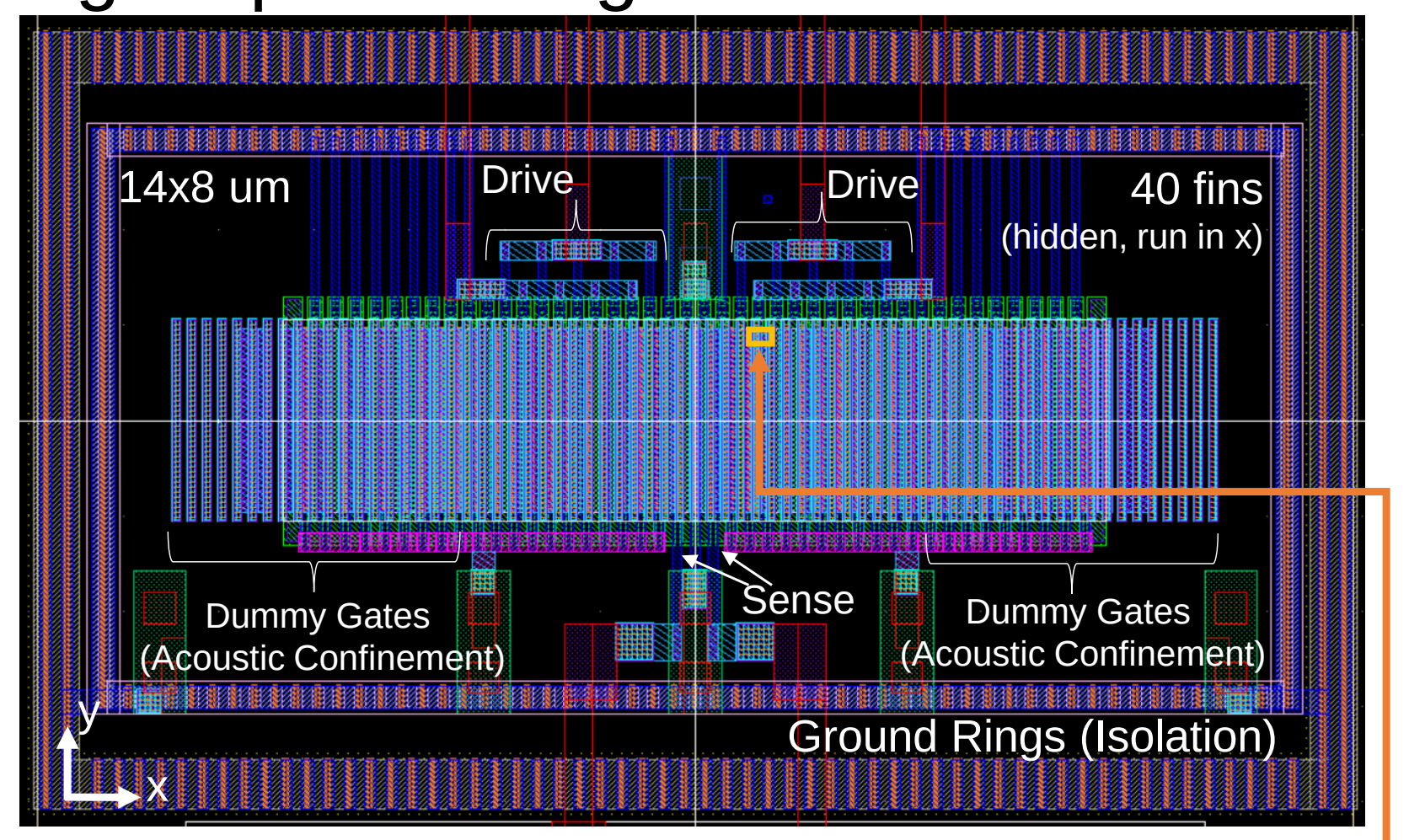


Figure 3: Oscillator output power as a function of frequency (x-axis) and injection frequency (y-axis), tiled over injection voltage dc biases and ac magnitudes.

Case 2: Active Resonator Testing

Motivation: Active solid state resonators are being investigated as a way to implement acoustic devices in leading-edge CMOS nodes for low-power radios, local clock signals, and analog signal processing.




14x8 um
Drive
Drive
40 fins (hidden, run in x)
Dummy Gates (Acoustic Confinement)
Sense
Dummy Gates (Acoustic Confinement)
Ground Rings (Isolation)
y
x

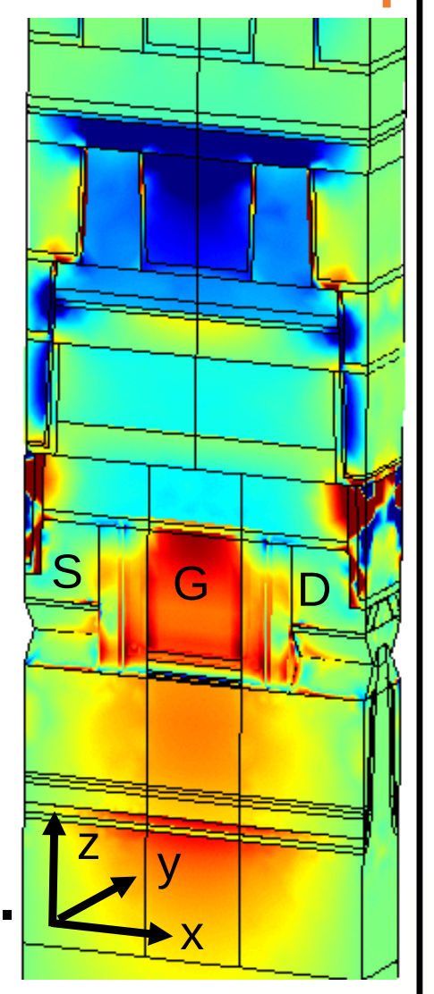
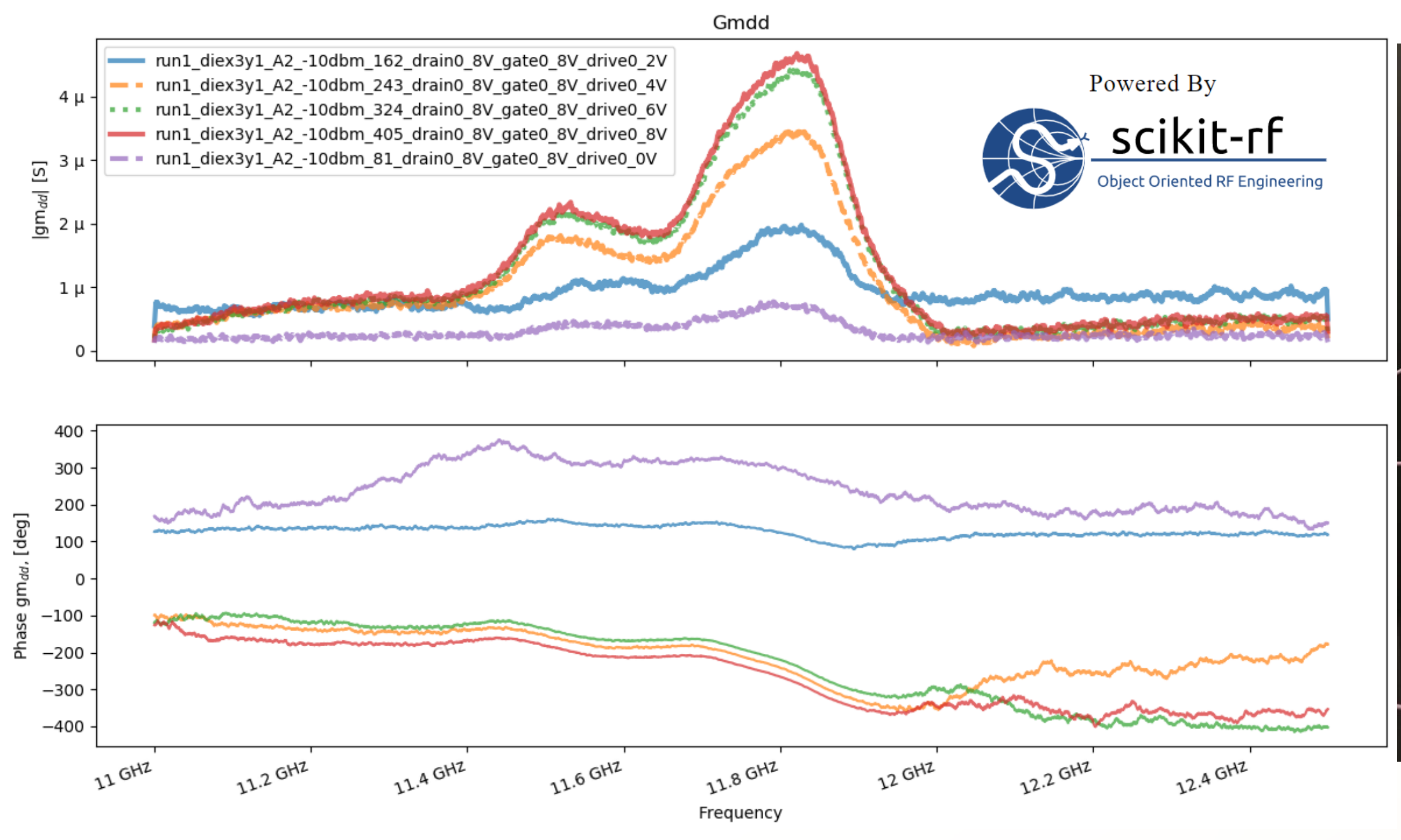


Figure 4: Typical resonator test configuration (left), overhead view of device layout (right), and 3D x-section of orange box, showing acoustic mode in transistor channel (below).

Challenges:
- RF performance dependent on DC biasing – need three SMUs and a Network Analyzer for characterization.
- A single RF sweep can take 15 minutes, single device requires up to 81 sweeps

Solution: Automated DC biasing allows for days-long measurements without any operator intervention. Ability to use arbitrary number of SMUs allows flexibility for use on other devices.



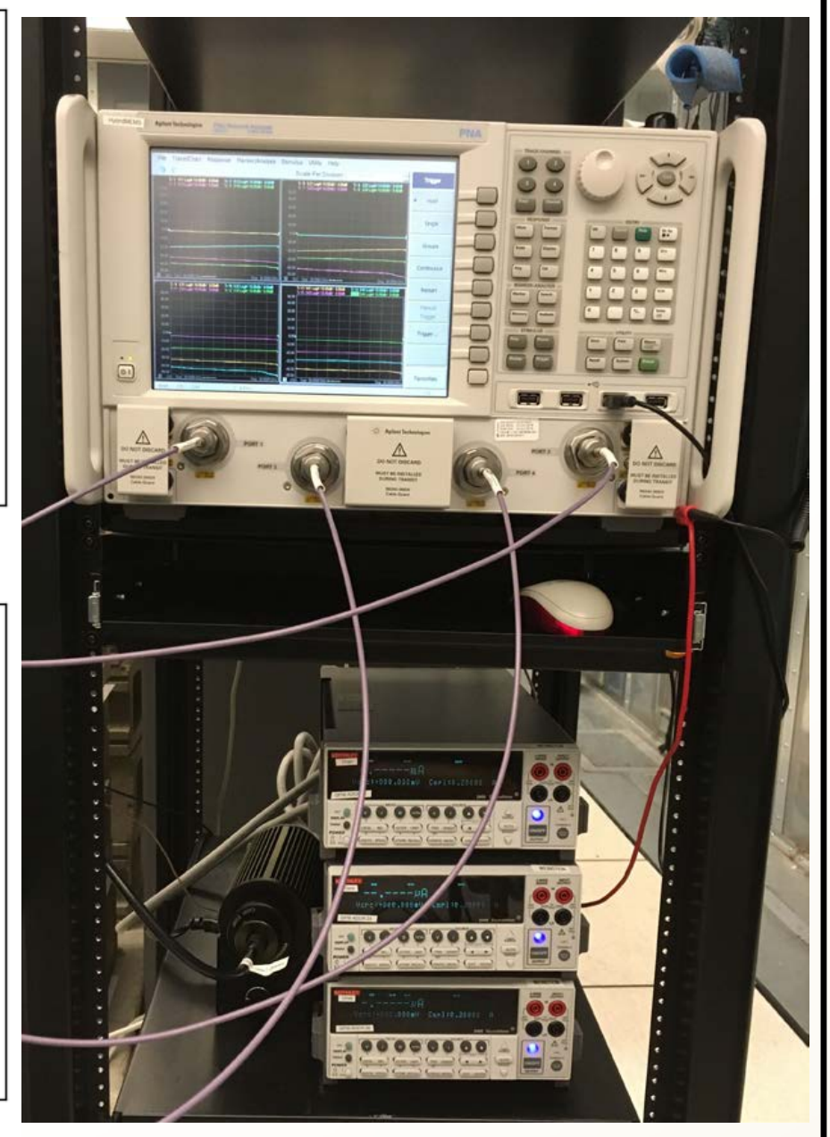


Figure 5: Device response as a function of sense transistor drain bias (left) and test setup (right), consisting of a Keysight PNA and three Keithley SMUs.

Conclusions

A variety of RF device measurements have been successfully automated using python-based instrument wrappers, enabling drastically increased test efficiency and unlocking new insights through vast data collection.

Future work:
1. Evaluate compatibility with existing Python packages – can it be merged?
2. Implement generic classes for categories of common equipment.
3. Implement more robust error handling and unit testing.

References

[1] Nikonov, Dmitri E, Ian A Young, and George I Bourianoff. "Convolutional Networks for Image Processing by Coupled Oscillator Arrays," n.d., 23.
[2] Bahr, B., Y. He, Z. Krivokapic, S. Banna, and D. Weinstein. "32GHz Resonant-Fin Transistors in 14nm FinFET Technology." In *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*, 348–50, 2018. https://doi.org/10.1109/ISSCC.2018.8310327.

Many thanks to all the members of HybridMEMS Lab, especially Yanbo He, Udit Rawat, Shreyas Shah, and Bichoy Bahr who were/are involved in the projects mentioned as case studies. Thanks also to the Rhoads research group, who collaborate on the injection locking project, and to the Birck Nanotechnology Center at Purdue University.
This work was supported in part by the DARPA UPSIDE program.
Prepared for Scipy2019, Austin, TX.

PURDUE UNIVERSITY®