Testing Environment for Accessing and Monitoring Networked Automation and Measurement Equipment

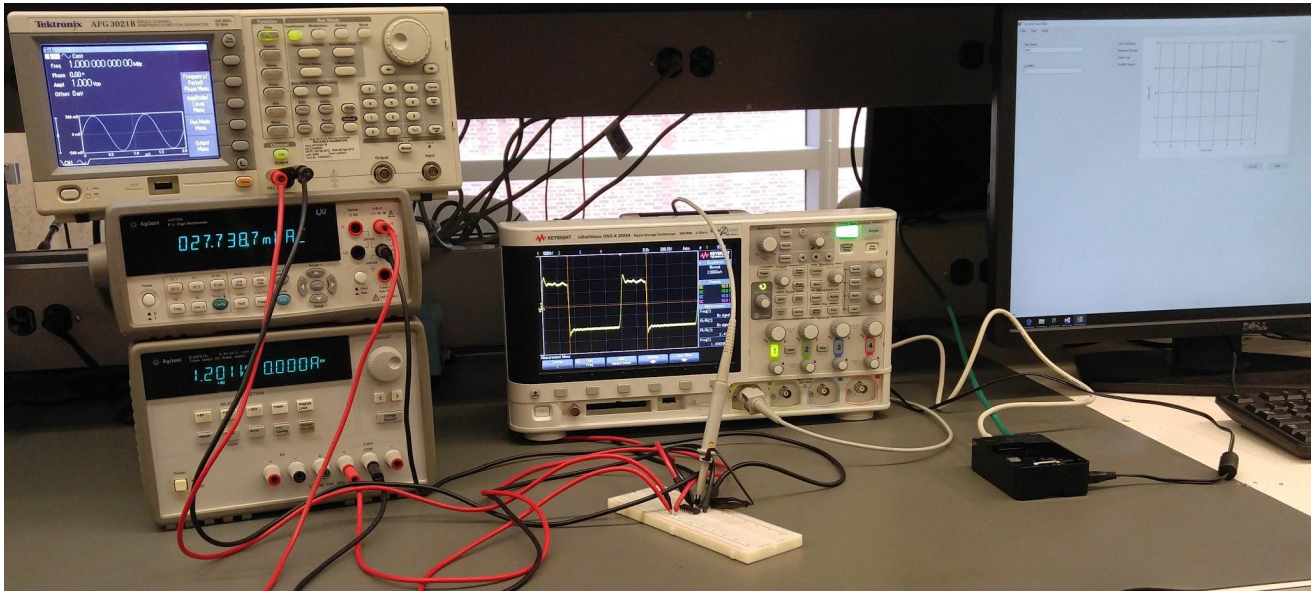
Antonio Montoya
Christian Hurst
Braden Rosengren
Ben Wiggins
Chris Little

Purpose/Problem Statement

In this project, we aim to:

- Develop remote test automation and monitoring capabilities
- Develop an API for user creation of custom tests
- Provide inexpensive alternative to more costly solutions
- Demonstrate conceptual viability of standardized system
- Present plan for creating an open source platform

Project Vision



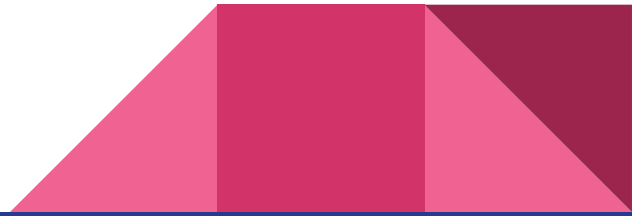
For illustration purposes only

Functional Requirements

- Web server and browser-based UI/UX
- Raspberry Pi must interface with lab equipment
- Raspberry Pi must initialize all required processes on startup
- PCB breakout board for computer to chip communication
- Create variable logic level shifter for SPI, I2C and GPIO busses

Non-functional Requirements

- Software is well organized and readable
- Create a guide for writing tests and supporting additional test equipment
- Connectors are spaced to allow for easy connections
- Demonstrate proof of concept and potential usage cases

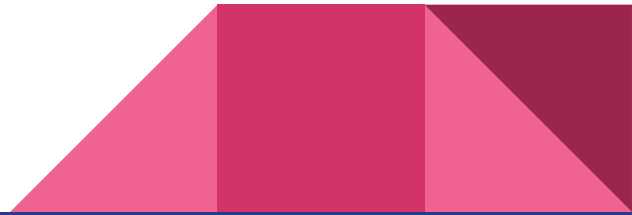


Constraints & Considerations

- We need to create a cost-effective platform
- The remote interface should be cross-platform compatible
- We need to support GPIB commands over IEEE-488, USB, and Ethernet
- Enable users to create and execute custom tests

Market Survey & Usage Target

- Saw a need for remote test hardware automation in both academia and industry
- Creating a testing platform usable independent of the user environment
- Current GPIB interfaces may be economically infeasible for those with smaller budgets



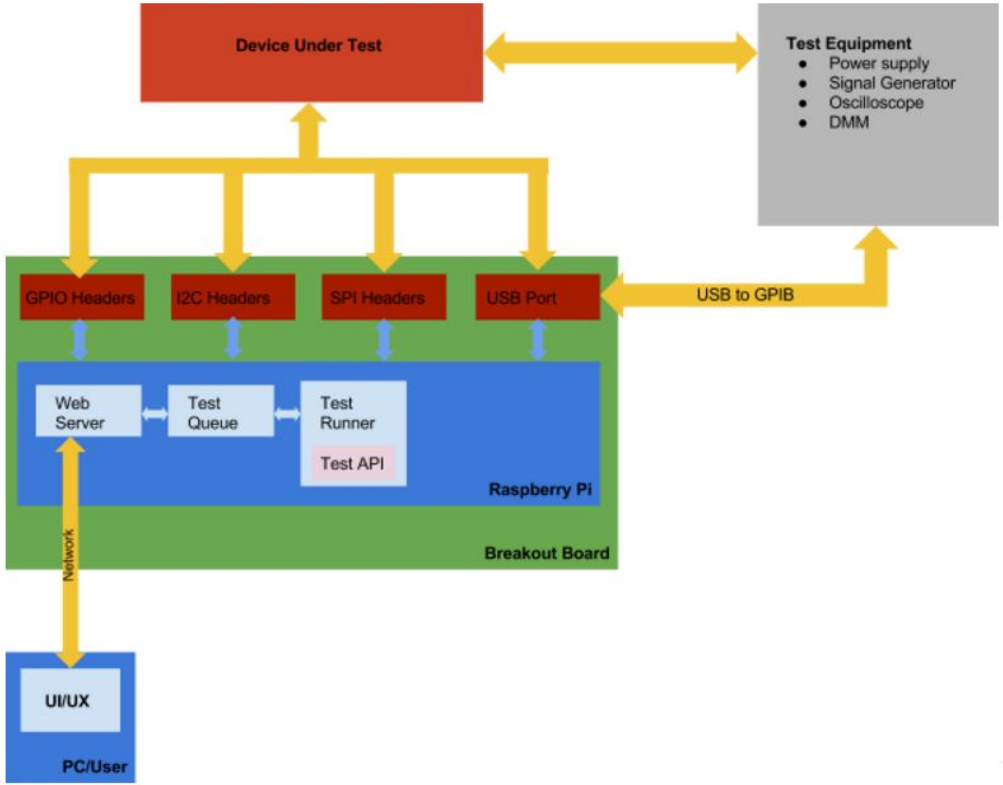
Project Development Timeline

Semester 1 software timeline	
Week of 10/3	Get PI up and running
Week of 10/10	Implement basic server functionality Compare and choose possible scripting languages
Week of 10/17	Create a basic test runner script that can save files and can do basic file IO
Week of 10/24	Server needs to be able to interact with the test runner
Week of 10/31	Finish Server test runner interaction code. Test IO
Week of 11/7	HTML API Start GPIO API Start GPIB support
Week of 11/14	Implement GPIO API Implement GPIB support
Week of 11/21	Thanksgiving break
Week of 11/28	Put in effort where it is needed
Week of 12/5	Dead Week and Finals Weeks No required work

Semester 1 Hardware Timeline	
Week of 10/3	Determine the schematic and layout software that contains the libraries required for this project.
Week of 10/10	Determine modes of communication between Raspberry pi and the test equipment. Also the best uses to breakout the rest of the pins.
Week of 10/17	Researching variable level shifters to 1.5, 1.3, 1.1, 1, and .8 volts. Pinouts to ADC with resolution 10 bits.
Week of 10/24	Revision 1 of PCB needs to be finalized and ordered.
Week of 10/31	Check over design while waiting for PCB.
Week of 11/7	Test PCB
Week of 11/14	Make needed changes to PCB. Reorder PCB
Week of 11/21	Thanksgiving break.
Week of 11/28	Test Revision 2 of PCB
Week of 12/5	Dead Week and Finals Week Documentation and Presentation.

Semester 2 timeline	
To be determined once we have a clearer picture of what needs to be accomplished	Testing Stretch goals

Functional Description



Technology Utilization (HW/SW used)

- Raspberry Pi
- GPIB-USB Controller
- Apache Web Server
- Python
- Eagle

Detailed Design (Modular design specifics)

PC UI/UX

- Will be created with Apache Web Server and Python CGI
- Browser-based design ensures cross-platform compatibility

Breakout Board

- Level shifters allow communication with extended range of DUTs

USB to GPIB Connector

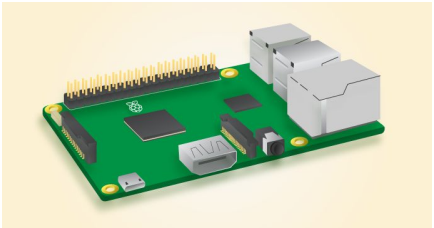
- Allows control of test equipment

Test API

- Allows users to easily define and implement their own tests
- Streamlines storage and retrieval of results

Resource & Cost Projection

- Raspberry Pi 3 Model B - $35.69 (Amazon.com)
- Prologix GPIB-USB Controller - 149.95 (Prologix.biz)
- Custom breakout board - TBD
- Open source software



http://prologix.biz/images/detailed/0/GPIB-USB-front.jpg
https://www.raspberrypi.org/wp-content/uploads/2016/02/Pi_3_Model_B.png

Planned Test Cases

- I2C and SPI compatible chips
- Use a previously verified DAC voltage drift test on a I2C DAC IC and monitor the results with the system
- Wafer Prober test setup

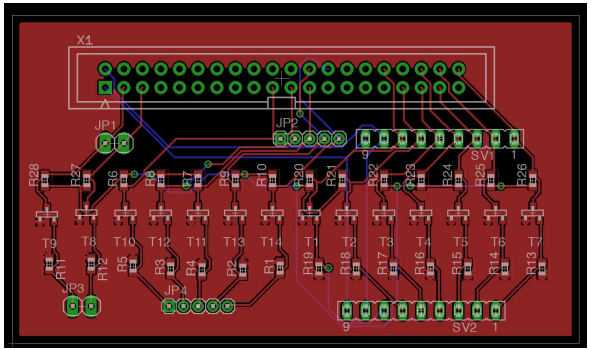
http://www.ti.com/graphics/folders/partimages/DAC6574.jpg

Module P.O.C. Tests

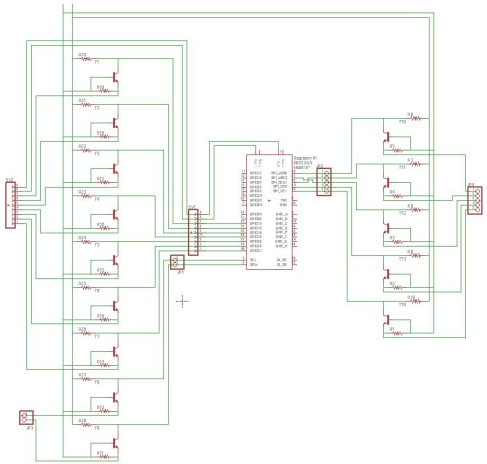
- Verified GPIB communication from Raspberry Pi to signal generator
- Verified dynamic content generation with Python CGI scripts
- Verified proposed level shifter schematic
- Verify PCB functionality
- Verify I2C communication

Milestone Completion

- Established control of test equipment over GPIB through the Raspberry Pi
- Began implementation of remote server and test runner software
- Designed a variable logic level shifter circuit
- Created the PCB



PCB Layout Rev. 2



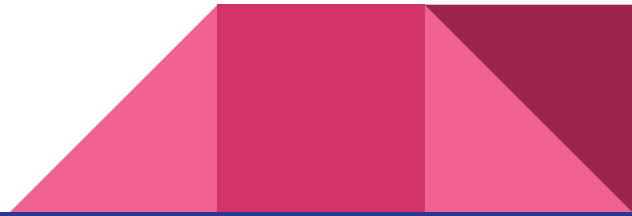
Schematic Rev. 2

Team Member Contributions

- Antonio Montoya - System level design lead, hardware design and documentation, and system test designer
- Christian Hurst - Platform software development, user interface design, and group website design
- Ben Wiggins - Assist with software development and document team and advisor meetings
- Braden Rosengren - Hardware design and implementation
- Chris little - Hardware concept creation and design, modular hardware functional testing and simulation, DUT and POC research

Coming Soon…

- An API to allow users to easily define their own tests
- A breakout board with level-shifted busses
- An intuitive browser-based interface



Questions?