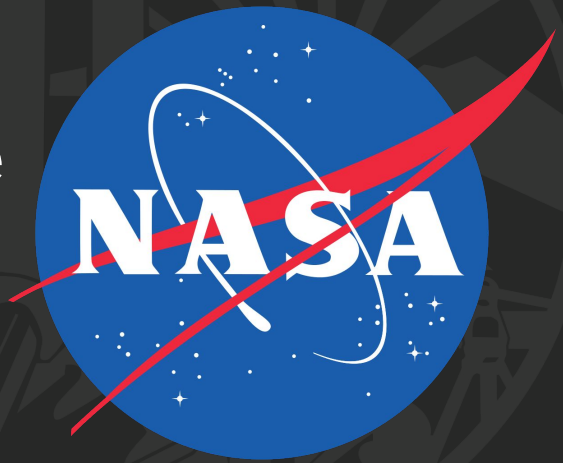
Flywheel Energy Storage Controls System



Andrew Jones (Computer Engineering)
Brian Cartwright (Computer Science)
Ian Tanimoto (Computer Science)

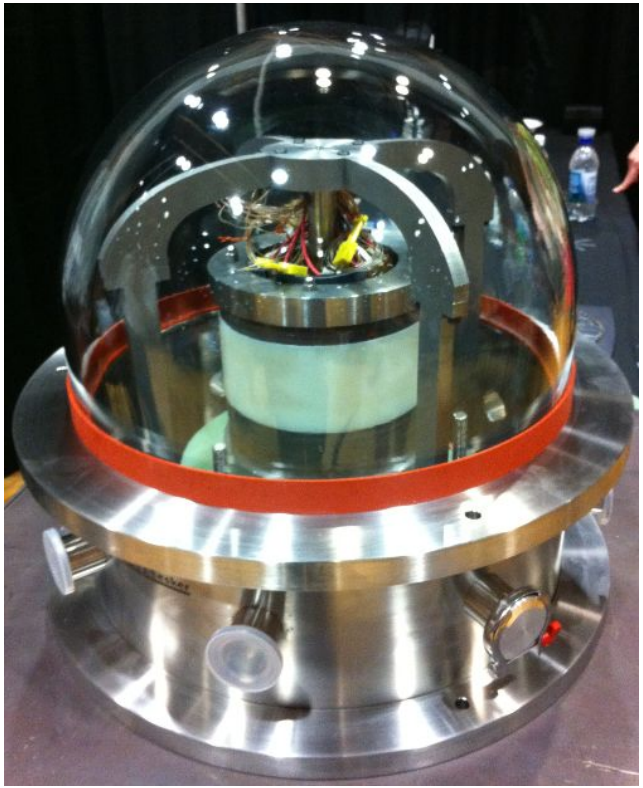
University of Idaho
College of Engineering

Project Goal

- Finish the multi-year Low Speed Flywheel project
  - Develop accurate and precise control structures to safely accelerate and stabilize the flywheel rotor
  - Identify strengths and weaknesses of existing code
  - Design and develop improved control code
  - Close the book on the Low Speed FESS so future teams can focus on the High Speed design
- Advise on changes needed for the High-Speed Flywheel controls

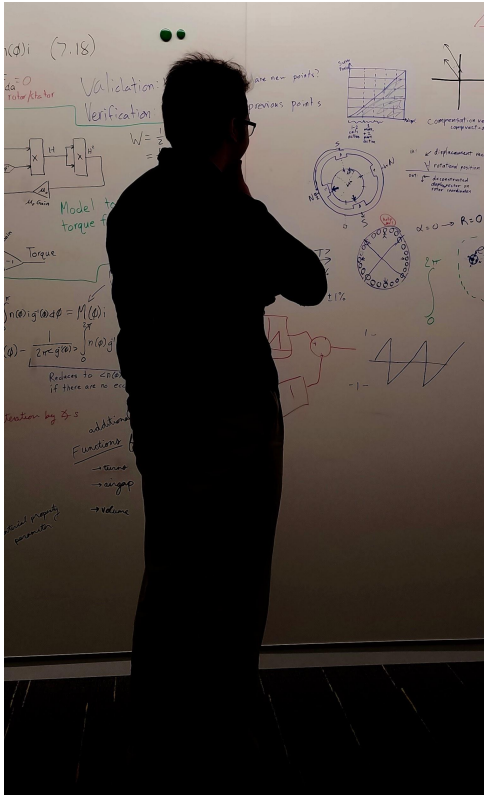
What is a Flywheel?

- A flywheel is a mechanical device that stores rotational energy. The faster it spins, the more energy it stores.
- The UIFESS uses an inside-out arrangement, with the rotor on the outside of the stator. This allows the rotor to have a larger radius, increasing the system's energy capacity.



Resources Used

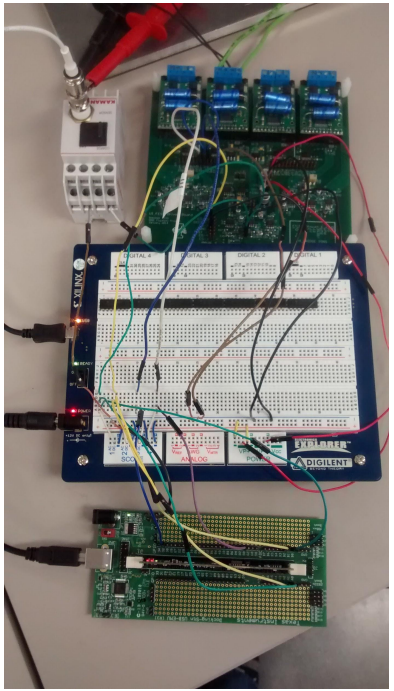
- Software
  - Texas Instruments Code Composer Studio
    - ControlSUITE Library
  - Digilent Waveforms
- Hardware
  - TI Delfino 335 Microcontroller
  - KD-2306 Displacement Sensor
  - Kevin Ramus' Power Electronics PCB
  - SASB Demonstrator Unit
  - Digilent Explorer Board
  - Power Supply Unit x3

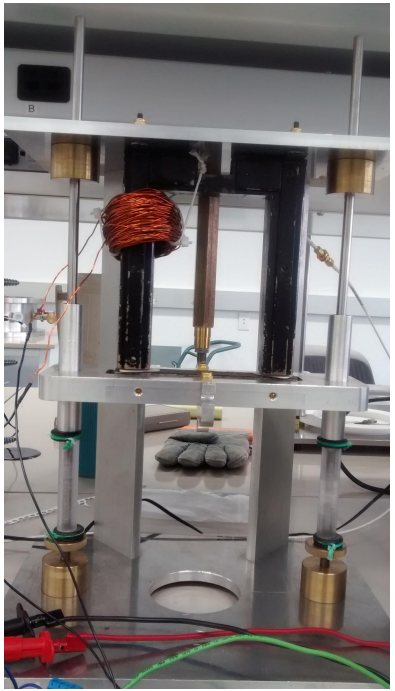


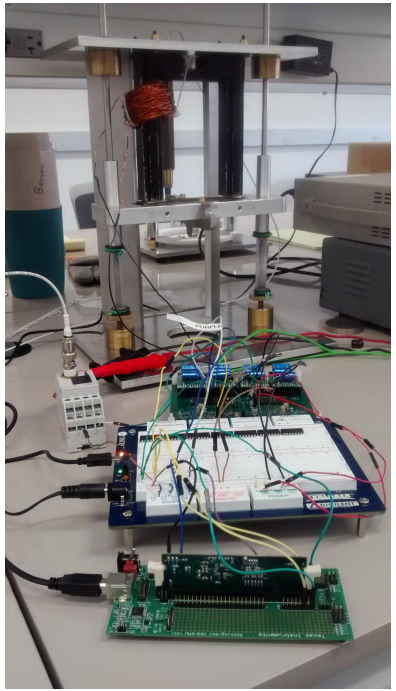
Completed Work

- Achieved levitation in a single axis
  - Built the test bench according to diagrams received from Kevin Ramus
  - Solved issues with the wiring setup and power requirements
  - Fully debugged the single-axis stabilization code
  - Ran the code on the SASB test setup
  - Completed testing of stabilization code in one axis
- Designed algorithms for acceleration and deceleration
- Reviewed dual-axis stabilization code
  - Expands the functionality of the working single-axis code
  - Should be trivial to get working once the prototype is assembled and balanced

SASB Hardware Setup







Discoveries since last review

- Finished testing the SASB code
  - There was a mysterious 5-second-periodic offset when we first ran the code. Investigation revealed that it was caused by a debugging function left in by the previous team. The code works as expected with that re-commented-out.
  - The strength of the SB & SASB's pulling force is determined by the duty cycle of the direction pin, not the direction of the direction pin. 50/50 timeshare is weakest, 100/0 in either direction pulls most strongly. This is usually not the case with the FRRM.
  - There were inaccuracies in the wiring diagram provided to us
    - The Pololu's ground and 24V bus positions were reversed

Work for Future Teams

- Assemble and wire the flywheel
- Perform 2-D stabilization testing on flywheel
- Understand and finish the FRRM code, incorporate acceleration
- Test the acceleration code for real
- Optimize algorithm and resource usage
  - Use dual-core capability of Delfino 77d to replace one 335
  - Investigate the SASB “sticking” problem
- Begin adaptation for High Speed FESS

Cost of Potential Replacements

- Resolute Absolute Ring Encoder - $2600
- Delfino 335 MCU - $28.31 per unit
  - ControlCARD - $69 per unit
  - Dock - $99 per unit
- Delfino 77d MCU - $41.49
  - ControlCARD - $159
  - Dock - $219
- KD-2306 Displacement Sensors: $1800 per unit
- Power Electronics PCBs - ~$66 per unit
- Pololu Power Converters - $64.95 per unit (or $584.50 for 10)