

# A New Paradigm for Online Education and Research in "Power" Worldwide

*Using ONR-funded Low-cost Rapid Real-time Platform*



UNIVERSITY OF MINNESOTA



CUSP™

# Consortium of Universities for Sustainable Power (CUSP)

**CUSP™**

What is CUSP™?  
CUSP™ Curriculum  
CUSP™ Members  
Join  
Forum

**Welcome**

CUSP Welcome Video

**DEVELOPED BY EXPERTS**

**NSF** NATIONAL ACADEMY OF ENGINEERING

**Upcoming Workshop (NSF-sponsored Approved by NAE)**  
**"Reinventing Electric Power Curriculum with Sustainability Focus"**  
June 15-17, 2017 [University of Minnesota](#), Minneapolis, Minnesota

[Agenda](#)  
[Highlights](#)

**Location**  
U of M campus in Minneapolis

**Courses**

**Power Systems**

- Electric Power Systems
- Electricity Markets
- Power Generation, Operation & Control
- Power System Protection
- Advanced Power Systems 1 & 2
- High Voltage Technology

**Power Electronics**

- Power Electronics
- Advanced Power Electronics I
- Advanced Power Electronics II

**Electric Machines & Drives**

- Electric Machines and Drives
- Electric Machine Design
- Vector Control of Drives
- FEA for Machine Design

**Renewable Energy**

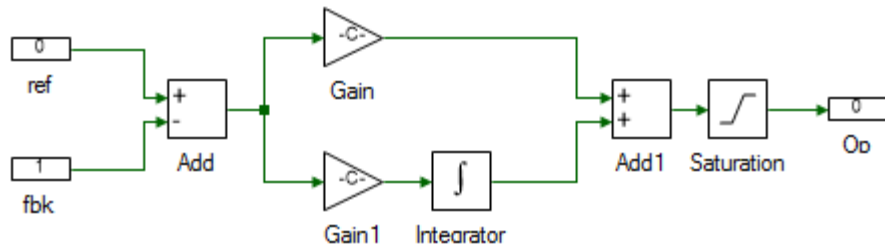
- Wind Energy Essentials

235 U.S. Universities as members (over 450 faculty)

# Increasing course adoption

- Cost of laboratory infrastructure major impediment
- Software such as MATLAB are essential but inaccessible
  1. Costs around \$4,500 for educational purpose
  2. Upwards of \$21,000 for commercial use
  3. Unaffordable to community colleges, universities in developing countries and startups
- Real-time prototyping hardware such as dSpace are equally expensive and not best suited for power electronics

# Sciamble Workbench



- Advanced coding environment.
- Supports matrix operations.

- Numerical simulation software.
- Model based drag and drop.

```
Public kpi As Native Double ! current loop proportional gain  
Public kii As Native Double ! current loop integral gain
```

```
! Parameter initialization function
```

```
Public Function Init()
```

```
! local variables, just for the purpose of computation and
```

```
Local  $\lambda_{rdq\theta}$  As Native Double =  $\sqrt{(\lambda_{rd\theta^2} + \lambda_{rq\theta^2})}$ 
```

```
Local  $\lambda_{sdq\theta}$  As Native Double =  $\sqrt{(\lambda_{sd\theta^2} + \lambda_{sq\theta^2})}$ 
```

```
Local  $\theta_{Isdq}$  As Native Double = Math:ATan2(Isq $\theta$ , Isd $\theta$ )
```

```
Local  $\theta_{Vsdq}$  As Native Double = Math:ATan2(Vsq $\theta$ , Vsd $\theta$ )
```

```
Local Isdq $\theta$  As Native Double =  $\sqrt{(Isq\theta^2 + Isd\theta^2)}$ 
```

```
Local Vsdq $\theta$  As Native Double =  $\sqrt{(Vsq\theta^2 + Vsd\theta^2)}$ 
```

# Workbench – Model based numerical simulation and real-time code generation

The screenshot displays the Workbench software interface for model-based numerical simulation and real-time code generation. The main workspace shows a complex block diagram of a motor control system, including an IMMotor and DCMotor block, various gain blocks, adders, multipliers, and PWM generators. The diagram is interconnected with various control blocks and sensors, such as Scope1, Scope2, and Scope3.

The left sidebar shows the Property panel with the following configuration options:

- Device Configuration:**
  - Device: 3 Inverter
  - Run mode: Forever
- Clock Configuration:**
  - Frequency (MHz): 150
- PWM Configuration:**
  - Enabled: Enabled
  - Frequency (Hz): 15000
  - Mode: Independent
  - Waveform: Sawtooth
- ADC Configuration:**
  - Enabled: Enabled
  - Trigger: Software
  - SH Window ( $\mu$ S): 1
- AQB Configuration:**
  - Enabled: Enabled
  - Encoder Lines: 1000
  - Slowest Refres...: 0.01
- Data Format:**
  - Double: 32 bits
  - Single: 32 bits
  - Long: 32 bits
  - Integer: 16 bits
  - Short: 16 bits
  - Boolean: 16 bits
- Data Logger:**
  - Transfer rate ...: 2880

The right sidebar shows the Solution Explorer with the following project structure:

- IMVectorControl3Inverter
  - IMParam
  - DCParam
  - ModelFile
    - IMMotor
    - VIToFlux
    - abcTodq
    - dqToabc
  - PIVd
  - PIVq
  - PIIq
  - dqToabc
  - EstimatorModel
  - abcTodq
  - DCMotor
  - PIVdc

The bottom panel shows the Compiler Messages window with the following information:

0 Errors 5 Warnings 0 Messages 0 Exceptions

#	Code	Description	File	Project
1	WM0006	One or more tool outputs are unconnected.	IMMotor	IMVectorControl3Inverter
2	WM0006	One or more tool outputs are unconnected.	ModelFile	IMVectorControl3Inverter
3	WM0006	One or more tool outputs are unconnected.	ModelFile	IMVectorControl3Inverter
4	WM0005	All of the tool outputs are unconnected. Hence might be ignored during code generation.	ModelFile	IMVectorControl3Inverter
5	WM0005	All of the tool outputs are unconnected. Hence might be ignored during code generation.	ModelFile	IMVectorControl3Inverter

# Workbench Features – Script Editor

```
IMParam
Public Module IMParam ! Induction motor parameter
  Public sqrt2by3 As Native Double =  $\sqrt{2} / \sqrt{3}$ 

  Public Rs As Native Double = 1.79 ! Stator Resistance
  Public Rr As Native Double = 1.05 ! Rotor Resistance
  Public Lls As Native Double = 6.2E-3 ! Stator leakage inductance
  Public Llr As Native Double = 5.8E-3 ! Rotor leakage inductance
  Public Lm As Native Double = 30E-3 ! Mutual Inductance
  Public Jeq As Native Double = 0.00015 ! Rotor Inertia
  Public p As Native Double = 4 ! Number of poles

  Public Ls As Native Double = Lm + Lls ! Stator Inductance
  Public Lr As Native Double = Lm + Llr ! Rotor Inductance

  Public f As Native Double = 50 ! Rated frequency
  Public VLLrms As Native Double = 14.7 ! Rated line to line voltage
  Public s As Native Double = 0.01 ! Rated slip

  Public  $\omega_{syn}$  As Native Double = 2 *  $\pi$  * f ! Synchronous speed at rated frequency
  Public  $\omega_m$  As Native Double = (1 - s) *  $\omega_{syn}$  ! Speed at rated slip
  Public Va As Native Double = VLLrms * sqrt2by3 ! Phase voltage peak
  Public Vs0 As Native Double = 1.5 * Va

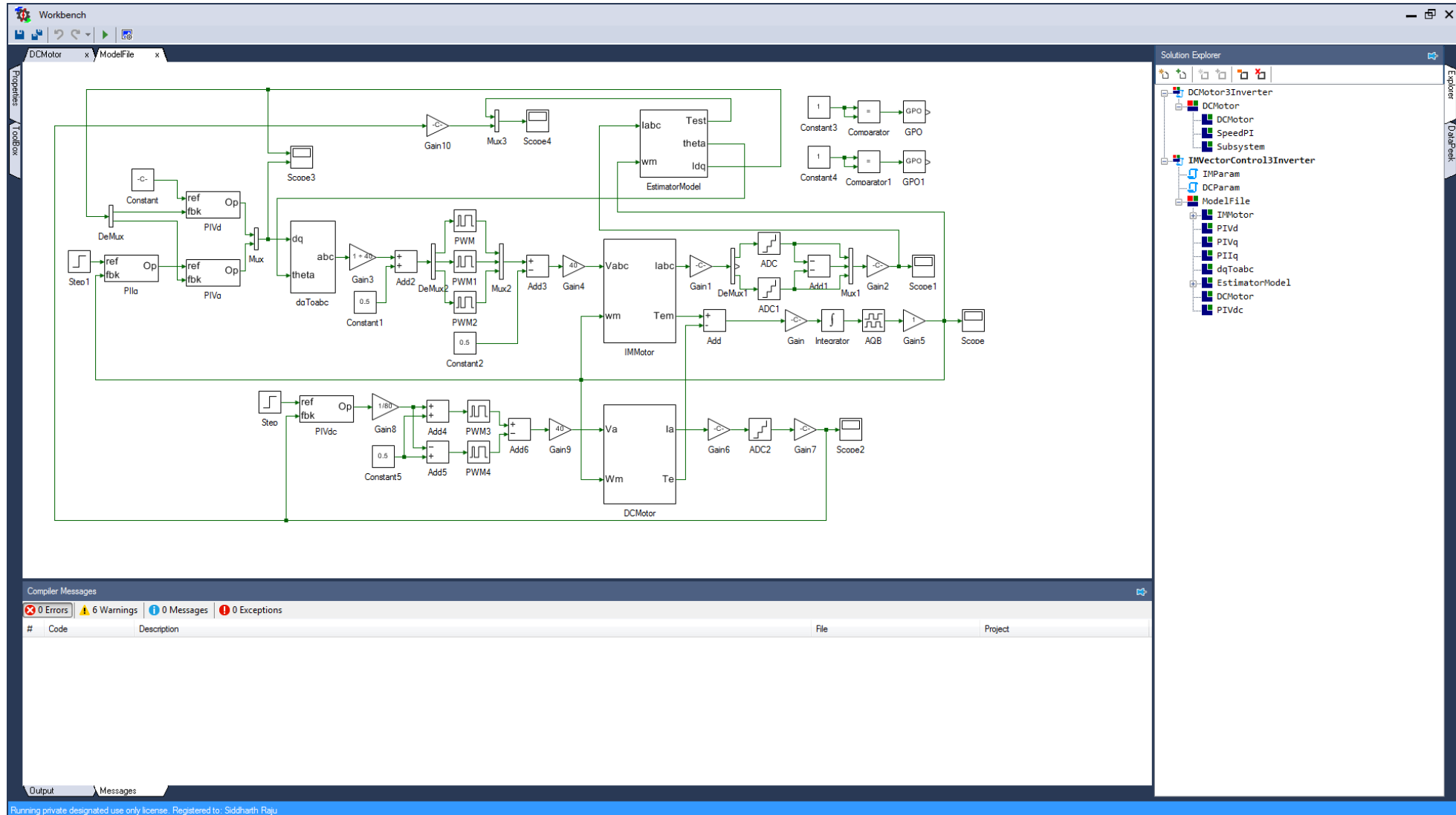
  Public  $\theta_{Vs0}$  As Native Double = 0
  Public  $\theta_{da0}$  As Native Double = 0

  ! Initial stator dq Voltages
  Public Vsd0 As Native Double = sqrt2by3 * Math:Abs(Vs0) * Math:Cos( $\theta_{Vs0}$  -  $\theta_{da0}$ )
  Public Vsq0 As Native Double = sqrt2by3 * Math:Abs(Vs0) * Math:Sin( $\theta_{Vs0}$  -  $\theta_{da0}$ )

  Public  $\tau_r$  As Native Double = Lr / Rr
  Public A As Double = [[Rs, - $\omega_{syn}$  * Ls, 0, - $\omega_{syn}$  * Lm], [ $\omega_{syn}$  * Ls, Rr,  $\omega_{syn}$  * Lm, 0], [0, -s *  $\omega_{syn}$  * Lm, Rr, -s *  $\omega_{syn}$  * Lm], [0, 0, 0, 0]]
  Public Ainv As Double = 1 / A
  ! Initial dq stator and rotor voltages (as row matrix)
  Public Vdq0 As Double = [[Vsd0], [Vsq0], [0], [0]]
  Public Idq0 As Double = Ainv * Vdq0 ! Initial dq stator and rotor currents
  ! Retriving individual current from row matrix
  Public Ids0 As Native Double = Idq0(1) / p
```

- Language designed from scratch to support dynamic compilation.
- Extremely easy to use.
- Inbuilt matrix operation support.

# Workbench Features – Real-time Code Gen



Running private designated use only license. Registered to: Siddhant Raju

# Workbench Features – Real-time Code Gen

The screenshot displays the Simulink Workbench environment for a motor control system. The main workspace contains a complex block diagram of a motor control system. Key components include:

- Control Loop:** A feedback loop starting with a step input (Step1) and a reference (ref) signal. It passes through a summing junction (Op) and a gain block (Gain3) to produce a reference current (I<sub>ref</sub>). This is compared with a feedback signal (fbk) to generate an error signal, which is then processed by a PI controller (PIVd).
- Current Control:** The reference current (I<sub>ref</sub>) is compared with the actual current (I<sub>abc</sub>) to generate a current error. This error is processed by a PI controller (PIVa) and a summing junction (Add2) to produce a reference current (I<sub>dq</sub>). This is compared with the actual current (I<sub>dq</sub>) to generate another error signal, which is processed by a PI controller (PIVb) and a summing junction (Add3) to produce a reference current (I<sub>abc</sub>).
- Motor Model:** The reference current (I<sub>abc</sub>) is compared with the actual current (I<sub>abc</sub>) to generate a current error. This error is processed by a PI controller (PIVc) and a summing junction (Add4) to produce a reference current (I<sub>dq</sub>). This is compared with the actual current (I<sub>dq</sub>) to generate another error signal, which is processed by a PI controller (PIVd) and a summing junction (Add5) to produce a reference current (I<sub>abc</sub>).
- Power Electronics:** The reference current (I<sub>abc</sub>) is compared with the actual current (I<sub>abc</sub>) to generate a current error. This error is processed by a PI controller (PIVd) and a summing junction (Add6) to produce a reference current (I<sub>abc</sub>). This is compared with the actual current (I<sub>abc</sub>) to generate another error signal, which is processed by a PI controller (PIVc) and a summing junction (Add7) to produce a reference current (I<sub>abc</sub>).
- Estimator Model:** An EstimatorModel block receives the reference current (I<sub>abc</sub>) and the actual current (I<sub>abc</sub>) to estimate the motor's state (theta, Idq).
- ADC and DAC:** The reference current (I<sub>abc</sub>) is converted to a digital signal (ADC) and the actual current (I<sub>abc</sub>) is converted to an analog signal (DAC).
- Scope:** Several Scope blocks (Scope1, Scope2, Scope3, Scope4) are used to monitor the system's performance.



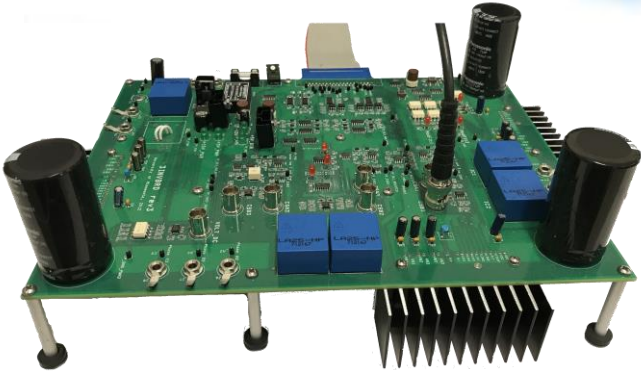
The Solution Explorer on the right shows the project structure, including the following components:

- DCMotor3Inverter
  - DCMotor
  - SpeedPI
  - Subsystem
- IMVectorControl3Inverter
  - IMParam
  - DCParam
  - ModelFile
    - IMMotor
    - PIVd
    - PIVq
    - PIIq
    - dqToabc
    - EstimatorModel
    - DCMotor
    - PIVdc

The Compiler Messages window at the bottom shows 0 Errors, 6 Warnings, 0 Messages, and 0 Exceptions. The Output and Messages windows are also visible at the bottom.



# Previously used system in electric drives lab (Spring 2017)

Academic	Non-academic	
Cost: > \$7,000	Cost: > \$20,000	
Cost: ≈ \$6,000	Cost: ≈ \$20,000	
Cost: ≈ \$7,500	Cost: ≈ \$7,500	
Total: > \$20,500	Total: > \$47,500	

# Currently used system in electric drives lab (Spring 2018)

Academic/Non-academic

---

Cost: FREE!



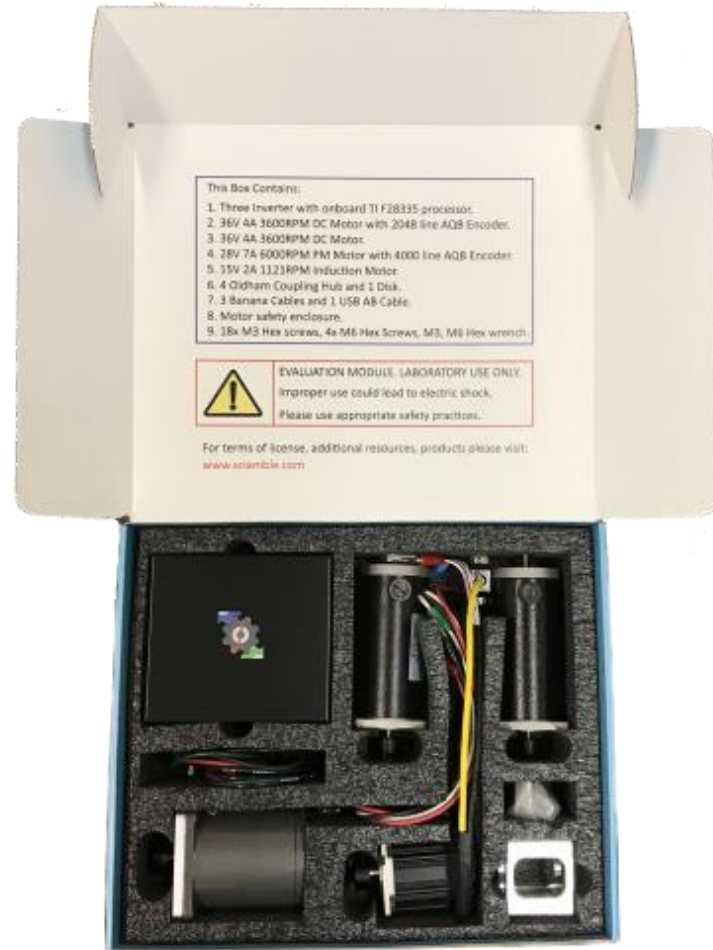
Cost: \$3,300



Total: \$3,300

---

# Electric Drives Lab



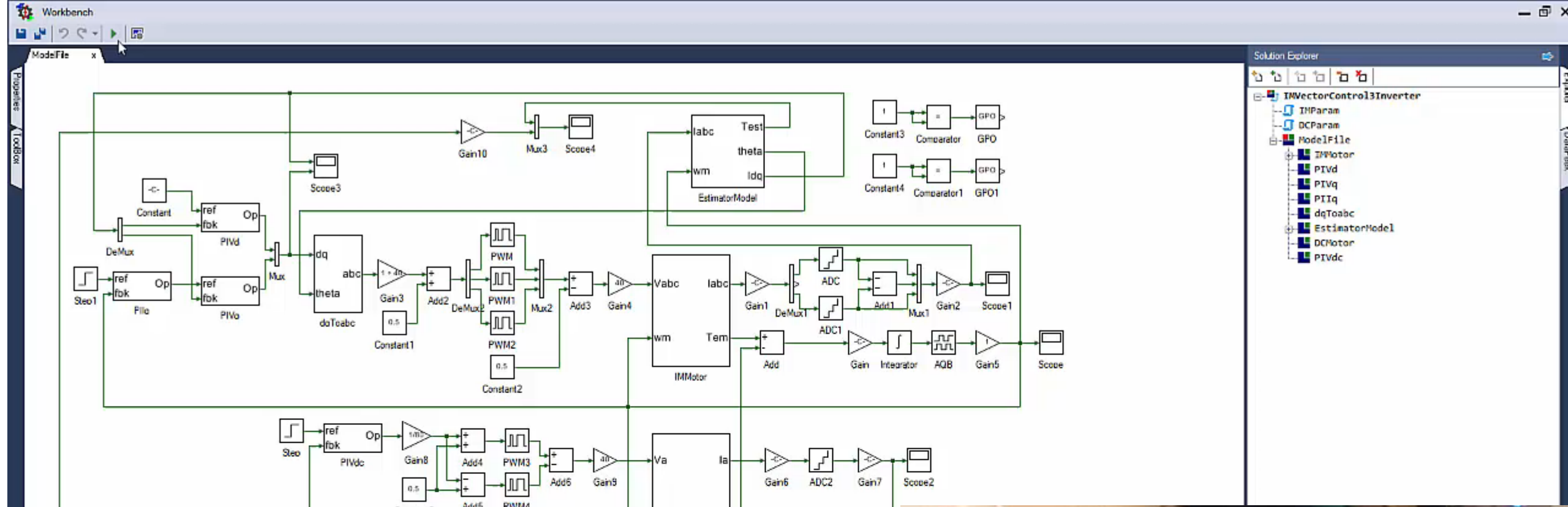
# List of Experiments

## Basic drives lab (undergraduate level)

1. Switched-mode DC-DC converter
2. Characterization of DC motor
3. DC motor closed-loop speed control
4. Four-quadrant operation of DC motor
5. Torque-load angle characteristics and speed control of PMAC motor
6. Determination of Induction motor parameters
7. Torque-speed characteristics and speed control of Induction motor

## Advanced drives lab (graduate level)

1. Characterization of Induction motor
2. Induction motor V/f control
3. Vector control of induction motor
4. Encoder-less vector control of induction motor
5. Direct torque control of induction motor
6. Space vector Pulse width modulation of two level three-phase inverter
7. Vector control of surface PMAC motor



Compiler Messages

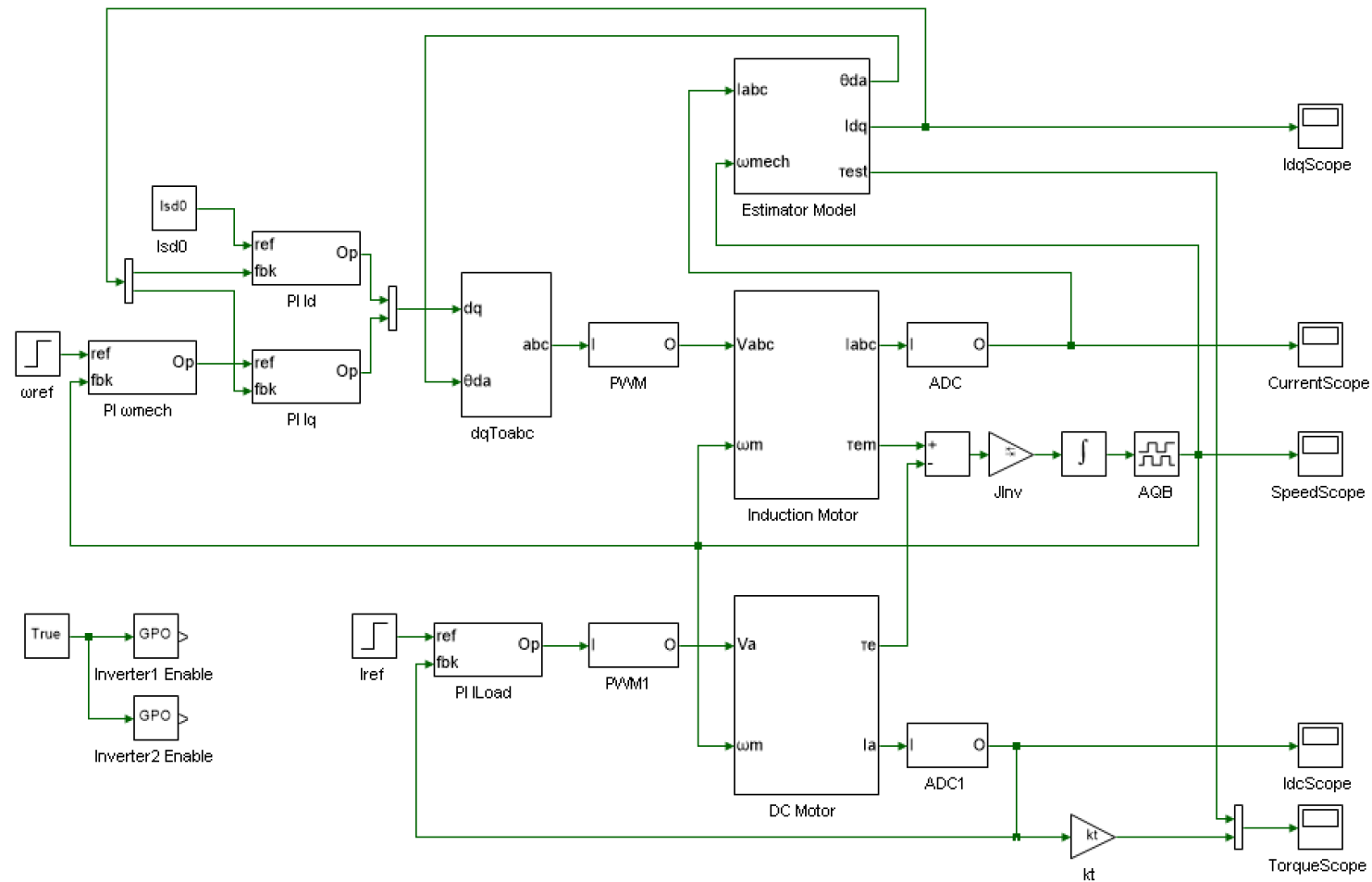
0 Errors | 4 Warnings | 0 Messages | 0 Exceptions

#	Code	Description	File

Output Messages



# Workbench Induction motor vector control



# Workbench Induction motor vector control

```
IMPParam x
Public Module IMPParam ! Induction motor parameter
  Public sqrt2by3 As Native Double =  $\sqrt{2} / \sqrt{3}$ 

  Public Rs As Native Double = 1.79 ! Stator Resistance
  Public Rr As Native Double = 1.05 ! Rotor Resistance
  Public Lls As Native Double = 6.2E-3 ! Stator leakage inductance
  Public Llr As Native Double = 5.8E-3 ! Rotor leakage inductance
  Public Lm As Native Double = 30E-3 ! Mutual Inductance
  Public Jeq As Native Double = 0.00015 ! Rotor Inertia
  Public p As Native Double = 4 ! Number of poles

  Public Ls As Native Double = Lm + Lls ! Stator Inductance
  Public Lr As Native Double = Lm + Llr ! Rotor Inductance

  Public f As Native Double = 50 ! Rated frequency
  Public VLLrms As Native Double = 14.7 ! Rated line to line voltage
  Public s As Native Double = 0.01 ! Rated slip

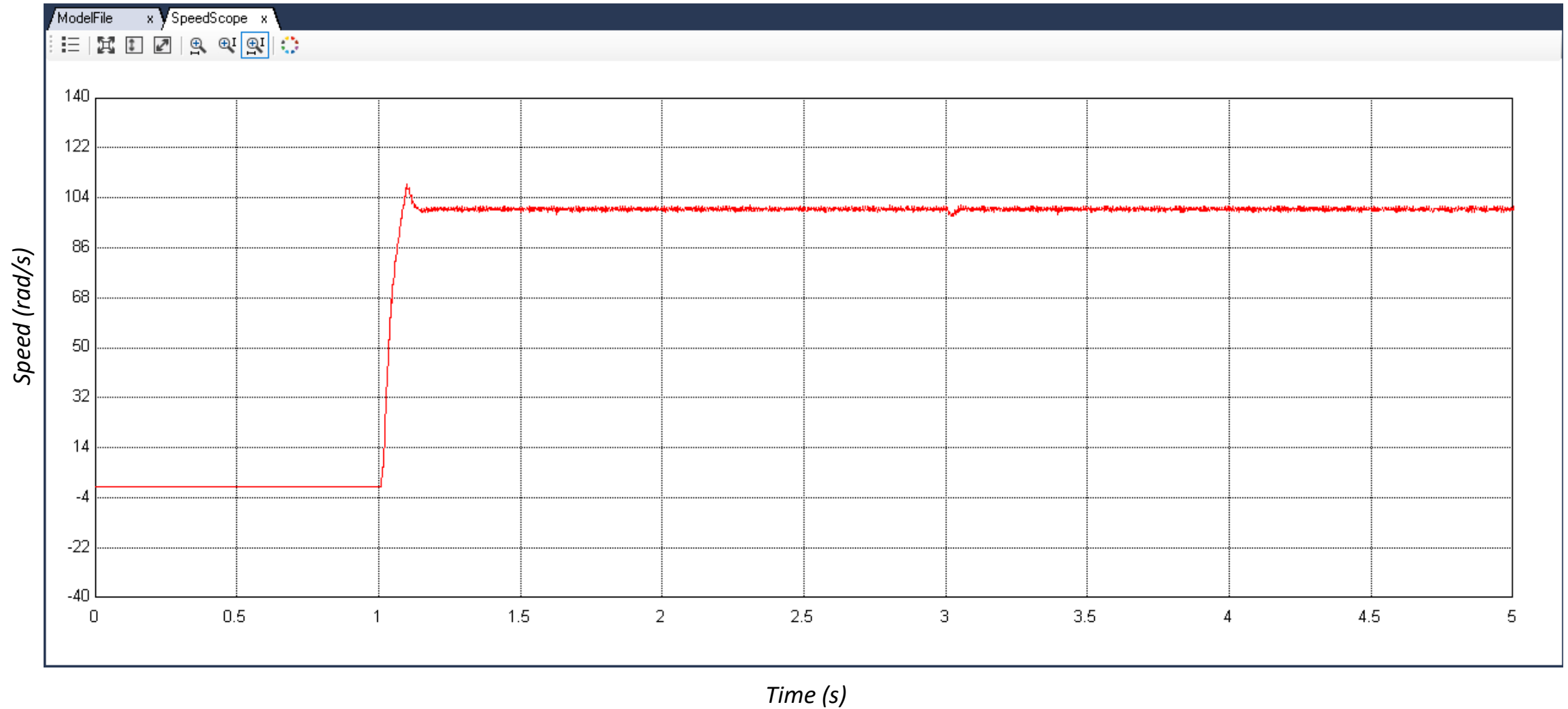
  Public  $\omega_{syn}$  As Native Double = 2 *  $\pi$  * f ! Synchronous speed at rated frequency
  Public  $\omega_m$  As Native Double = (1 - s) *  $\omega_{syn}$  ! Speed at rated slip
  Public Va As Native Double = VLLrms * sqrt2by3 ! Phase voltage peak
  Public Vs0 As Native Double = 1.5 * Va

  Public  $\theta_{Vs0}$  As Native Double = 0
  Public  $\theta_{da0}$  As Native Double = 0

  ! Initial stator dq Voltages
  Public Vsd0 As Native Double = sqrt2by3 * Math:Abs(Vs0) * Math:Cos( $\theta_{Vs0}$  -  $\theta_{da0}$ )
  Public Vsq0 As Native Double = sqrt2by3 * Math:Abs(Vs0) * Math:Sin( $\theta_{Vs0}$  -  $\theta_{da0}$ )

  Public  $\tau_r$  As Native Double = Lr / Rr
  Public A As Double = [[Rs, - $\omega_{syn}$  * Ls, 0, - $\omega_{syn}$  * Lm], [ $\omega_{syn}$  * Ls, Rs,  $\omega_{syn}$  * Lm, 0], [0, -s *  $\omega_{syn}$  * Lm, Rr, -s *  $\omega_{syn}$ ], [0, 0, s *  $\omega_{syn}$  * Lm, Rr]]
  Public Ainv As Double = 1 / A
  ! Initial dq stator and rotor voltages (as row matrix)
  Public Vdq0 As Double = [[Vsd0], [Vsq0], [0], [0]]
  Public Idq0 As Double = Ainv * Vdq0 ! Initial dq stator and rotor currents
  ! Retriving individual current from row matrix
  Public Isd0 As Native Double = Idq0(1,1)
```

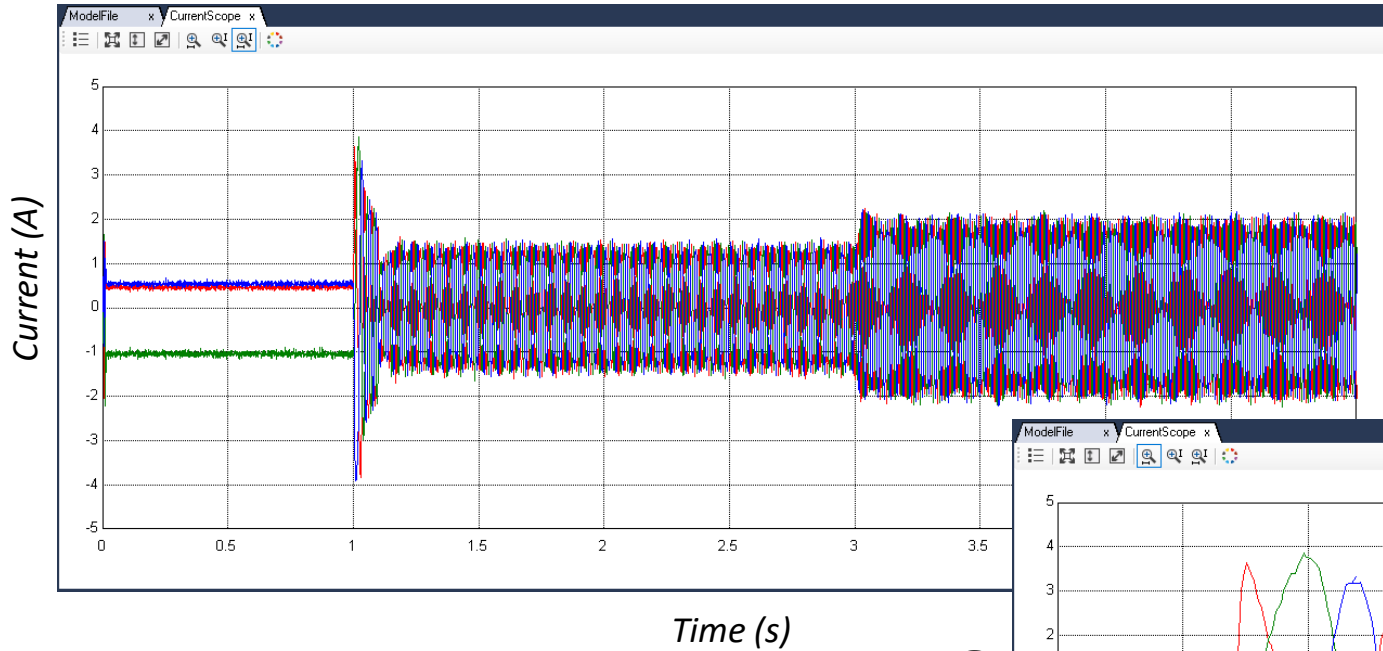
# Induction motor real-time vector control results



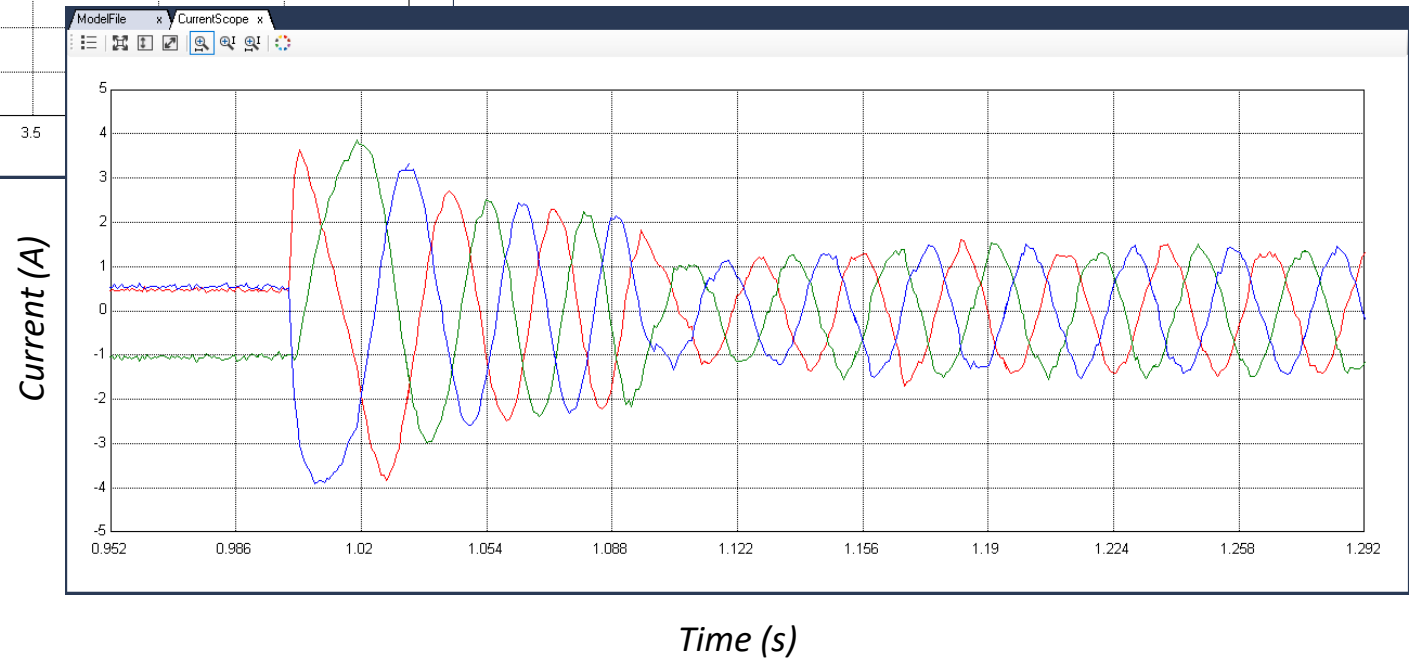


# Induction motor real-time vector control results

Induction motor 3 $\phi$  stator current

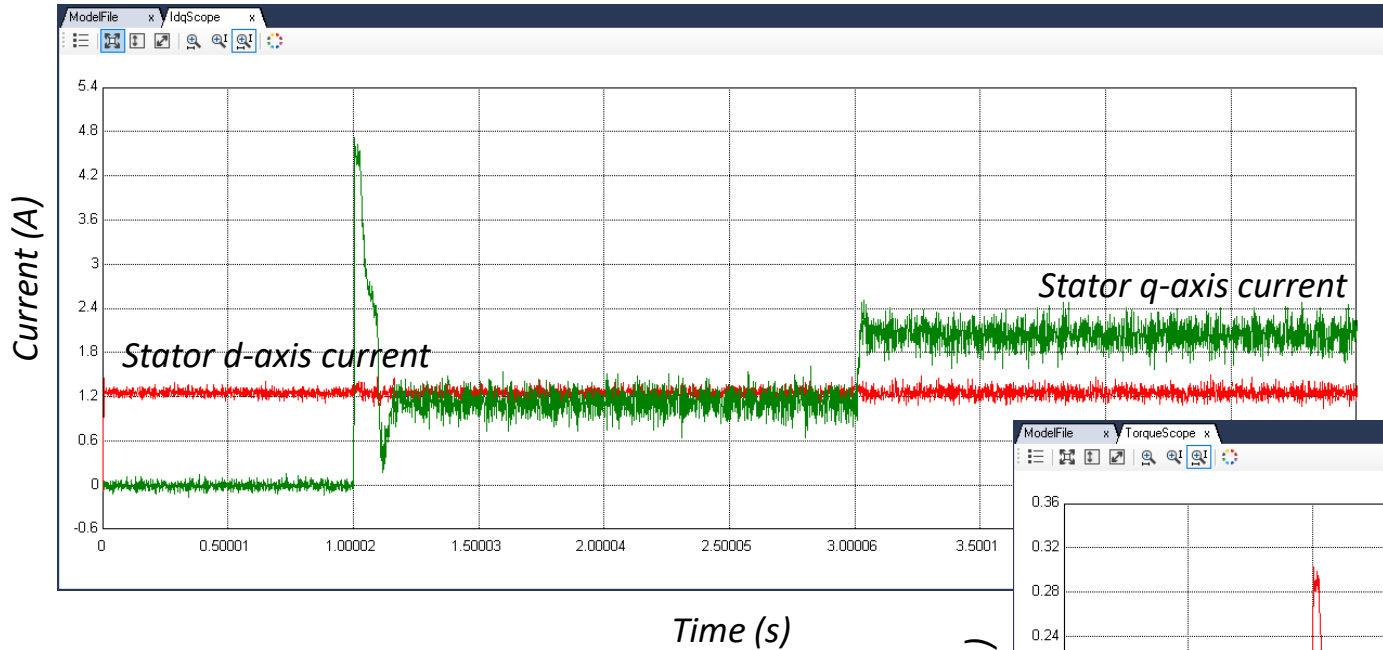


Induction motor 3 $\phi$  stator current (zoomed in)

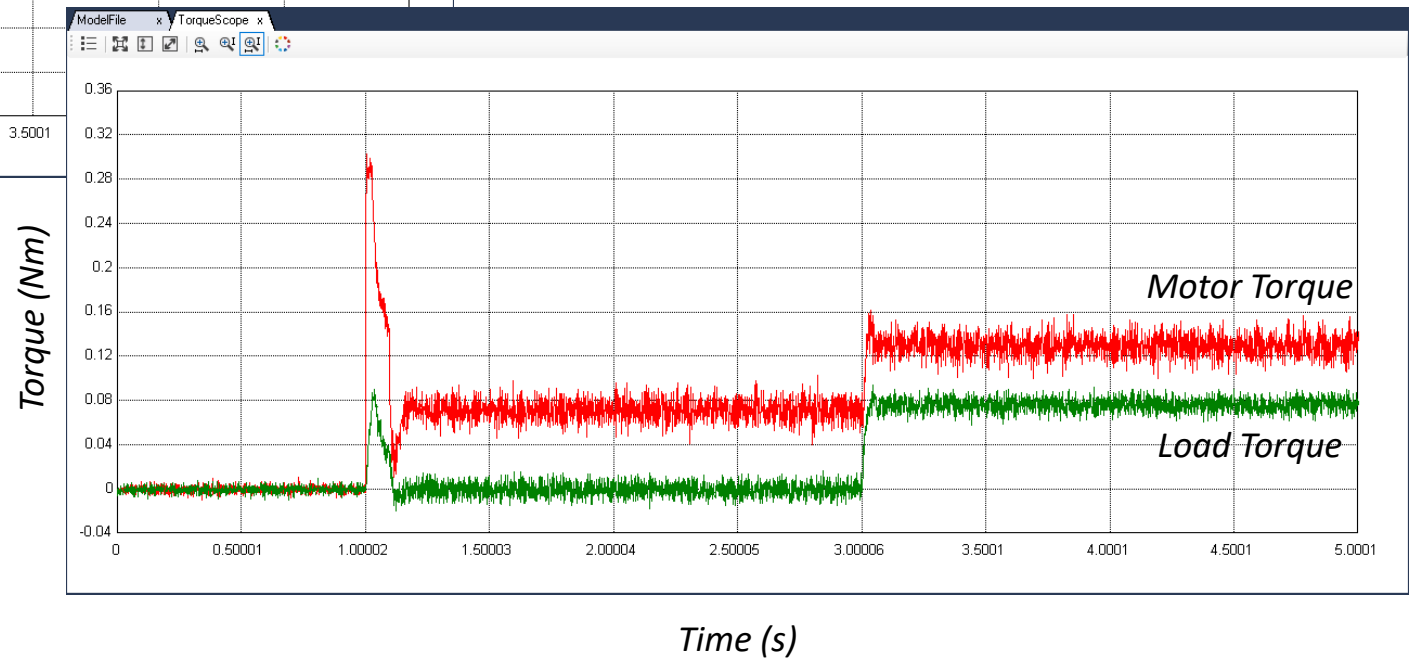


# Induction motor real-time vector control results

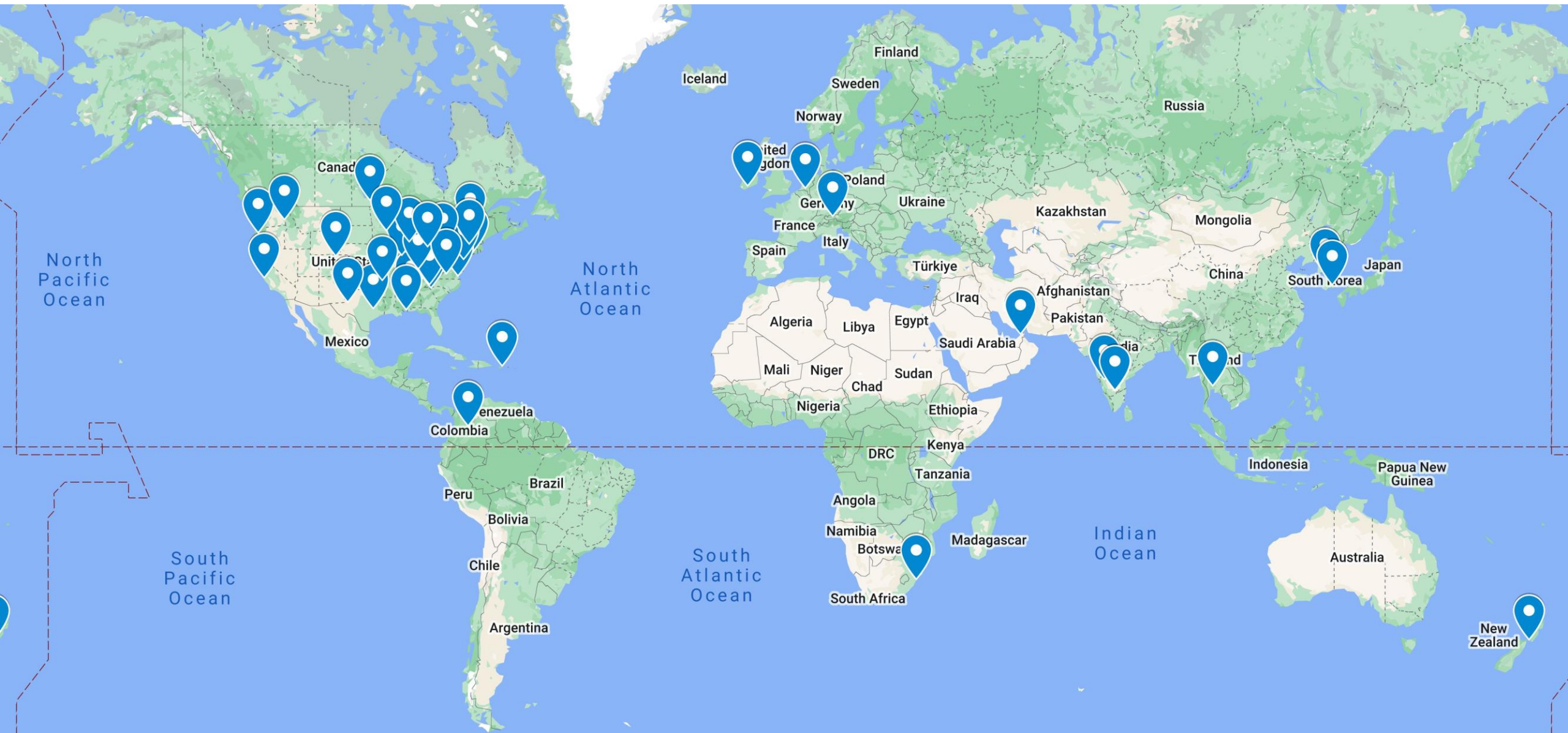
## Induction motor stator dq currents



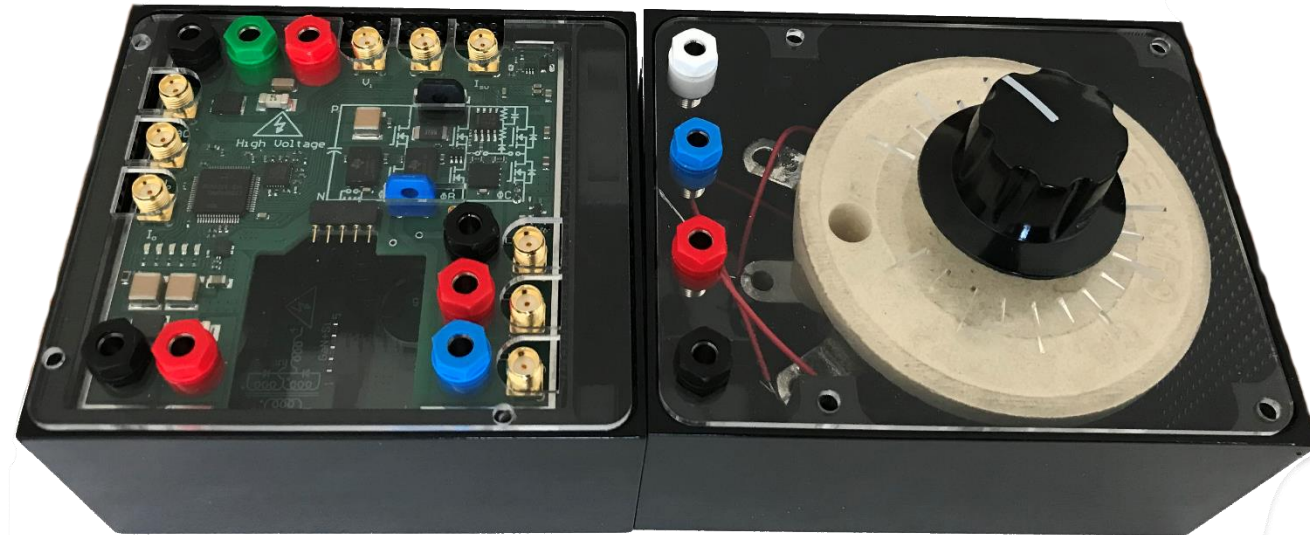
## Induction motor torque and load torque



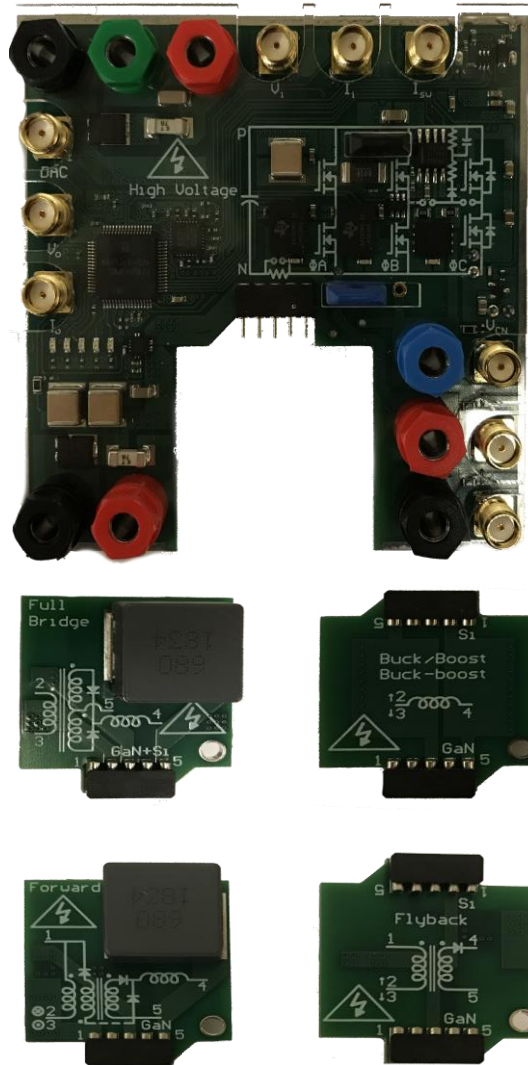
# Electric drives lab adoption - Worldwide



# WBG based Digitally controlled Power Electronics Lab



# List of Experiments



## Power electronics lab (undergraduate level)

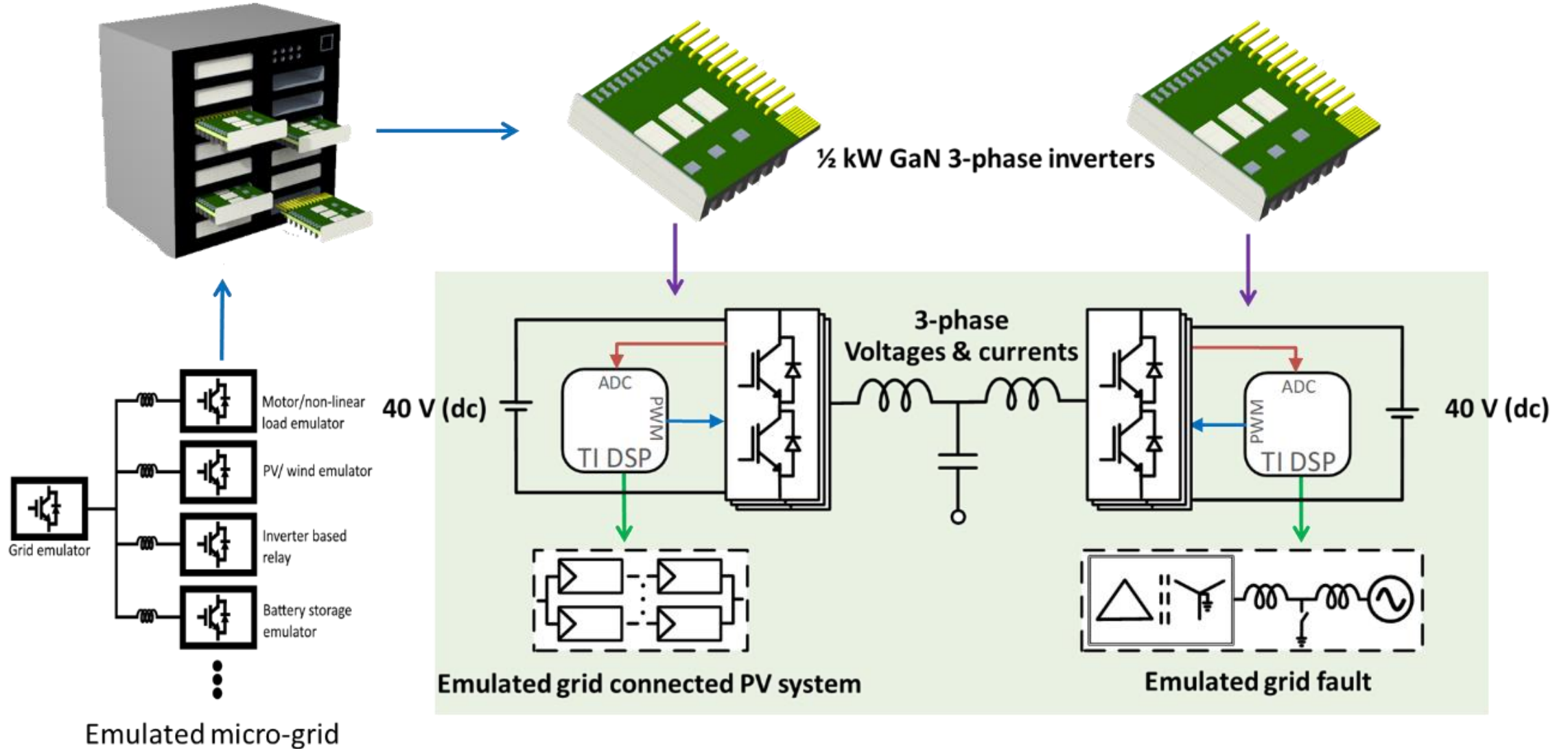
1. Si and GaN power-device characteristics
2. Buck converter
3. Boost converter
4. Buck-boost converter
5. Digital voltage mode control
6. Digital current mode control
7. Flyback converter
8. Forward converter
9. Full-bridge converter
10. Single-phase DC-AC inverter
11. Three-phase DC-AC inverter

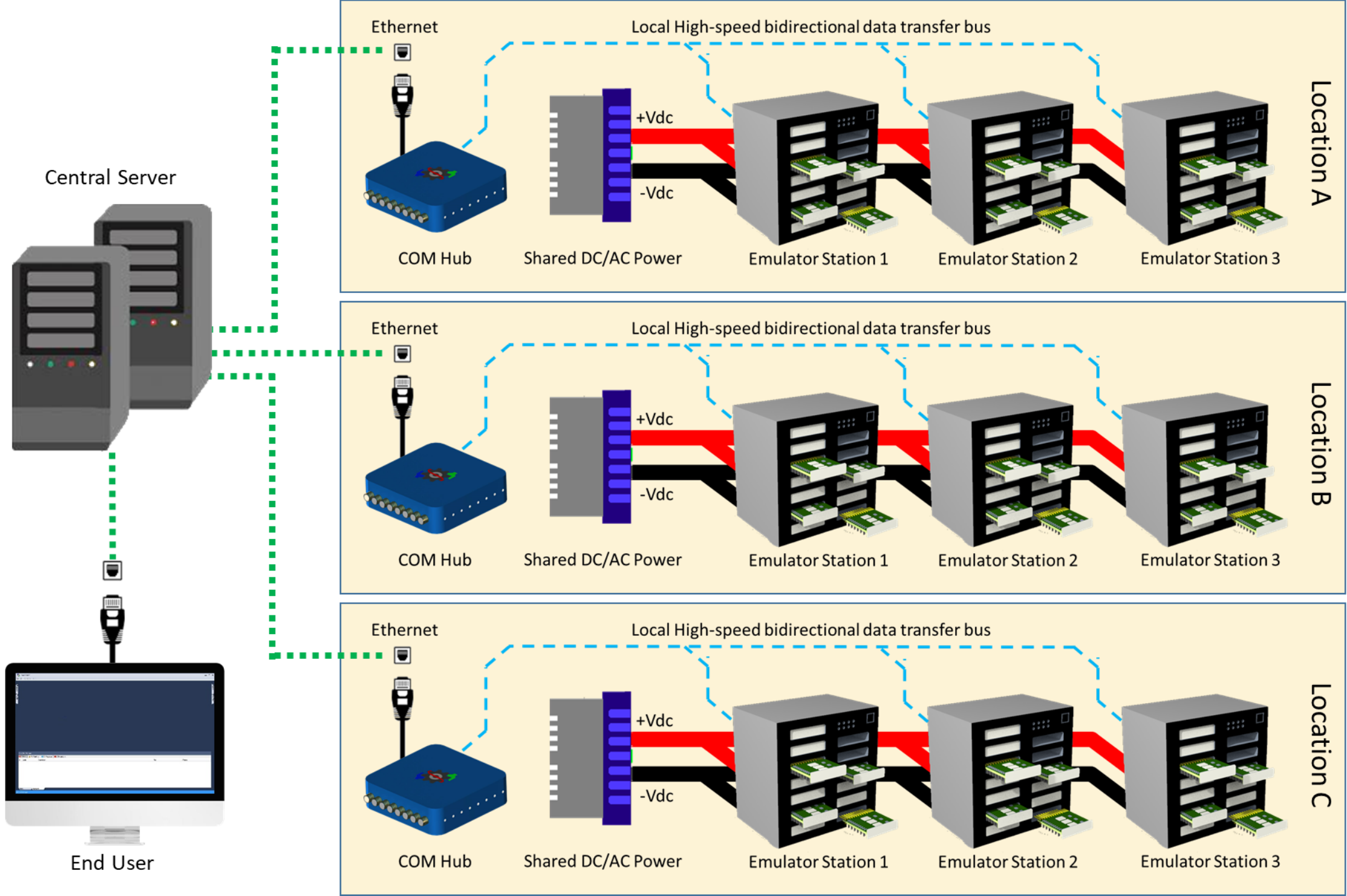
# Low-cost general purpose prototyping platform



- 
- Code-free model based design enabled rapid prototyping platform.
  - Cost \$25 and has similar capability as \$9000 dSpace platform.
  - Measures 1 x 1 in and can be easily plugged into any real-world control application.
  - 100 MHz clock, 16 PWMs, 14 ADCs, 33 GPIOs, 1 SPI/SCI/CAN COM.
  - On-board datalogger and programmer that interfaces directly with Workbench.

# Emulated Power Systems Lab







# Acknowledgement

We are greatly indebted for the three grants to the University of Minnesota from the Office of Naval Research (ONR):

- a. N00014-22-1-2491 “A Low-Cost Scalable Tabletop Emulator for Shipboard Power System”
- b. N00014-19-1-2018 “Developing WBG-Based, Extremely Low-Cost Laboratories for Power Electronics, Motor Drives, and Power System Protection and Relays for National Dissemination”
- c. N00014-15-1-2391 “Web-Enabled, Instructor-Taught Online Courses”

These grants allowed the development of the Workbench simulation platform, which is available free-of-cost for educational purposes. These grants also allowed the development of a low-cost hardware laboratory, available from Sciamble Corp (<https://sciamble.com>) – a University of Minnesota startup.

Thank you



UNIVERSITY OF MINNESOTA



CUSP™