ICEM 2024, Torino, September 1-4 2024

Tutorial Proposal

TUTORIAL TITLE: Embedded Code Generation from PLECS for Electric Drives

TUTORIAL PRESENTERS (Full Names, affiliation and e-mail addresses):

Beat Arnet, Plexim, arnet@plexim.com

BIOS OF THE PRESENTERS (150 words each):

Dr. Arnet is an expert in the field of power electronics, with over 30 years of experience in power converter design for automotive and renewable energy applications. He is General Manager for Plexim Inc (USA) where he applies his industry experience to oversee the development of the PLECS embedded code generation tools. He also teaches electives courses as an adjunct instructor at the Olin College of Engineering.

Before joining Plexim, Dr. Arnet led electric drive component development at Azure Dynamics (AZD) and oversaw the design of power electronics, electric machines and control software.

Dr. Arnet holds a diploma in Electrical Engineering from the Swiss Federal Institute of Technology in Zurich (ETH) and received his Electrical Engineering PhD from the Swiss Federal Institute of Technology in Lausanne (EPFL). He is the principal inventor of 5 US patents and has authored and co-authored over 15 publications. He is a Senior Member of IEEE, and is a Member of both the Power Electronics and Industry Applications societies.

ABSTRACT (200 - 300 words):

Developing an electric drive system is a multidisciplinary endeavor. It includes not only the inverter and electric motor design, but also the development of advanced controls, which are often implemented on a microcontroller (MCU). Yet few engineers are equally skilled in hardware, electric machines, motor controls, and software design. Electrical engineers, though skilled in hardware design and motor control, usually lack formal software development training.

Meanwhile, in today's fast-paced market there's little incentive to create reusable and maintainable code, resulting in a codebase that lacks modularity, clear structure, and proper documentation. Automatic code generation from a simulation model is a solution to this problem, simplifying the development process right from the start and ties seamlessly into the simulation work that occurs during the design of control algorithms.

PLECS, coupled with the PLECS Coder, enables intuitive modeling, simulation, and implementation of controls on C2000 and STM32 MCUs. This code generation process requires no specialized software development skills or in-depth MCU knowledge. The iterative development approach using PLECS models facilitates the evolution of a design from concept to implementation, serving as both control algorithm definition and documentation.

Benefits include simplified configuration of MCU peripherals without any register-level knowledge, offline software-in-the-loop (SIL) simulation for control verification, and identification of discrepancies between the model and C code. Real-time testing and debugging on the target MCU are facilitated through PLECS, allowing users to observe and interact with real-time data.

In this tutorial, participants are encouraged to bring laptops for a hands-on and interactive walkthrough of the PLECS code generation process. Participants will learn how to design and configure a FOC algorithm with synchronous sampling and nested controls. A PLECS model will be built up incrementally to demonstrate the concepts of basic powerstage management and protections, open-loop control, and closed-loop control with synchronous frame decoupling and anti-windup.

List of contents:

1. Introduction to offline control system modeling in PLECS (30 minutes)
    a. Power electronics modelling
    b. Electric machine modelling
    c. Signal processing and control blocks
        i. Transfer functions
        ii. Discontinuous and non-linear blocks
        iii. PID controllers
    d. State Machine environment
    e. Advanced blocks (e.g., C-Script)
2. Introduction to PLECS code generation workflow (30 minutes)
    a. Exploration of Target Support Package block libraries
    b. Trigger event concept
    c. Code generation subsystem concept
    d. Multi-tasking environments
    e. PLECS Coder GUI and target setup

3. Interactive step-by-step realization of a field oriented motor control (60 minutes)
    a. Set-up interrupt driven multi-tasked control loops
    b. Configure synchronous sampling
    c. Implement an open-loop control model to identify motor parameters
    d. Implement and tune a field-oriented closed-loop control
    e. Add decoupling and anti-windup to the synchronous frame controller

Do not forget to attach one photo of each presenter to this document