Tutorial - Buck Converter

Description

This tutorial guides you through the development of a **buck converter control system** using the **DL 2106T06** module with bidirectional serial communication between a **Simulink-based Host Model** which will be running in the master (PC) and a **Target Model** running on **DL RCPCORE** module. The system implements basic control, monitoring, and fault protection based on real-time analog acquisition and serial communication.

To understand better how communication between the host and target models works, it is advised to carefully review the Host-Target Communication demos in getting started.

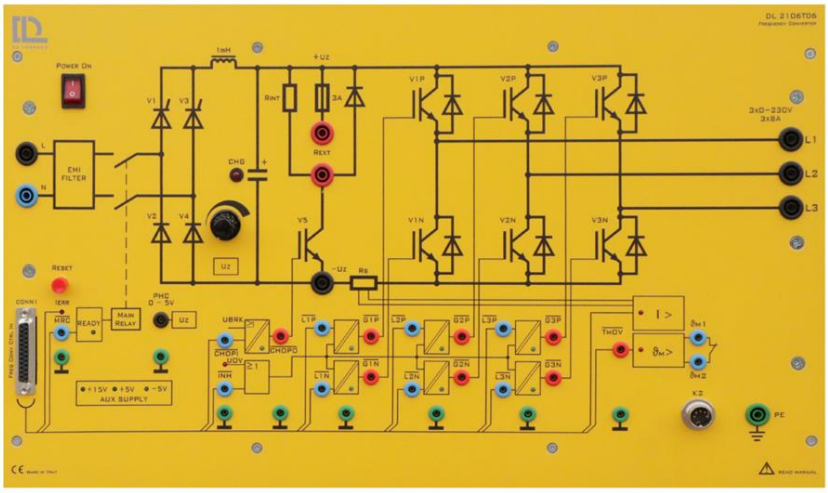
Learning Objectives

By the end of this tutorial, learners will be able to:

1. Understand how to set up a Buck converter with the DL 2106T06.
2. Construct master–slave models in Simulink, with the master model dedicated to debugging and monitoring, while the slave model executes control algorithms, performs data acquisition, and integrate protection logic with constraints.
3. Implement serial communication between the master and slave.

Part 1: Set up a Buck converter with the DL 2106T06

The DL 2106T06 is a module that integrates a controlled single-phase rectifier, a DC link, and three IGBT legs at the output. The module was originally designed to function as a frequency converter. Nevertheless, owing to the independent controllability of its output inverter, it can also be employed to construct other types of power converters.



To use the DL 2106T06 frequency converter as a **buck converter**, the system must be reconfigured in the following way:

- **Leg 1** of the output inverter: only the **upper IGBT** (V1P) is controlled using PWM. The lower IGBT (V1N) is forced to be off so it will function as the freewheeling diode of the buck converter.
- **Leg 2**: both IGBTs are off since this leg is not used in the buck converter.
- **Leg 3**: The lower IGBT (V3N) is forced on to provide a valid return path for the buck converter current back to the power supply.

Part 2: Construct Simulink Host and Target Models

To open the example for reference, run **openDLDemo('buck')** in the command window.

Create a folder for the new project

💡 *It is recommended that each project has its own separate folder, and the folder path does not contain any spaces.*

Navigate the MATLAB working directory to the new project folder

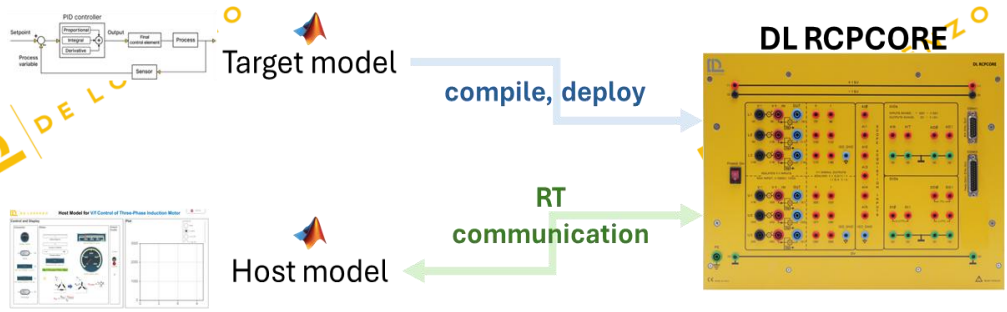
Execute the CreateNew command in the MATLAB Command Window and enter the project name in the pop-up dialog box.

💡 *Avoid using special characters and spaces in the project name.*

Wait until the "Busy" status in the lower-left corner of MATLAB disappears, and no warnings or errors are displayed in the Command Window.

Two models are created by the system:

- **Target model**: This model will be compiled, deployed, and executed on the DL RCPCORE hardware. Its primary responsibility is to execute control algorithms and support real-time communication with the host model.
- **Host model**: This model runs on the PC in simulation mode. Its role is to debug parameters in the target model and display specific signals.



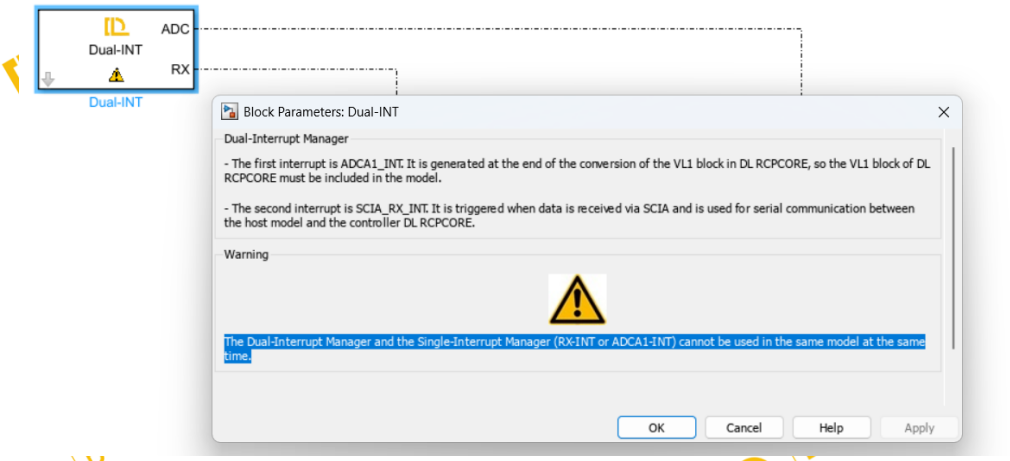
An **.m file** for setting parameters will also be generated automatically.

Step 1: Target Model Structure and Basic Blocks

In the target model, the essential blocks have been added. Read the descriptions and warnings of these blocks to understand their significance and function as essential blocks.

Navigate through the Simulink Library Browser to locate the blocks in the DL library that are required to interface the model with the DL PEL-HIL hardware modules. Add the following blocks:

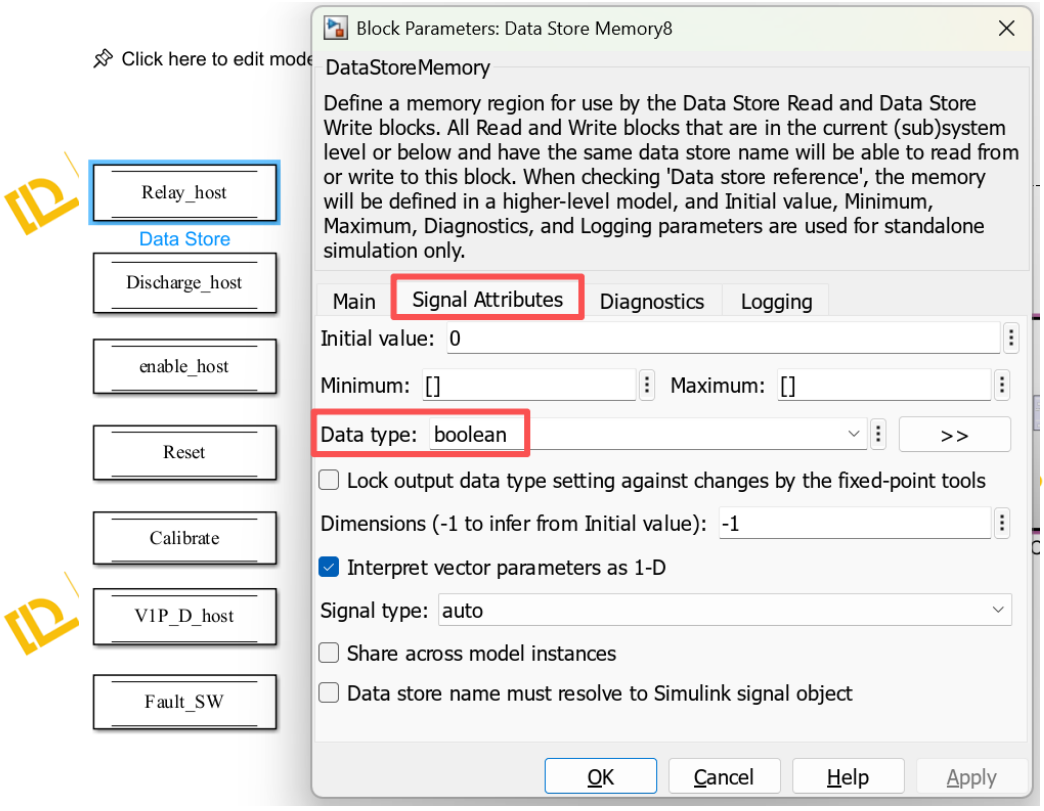
- Dual-INT: The **Dual-Interrupt block** is used to **enable hardware-triggered execution** of two key tasks on the target microcontroller: the **ADC interrupt** (ADC), which ensures precise and timely execution of ADC conversion and related control routines, and the **RX interrupt** (RX), which allows efficient reception of serial data only when needed, saving processing resources. By relying on hardware interrupts instead of fixed sample times, the model becomes more reasonable and practical.



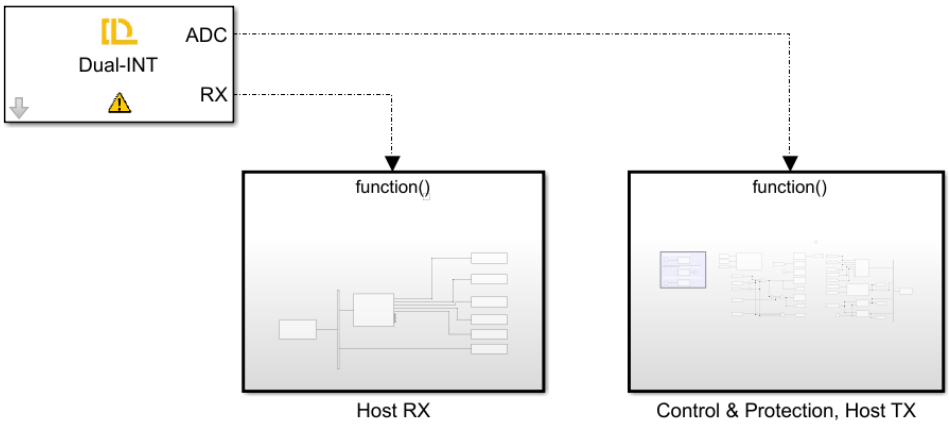
- Preassign key variables in RAM for real-time access using the **Data Store Memory** block.
    - **Relay_host**
      Boolean
      Relay control signal sent from the host; whether the target follows it depends on the protection algorithm.
    - **Discharge_host**
      Boolean
      DC-link discharge control signal sent from the host; whether the target follows it depends on the protection algorithm.
    - **enable_host**
      Boolean
      Enable control signal sent from the host for IGBT V1P; whether the target follows it depends on the protection algorithm.

- **Fault_SW**
  Boolean
  Based on the results of the software protection algorithm specifically defined for this project. Please distinguish it from hardware fault triggered by hardware protection. The determination of hardware fault is performed entirely by the hardware and follows fixed rules.
- **Reset**
  Boolean
  A signal sent by the host to clear latched software faults.
- **Calibrate**
  Boolean
  A signal sent by the host to enable hardware data acquisition offset calibration.
- **V1P_D_host**
  Single
  A signal sent by the host to drive IGBT V1P with a duty cycle ranging from 0 to 1.

Do not forget to specify the Data type of the stored data through Signal Attributes tab, as it is important for efficient and correct communication. Please refer to ***Host-Target Communication – Fundamentals in DL PEL-HIL Getting Started to understand the recommended type of data to be used.***

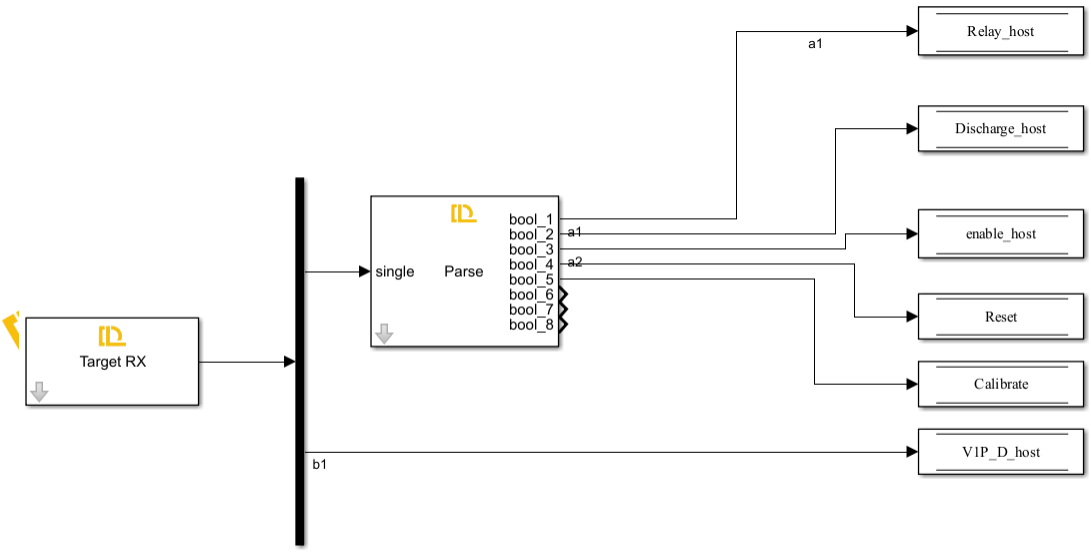


- Create two subsystems and connect them to the two outputs of the Dual-INT block to perform their specified functions as explained before.



The first subsystem is for receiving serial data from the host through Target **RX Block** and writing those data to their specified data store memory blocks to allow them to be read elsewhere in the model.

💡 *please refer to **Host-Target Communication – Fundamentals in DL PEL-HIL Getting Started** to understand how the serial communication between host and target works, and what parse blocks is used for.*



In this model, the data received from the host include **Relay_host**, **Discharge_host**, **Enable_host**, **Reset**, **Calibrate**, and **V1P_D_host**. All of them are stored in the data store memory with their data types specified.

The second subsystem, triggered by the ADC interrupt, performs data acquisition, converter control, protection, and data transmission to the host.

💡 *To enable periodic sampling, the modules in the DL Library are designed as follows: using a specific point of the carrier that generates the V1P switching pulse*

as the reference, the Start of Conversion (SOC) for analog data acquisition is triggered. When the data acquisition of VL1 (of DL RCPCORE) is completed, an interrupt event is generated. This process repeats continuously. Therefore, this subsystem operates at the PWM frequency of V1P.

Ensure V1 or V1P/N block exists if any analog data acquisition block is used.

Ensure VL1 block and the V1 or V1P/N block exist if the Dual-INT or ADCA1-INT block is used.

In this subsystem, add the following blocks from DL library:

- **ctrl en**
  Automatically added and configured by the CreateNew command.
  This block determines whether the DL RCPCORE control outputs are enabled. It prevents the outputs from being undefined during target model compilation and deployment. The outputs will be enabled only after deployment, so the input of this block is often set to a fixed Boolean value of 1.
- **VL1**
  Automatically added and configured by the CreateNew command.
  VL1 measures high voltage from channel VL1 of the DL RCPCORE. This block triggers ADCINT1 interrupt and is essential for scheduling with ADC interrupt whether it will used or not.
- **V1 or V1P/N**
  Automatically added and configured by the CreateNew command.
  The block controls the 1st IGBT leg of the inverter of DL 2106T06 from the Simulink model by duty cycle ranging from 0 to 1.
  It enables the Start of Conversion (SOC) for all other analog measurement blocks. Therefore, it must be included whenever analog signal measurement is required.
  To keep leg 1 from being uncontrolled, V1 or V1P/N must be present with a valid duty cycle, even if the leg is not used in the topology.
- **V2 or V2P/N**
  Automatically added and configured by the CreateNew command.
  The block controls the 2nd IGBT leg of the inverter of DL 2106T06 from the Simulink model by duty cycle ranging from 0 to 1.
  To keep leg 2 from being uncontrolled, V2 or V2P/N must be present with a valid duty cycle, even if the leg is not used in the topology.
- **V3 or V3P/N**
  Automatically added and configured by the CreateNew command.
  The block controls the 3rd IGBT leg of the inverter of DL 2106T06 from the Simulink model by duty cycle ranging from 0 to 1.

To keep leg 3 from being uncontrolled, V3 or V3P/N must be present with a valid duty cycle, even if the leg is not used in the topology.

- **Fault**
  This block reads whether the DL 2106T06 has a hardware fault.
- **Relay**
  This block controls the input relay of the DL 2106T06.
- **Vdc**
  This block acquires the DC link voltage from the DL 2106T06.
- **Iinv**
  This block acquires the DC link output current, i.e., the inverter's supply current.

The control input is determined by both the host-issued **Discharge_host** command and the protection algorithm. When the host initiates DC link discharge to release residual charge, **the target checks the relay state**: the discharge command is passed to the hardware only if the relay is open, preventing overload of the discharge resistor.

As shown here, the Discharge command executed by the target (**Discharge_target**) may differ from the host-issued command (**Discharge_host**), for safety and justified reasons.

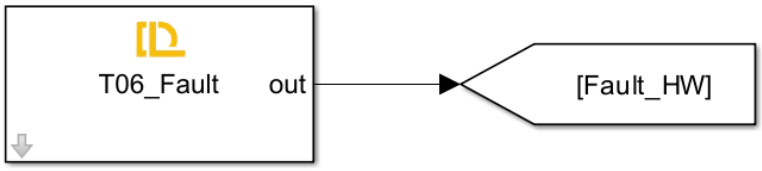
Step 2: Target Model Fault Detection & Protection Logic

1. **Fault detection**

Faults are classified into two types: hardware-reported DL 2106T06 faults and software-detected faults defined by the user for the application.

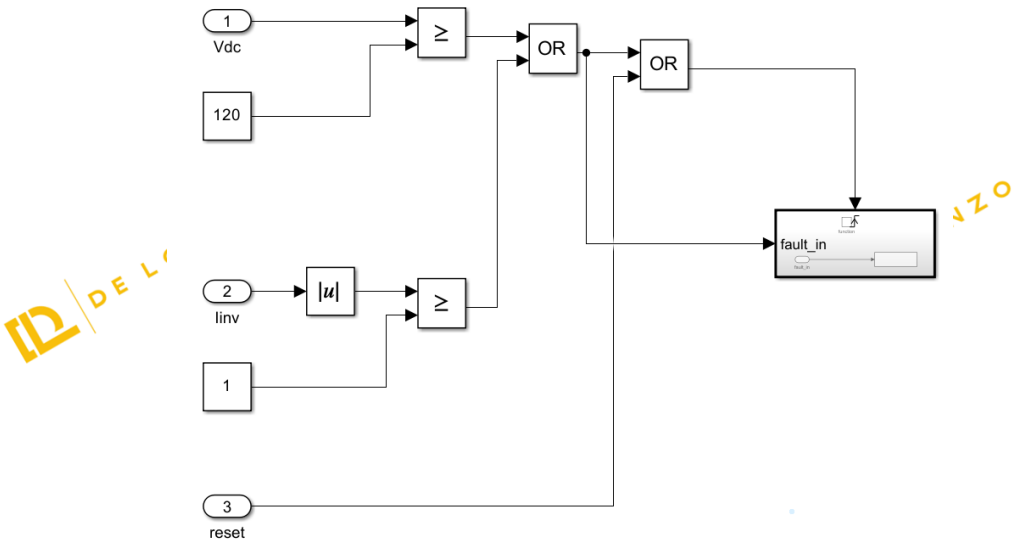
Fault_HW

As mentioned earlier, the Fault block monitors hardware faults in the DL 2106T06.



Fault_SW

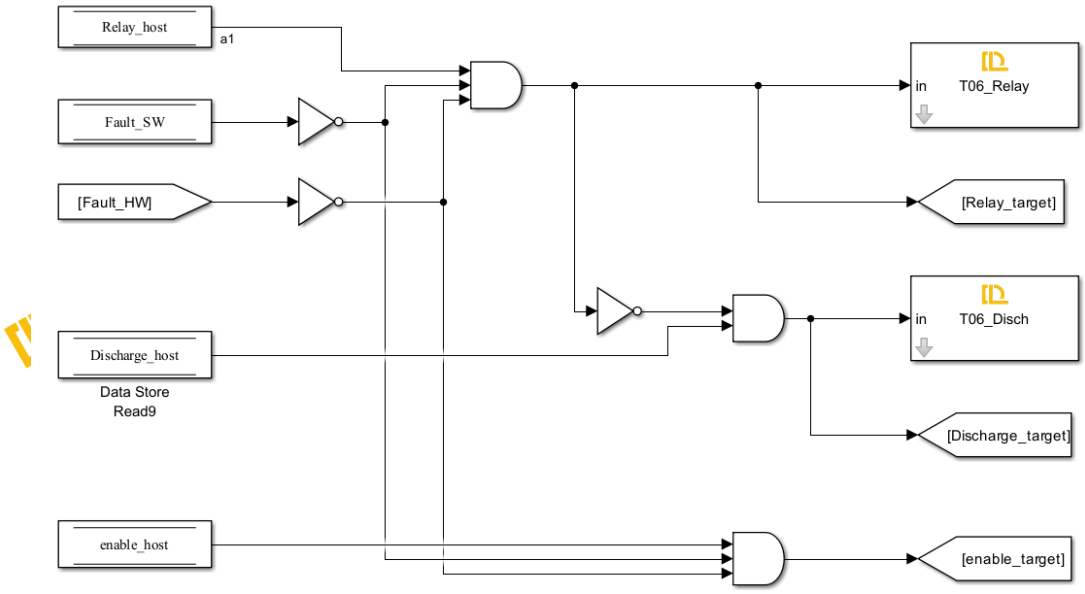
To ensure safe operation, a fault is triggered when the DC link exceeds 120 V or the absolute inverter supply current exceeds 1 A. Once a fault occurs, it is latched — meaning the fault remains active even if the conditions return to normal. The fault can only be reset if the user presses the reset button and both the voltage and current fall below their respective thresholds.



2. Protection Logic

In the target model, several control constraints are implemented to ensure safe and reliable operation of the buck converter. These include:

- The **enable_target** signal is activated only when:
  - enable_host = 1 **and**
  - No hardware or software faults are present.
- The **Relay_target** signal (which allows connection of the AC source to the DC link) is set only when:
  - relay_host = 1 **and**
  - No hardware or software faults are present.
- The **discharge_target** signal (used to safely discharge the DC link capacitor) is activated only when:
  - Discharge_host = 1 **and**
  - Relay_target = 0 (ensuring no charging source is connected while discharging to avoid overload).
- The **V1P_target** signal (which controls the upper switch of the first leg, acting as the buck switch) is updated with the value from the host (V1P_host) **only if** enable_target = 1; otherwise, it is forced to zero.



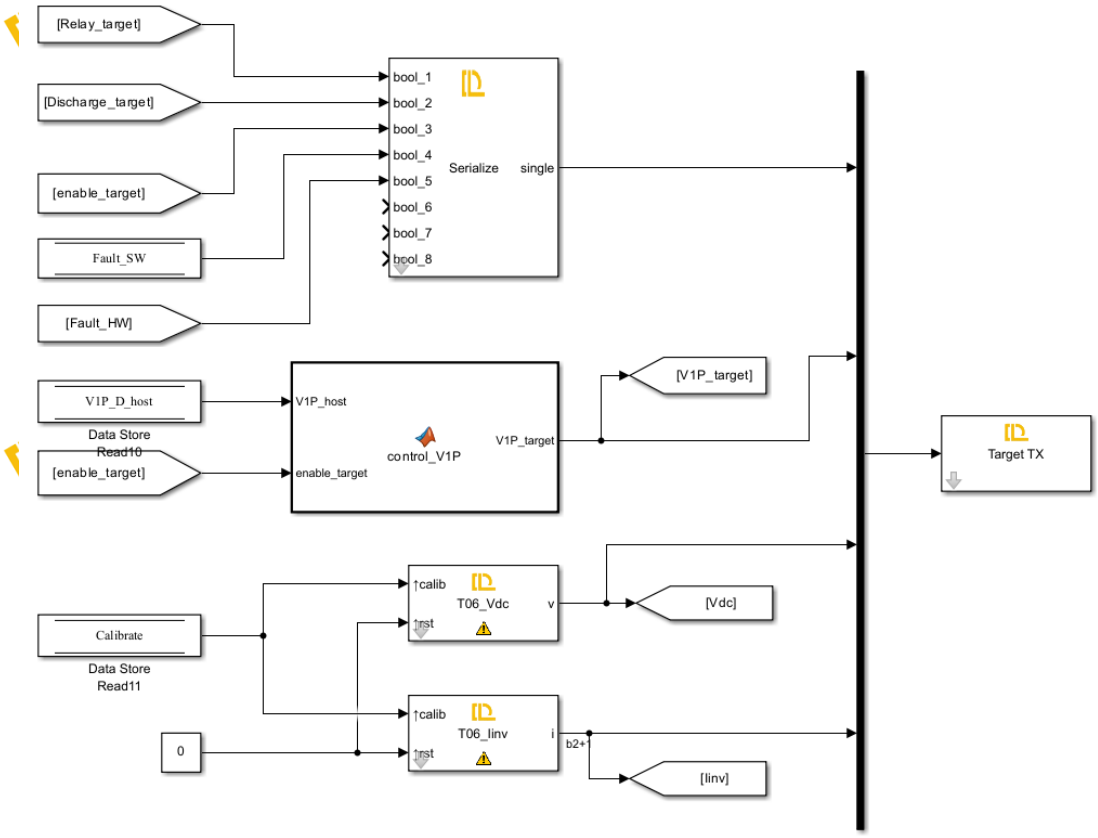
These constraints are essential for the following reasons:

- **Safety**: They prevent power flow or switching commands when a fault is detected, reducing the risk of hardware damage or electrical hazards.
- **Controlled Discharge**: They avoid discharging the capacitor while the relay is still connected to the power supply, which could cause dangerous short circuits.
- **Predictable Behavior**: Ensuring switching only occurs when explicitly allowed by both host and safety logic improves the reliability and traceability of system behavior.

Step 3: Target Model TX

To observe the target's real-time operation in the Host model, signals including relay, DC discharge, control enable, software and hardware faults, Vdc, Iinv, etc, are transmitted back to the host.

See tutorial 'Host-Target Communication' for details.



Step 4: Target Model Build, Deploy and Start

Once deployed, the DL RCPCORE runs the target model continuously at startup, while certain parameters and commands rely on real-time host communication.

Step 5: Host Model

The Host model mainly consists of two parts: target communication, requiring proper data and timing alignment, and a dashboard for debugging and monitoring. Refer to the 'Host-Target Communication' tutorial for details.