DQ-type PLL

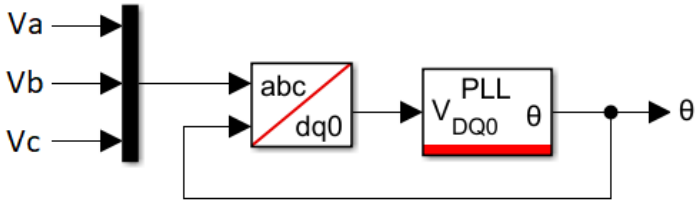
SD035 | Posted on July 12, 2022 | Updated on May 27, 2025

Jessy ANÇAY
Sales & Project Engineer
imperix

Table of Contents

The DQ-type PLL is a basic Phase-Locked Loop (PLL) used to extract the phase information of three-phase voltages. It operates by minimizing the voltage projected on the quadrature axis in a rotating frame reference. Further details on the implementation of DQ-type PLL can be found on the page: Synchronous reference frame (SRF) PLL.

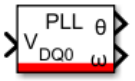
A typical use case is shown below, where the PLL block and an abc-to-dq0 transformation are used to extract the angle of three-phase voltages:



Simulink block

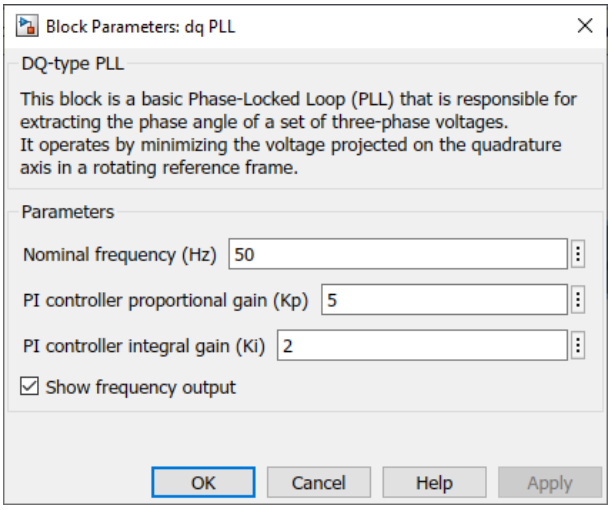
Signal specification

- The input signal V_{DQ0} is the *dq0* representation (obtained using the Park transform [abc->dq]) of a set of three-phase voltages.
- The output signal θ is the phase angle of the set of three-phase voltages, in radians.
- The output ω is the angular frequency in of the set of three-phase voltages, in rad/s.



Parameters

- `Nominal frequency (Hz)` is the three-phase voltages' nominal frequency.
- `PI controller proportional gain (Kp)` is the proportional gain of the PI controller inside the PLL.
- `PI controller integral gain (Ki)` is the integral gain of the PI controller inside the PLL.
- `Show frequency output` checkbox adds an additional output signal to the DQ-type PLL block, containing the angular frequency of the three-phase voltages.



PLECS block

The DQ-type PLL block is not available in PLECS. The PLECS block *Three-Phase PLL* can be used instead.

C++ functions

The user template located in the installation folder of CPP SDK contains an API folder with implementations of the PLLs functions. The *DQ-type PLL* functions are the following:

`ConfigDQPLL` **— Configures a DQ-type PLL**

`void ConfigDQPLL(DQPLLParameters* me, float kp, float ki, float omega0, float tsample)`Code language: C++ (cpp)

Routine to initialize the PLL based on dq transformation (loop filter on the q-axis)

It has to be called in `UserInit()`.

Parameters

- `me`: the corresponding PLL pseudo-object (parameters and state quantities)
- `kp`: proportional gain of the inner PI controller
- `ki`: integral gain of the inner PI controller
- `omega0`: nominal angular frequency (feedforward term)
- `tsample`: sampling (interrupt) time

`RunDQPLL` **— Run the DQ-type PLL**

`float RunDQPLL(DQPLLParameters* me, const SpaceVector *vin_dq0)`Code language: C++ (cpp)

Run the PLL based on DQ transformation only (minimization of the quadrature voltage). It returns the phase angle (typically of the grid voltage).

It can be called in the control interrupt routine

- me: the corresponding PLL pseudo-object (parameters and state quantities)
- ug_dq0: the reference quantity (typ. the grid voltage in dq0 reference frame)

The structures used in the above functions are detailed below:

```
/**
 *  Parameters for the DQ-based Phase-Locked Loop. Parameter omega0 refers to
 *  the nominal angular frequency of the sinusoidal signal
 */
typedef struct{
        float theta;                    // Phase angle of the grid voltage
        float omega;                    // Debug only: is not a state variable
        float omega0;                   // Default grid frequency (feedforward quantity)
        float ts;                       // Sampling interval
```

```
        PIDController PI_reg;           // Corresponding PID controller pseudo-object
} DQPLLParameters;Code language: C++ (cpp)

// Three-phase quantity in complex form (ABG or DQ0 reference frames)
typedef struct{
        float real;
        float imaginary;
        float offset;
} SpaceVector;Code language: C++ (cpp)
```