```
#ifndef __FOC_tests_double_sim_hpp__
#definde __FOC_tests_double_sim_hpp__


#include "controlSuite_double_sim.hpp"




void FOC_tests_init(void)
{
        //One time initialization.

    init_cm1();
    init_pi_spd_pi_id_pi_iq();

}


void FOC_tests(void)
{
    // This function called by the simulator every 50 uSec.




    //From /home/maiello/Desktop/Development-Simulation/InductionMotor_DTC_SVM/
controlSUITE_ref/HVACI_Sensored/HVACI_Sensored.c

    // Using the the test labeled "LEVEL 5" as a reference...

    // ================================ LEVEL 5
====================================
    //  Level 5 verifies verifies the speed regulator performed by PI module.
    //  The system speed loop is closed by using the measured speed from capture
    //  signal as a feedback.
    //
================================================================================







    // Our equivilent connections to the simulator...

    // Inputs from the simulator:  is[0], is[1], is[2], omegad, omega_r
    // Outputs to the simulator:   vdd, vdq, theta_ctrl


    //
--------------------------------------------------------------------------------
    //  Connect inputs of the CLARKE module and call the clarke transformation
macro
    //
--------------------------------------------------------------------------------
        clarke1.As = is[0];       // Phase A curr.
        clarke1.Bs = is[1];       // Phase B curr.
        clarke1.Cs = is[2];       // Phase C curr.
```

```
        CLARKE1_MACRO(clarke1)

    //
-------------------------------------------------------------------------------
    //  Connect inputs of the PARK module and call the park trans. macro
    //
-------------------------------------------------------------------------------
        park1.Alpha = clarke1.Alpha;
        park1.Beta  = clarke1.Beta;
        park1.Angle = cm1.Theta;
        park1.Sine  = sin(park1.Angle);
        park1.Cosine= cos(park1.Angle);
        PARK_MACRO(park1)

        //
-------------------------------------------------------------------------------
        //  Connect inputs of the PI module and call the PID IQ controller macro
        //
-------------------------------------------------------------------------------

        pi_spd.Ref = omegad;
        pi_spd.Fbk = omega_r;
        PI_MACRO(pi_spd)


        //
-------------------------------------------------------------------------------
        //  Connect inputs of the PI module and call the PID IQ controller macro
        //
-------------------------------------------------------------------------------
        pi_iq.Ref = pi_spd.Out;
        pi_iq.Fbk = park1.Qs;
        PI_MACRO(pi_iq)

            //For this simple test, we fix the field current as a constant for the
entire simulation.
        IdRef = 20;

        //
-------------------------------------------------------------------------------
        //  Connect inputs of the PI module and call the PID ID controller macro
        //
-------------------------------------------------------------------------------
        pi_id.Ref = IdRef;
        pi_id.Fbk = park1.Ds;
        PI_MACRO(pi_id)

        //
-------------------------------------------------------------------------------
        //  Connect inputs of the INV_PARK module and call the inverse park trans.
macro
        //  (In our model, pi_id connects to vdd and pi_iq connects to vqd. We use
are own SV PWM algorithm which takes care of INV_PARK calculation)
        //
-------------------------------------------------------------------------------
        vdd = pi_id.Out;
        vqd = pi_iq.Out;
        theta_ctrl = cm1.Theta;
```

```
        //
-------------------------------------------------------------------------------
        //    Connect inputs of the CUR_MOD module and call the current model
        //    calculation function.
        //
-------------------------------------------------------------------------------
        cm1.IDs = park1.Ds;
        cm1.IQs = park1.Qs;
        cm1.Wr = omega_r;
        CUR_MOD_MACRO(cm1)

}


#endif
```