Pacdrive3 - Servoazionamento Lxm62

1. Configurazione PLC

L'asse deve essere impostato come azionamento lineare Lexium62 nel Machine Expert e l'identificazione del motore deve essere impostata come motore senza targhetta.

2. Impostazione della targhetta

```
st_MotorDataRW_01.i_xEnable := TRUE;                        // Enable the write function
st_MotorDataRW_01.i_etStorageLocation := mtp.ET_StorageLocation.Drive;  //  Storage Location (encoder or drive)
st_MotorDataRW_01.i_sFilename := 'ide0:MDF/UserMotorData.mdf';      //  file name of the typlete for drive
st_MotorDataRW_01.i_sFilenameBLH := 'ide0:MDF/';            //      file name of the typlete for encoder
st_UserMotorData_01.etMotorType := MTP.ET_MotorType.LinearPMSM;     // type of the motor
st_UserMotorData_01.sMotorname := 'Nytek';  //Armonic Drive         //  string of the motor name
st_UserMotorData_01.sMotorSerialNumber := '33333';              //  string of serial number
st_UserMotorData_01.sMotorArticleNumber := '444444';            //      string of article number
st_UserMotorData_01.stMotorDataPMSM.uiEncoderType := mtp.ET_EncoderType.SincosLinear; //    encoder type
st_UserMotorData_01.stMotorDataPMSM.uiNominalSpeed := 3998; // mm/s
st_UserMotorData_01.stMotorDataPMSM.rNominalVoltage := 220.0;
st_UserMotorData_01.stMotorDataPMSM.rNominalCurrent := 2.1; // A
st_UserMotorData_01.stMotorDataPMSM.rPeakCurrent := 8.0; // A , OPTIONAL, std: Nom*1.5
st_UserMotorData_01.stMotorDataPMSM.rContStallCurrent := 2.1; // A
st_UserMotorData_01.stMotorDataPMSM.rConstStallTorque := 110; // N, OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.rPeakTorque := 880; // Nm, OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.rPhaseResistance := 12; // Ohm , OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.rQuadraturePhaseInductance := 18000; // uH , OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.rDirectPhaseInductance := 16200; // uH
st_UserMotorData_01.stMotorDataPMSM.uiRotatingFieldDirection := 0; // OPTIONAL, std: 0
st_UserMotorData_01.stMotorDataPMSM.rEMK_Constant := 18;


(*************************************************)
st_UserMotorData_01.stMotorDataPMSM.uiPolePair := 1;                Number of pole pairs
(*************************************************)


st_UserMotorData_01.stMotorDataPMSM.uiMaxMotorTemperature := 90; //°C , OPTIONAL, std: 130
st_UserMotorData_01.stMotorDataPMSM.uiTempSensorType := 1;
st_UserMotorData_01.stMotorDataPMSM.uiTempSensorResistanceOvertemp := 4000; // Ohm, OPTIONAL (verw bei SensorType = 1), std: 4000
st_UserMotorData_01.stMotorDataPMSM.auiTempSensorCharacteristic[0] := 0; // Ohm, OPTIONAL (verw bei SensorType = 2)
st_UserMotorData_01.stMotorDataPMSM.rInsulationSystemVoltage := 0; // V, OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.uiEncoderMaxSpeed := 0; // Umin, OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.uiEncoderMaxTemp := 0; // °C, OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.uiEncoderTempSensor := 0; // OPTIONAL, std: 0
st_UserMotorData_01.stMotorDataPMSM.uiEncoderNumberOfTurns :=0; // Obligatorisch für Geber ohne Hiperface
st_UserMotorData_01.stMotorDataPMSM.uiEncoderLinesPerRevolution := 0; // Obligatorisch für Geber ohne Hiperface
st_UserMotorData_01.stMotorDataPMSM.uiModelC1 := 0; // OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.uiModelA12 := 0; // OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.uiModelDeltaA := 0; // OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.uiModelDeltaB := 0; // OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.uiMaxSpeed := 2500; // mm/s
//st_UserMotorData_01.stMotorDataPMSM.udiMotorIntertia := 6300; // g
st_UserMotorData_01.stMotorDataPMSM.udiMotorInertia := 1500;
st_UserMotorData_01.stMotorDataPMSM.uiBrake := 0;
st_UserMotorData_01.stMotorDataPMSM.uiBrakeDisconnectionTime := 0;
st_UserMotorData_01.stMotorDataPMSM.uiBrakeCouplingTime := 0; // ms, OPTIONAL, std: 100
st_UserMotorData_01.stMotorDataPMSM.rBrakeMinVoltage := 0; // V, OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.rBrakeMaxVoltage := 0; // V, OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.uiBrakeNomCurrent := 0; // V, OPTIONAL
st_UserMotorData_01.stMotorDataPMSM.uiThermalConstant := 1000; // ms, OPTIONAL, std: 1000

// For Linear Motors

st_UserMotorData_01.stMotorDataPMSM.rPeriodLength := 30000.0;           Equal to polepairpitch
st_UserMotorData_01.stMotorDataPMSM.rPolePairPitch := 30000.0;          length of N-S poles
```

3. Scrivere il file della targhetta nella scheda flash

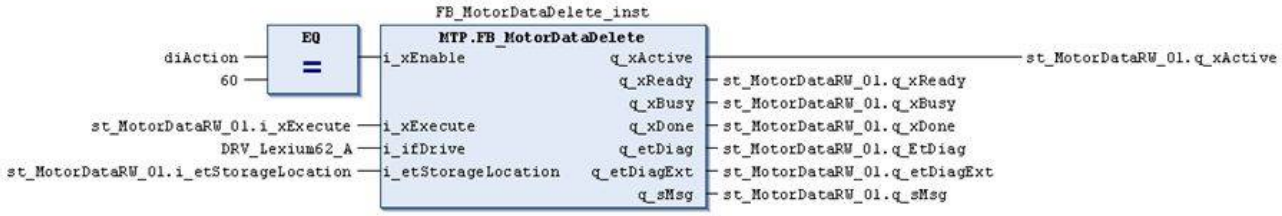
Per creare il file della targhetta nella flash, devi usare questa funzione:

```
xRet := MTP.FC_MotorDataFileCreate(i_sFilename := st_MotorDataRW_01.i_sFilename, i_stUserMotorData := st_UserMotorData_01,
                                   q_sMsg => sMsg_t,
                                   q_etDiag => etDiag_t);
```

Crea un file nella flash con i dati della struttura.

4. Eliminare i dati attuali nell'azionamento

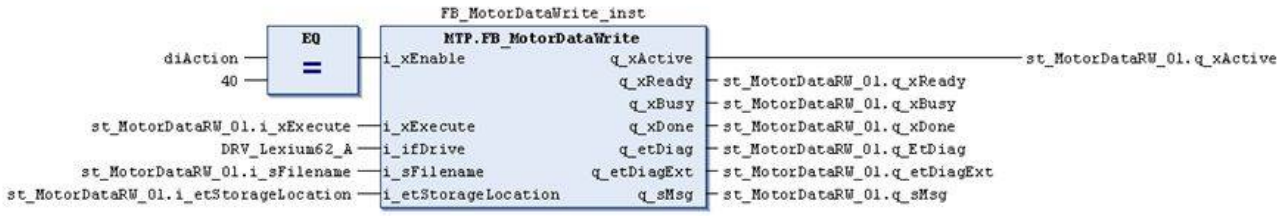
Prima di tutto devi impostare la fase Sercos a 2.
Poi devi eliminare tutti i dati nell'azionamento con questo blocco funzione.



Se non ci sono dati, il risultato mostra questa situazione. Se tutto il processo procede correttamente, il messaggio è “Done”.

5. Scrivere la targhetta nell'azionamento

Sercos ancora in fase 2.
Per scrivere i dati devi usare il blocco funzione.



6. Spegnere l'azionamento

Imposta la fase Sercos a fase 0 e poi a fase 4. Oppure spegni - accendi l'azionamento.
Se tutto è a posto, ottieni la luce verde sull'azionamento.

7. Commutazione motore

Accendi l'alimentazione.
Imposta PowerSupplyCheckSet del PSD a TRUE. Imposta a 2 / testa il MotorCommutationControl.

Il motore dovrebbe muoversi e la procedura di commutazione inizia.
Alla fine il MotorCommutationState ti fornisce il feedback corretto.

Porta a 0 il MotorCommutationControl e spegni il PowerSupplyCheckSet.