

Prog1

---

Table of Contents

Prog1 ..... 3

## Prog1

### Local Variables

Name	Alias	Data Type	Dimension	Initial Value	Project Value	Comment	Direction	Retained	String Size
step_1		BOOL		TRUE			Var	False	
step_2		BOOL		FALSE			Var	False	
step_3		BOOL		FALSE			Var	False	
step_4		BOOL		FALSE			Var	False	
step_5		BOOL		FALSE			Var	False	
MSG_MODBUS2_REMOTE_LEVER		MSG_MODBUS2		...	...		Var	False	
MB_LOCPAR_REMOTE_LEVER		MODBUS2LOC PARA		...	...		Var	False	
MB_TARPAR_REMOTE_LEVER		MODBUS2TAR PARA		...	...		Var	False	
MSG_MB_LOCADDR_REMOTE_LEVER		MODBUSLOCADDR		...	...	contains the value to write to REMOTE_LEVER modbus address	Var	False	
MB_LOCPAR_screwdriving_state		MODBUS2LOC PARA		...	...		Var	False	
MB_TARPAR_screwdriving_state		MODBUS2TAR PARA		...	...		Var	False	
MSG_MB_LOCADDR_screwdriving_state		MODBUSLOCADDR		...	...		Var	False	
MB_LOCPAR_torque_angle		MODBUS2LOC PARA		...	...		Var	False	
MB_TARPAR_torque_angle		MODBUS2TAR PARA		...	...		Var	False	
MSG_MB_LOCADDR_torque_angle		MODBUSLOCADDR		...	...	will contain the TORQUE and ANGLE data read from the MODBUS request	Var	False	

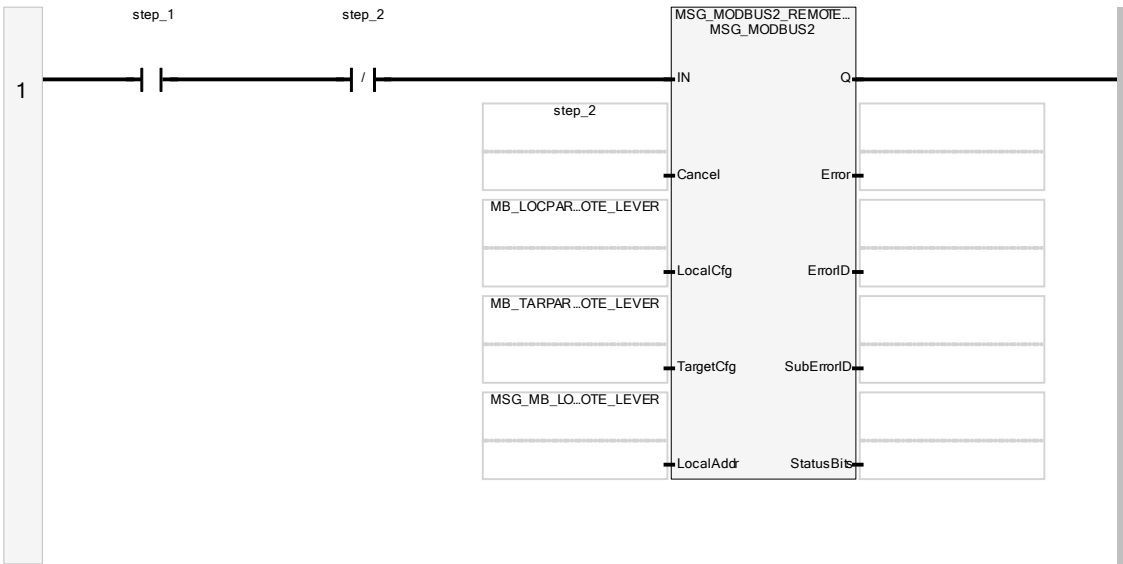
Name	Alias	Data Type	Dimension	Initial Value	Project Value	Comment	Direction	Retained	String Size
MB_LOCPAR_change_program		MODBUS2LOC PARA		...	...		Var	False	
MB_TARPAR_change_program		MODBUS2TAR PARA		...	...		Var	False	
MSG_MB_LOCADDR_change_program		MODBUSLOCA DDR		...	...	contains the value to write to currentpro gram modbus address	Var	False	
MSG_MODBUS2_get_torque_angle		MSG_MODBU S2		...	...		Var	False	
MSG_MODBUS2_change_program		MSG_MODBU S2		...	...		Var	False	
TON_5		TON		...	...		Var	False	
MSG_MODBUS2_screwdriving_state		MSG_MODBU S2		...	...		Var	False	
TON_1		TON		...	...		Var	False	
R_TRIG_1		R_TRIG		...	...		Var	False	
R_TRIG_5		R_TRIG		...	...		Var	False	
R_TRIG_4		R_TRIG		...	...		Var	False	
R_TRIG_3		R_TRIG		...	...		Var	False	
R_TRIG_2		R_TRIG		...	...		Var	False	

## Rung1 ASCII

**XIC** step\_1 **XIO** step\_2 **MSG\_MODBUS2** MSG\_MODBUS2\_REMOTE\_LEVER step\_2 MB\_LOCPAR\_REMOTE\_LEVER  
MB\_TARPAR\_REMOTE\_LEVER MSG\_MB\_LOCADDR\_REMOTE\_LEVER **????**

## Rung1 Diagram

**step 1-A: run screwdriver until it stops. LocalCfg conveniently offers a single or cyclic trigger. We set it with a 50ms cyclic trigger**



Rung2 ASCII

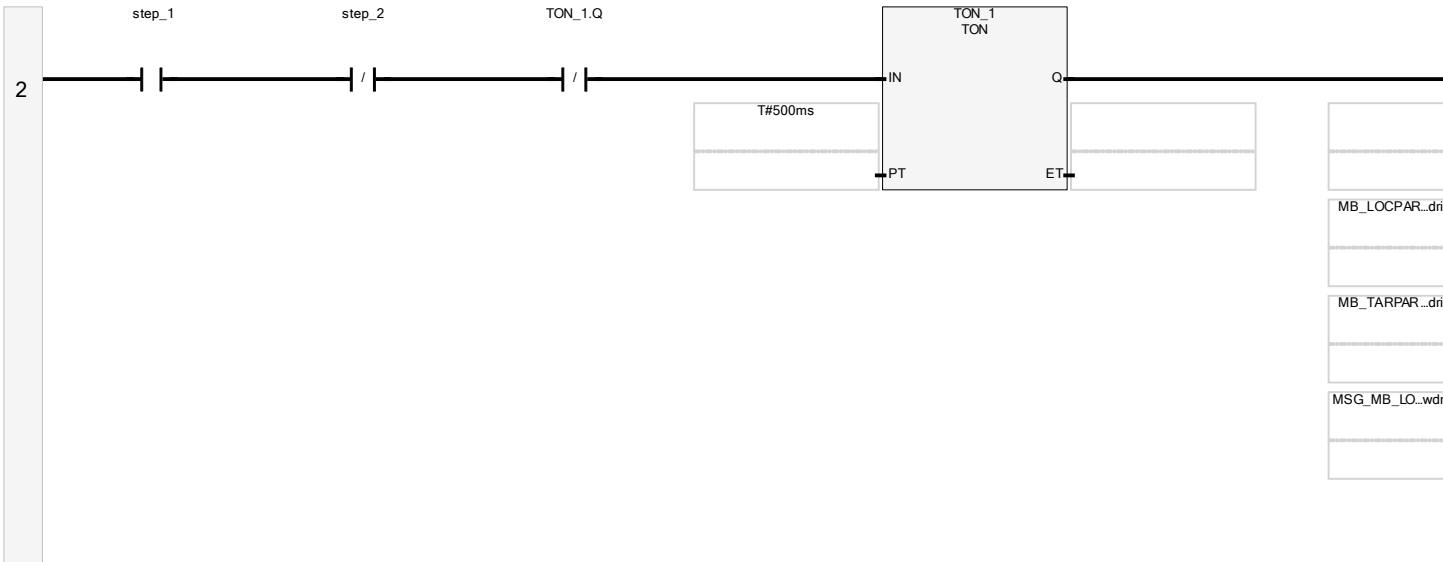
XIC step\_1 XIO step\_2 XIO TON\_1.Q TON TON\_1 T#500ms ? MSG\_MODBUS2 MSG\_MODBUS2\_screwdriving\_state ? MB\_LOCPAR\_screwdriving\_state MB\_TARPAR\_screwdriving\_state MSG\_MB\_LOCADDR\_screwdriving\_state ? ? ? ? <> MSG\_MB\_LOCADDR\_screwdriving\_state[1] 11 R\_TRIG R\_TRIG\_1 OTS step\_2

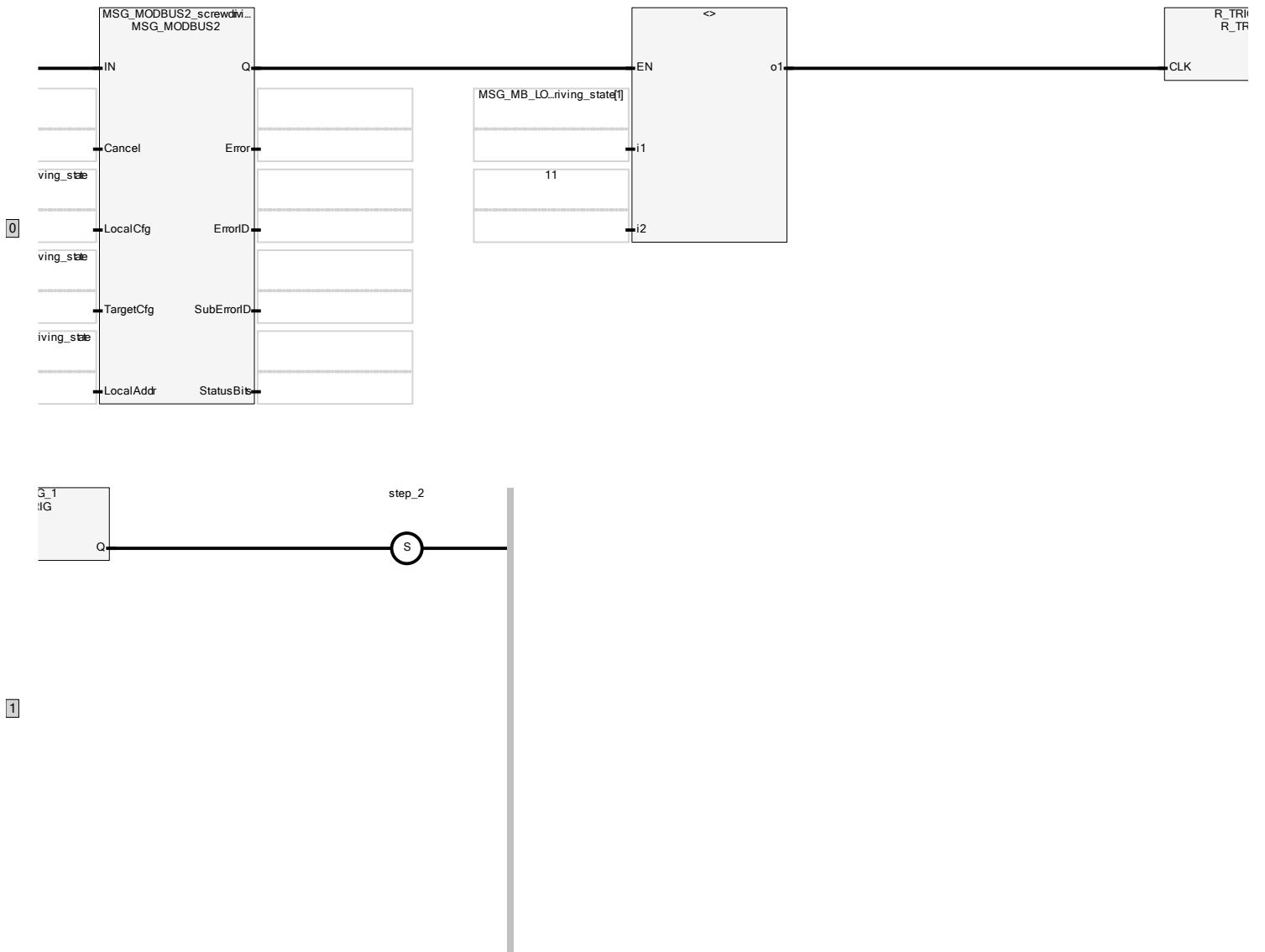
Rung2 Diagram

step 1-B: check screwdriving state starting 500ms after sending the REMOTE\_LEVER command. LocalCfg conveniently offers a single or cyclic trigger, but here we use a single trigger controlled with a timer.

The only reason we don't use it with a 500ms trigger is that we always want a ~0.5second delay between the initial send of the REMOTE\_LEVER and the checking of the screwdriving state (motor inertia, etc).

If screwdriving state is not 11=tightening (refer to K-DUCER modbus map), move to step 2. R\_TRIG here and in other steps is there because MSG\_MODBUS2 keeps its Q bit ON even if its IN bit goes off.



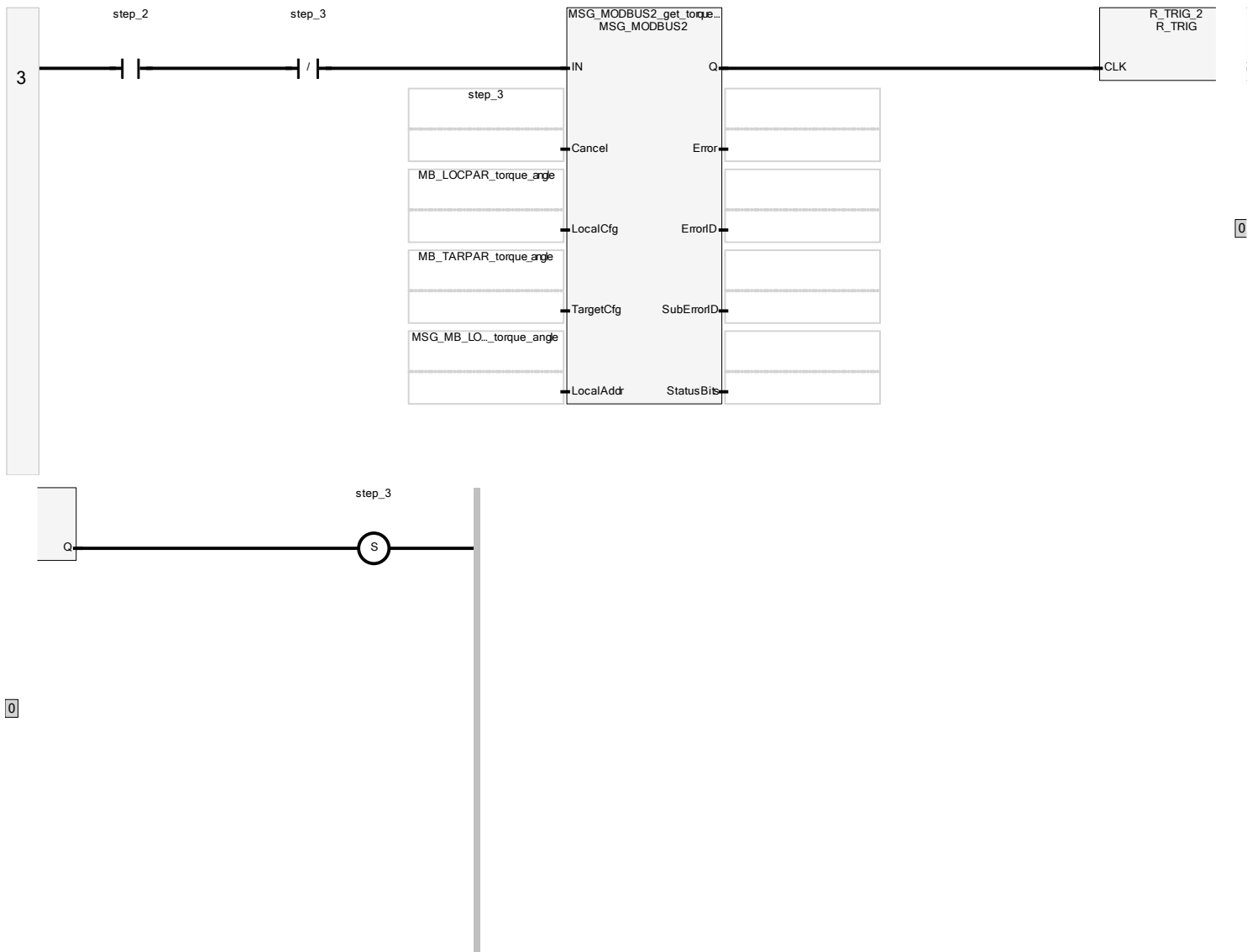


## Rung3 ASCII

XIC step\_2 XIO step\_3 MSG\_MODBUS2 MSG\_MODBUS2\_get\_torque\_angle step\_3 MB\_LOCPAR\_torque\_angle MB\_TARPAR\_torque\_angle MSG\_MB\_LOCADDR\_torque\_angle ? ? ? ? R\_TRIG R\_TRIG\_2 OTS step\_3

## Rung3 Diagram

**Step 2: get torque and angle results.** LocalCfg conveniently offers a single or cyclic trigger. We set it for a single trigger. R\_TRIG here and in other steps is there because MSG\_MODBUS2 keeps its Q bit ON even if its IN bit goes off.

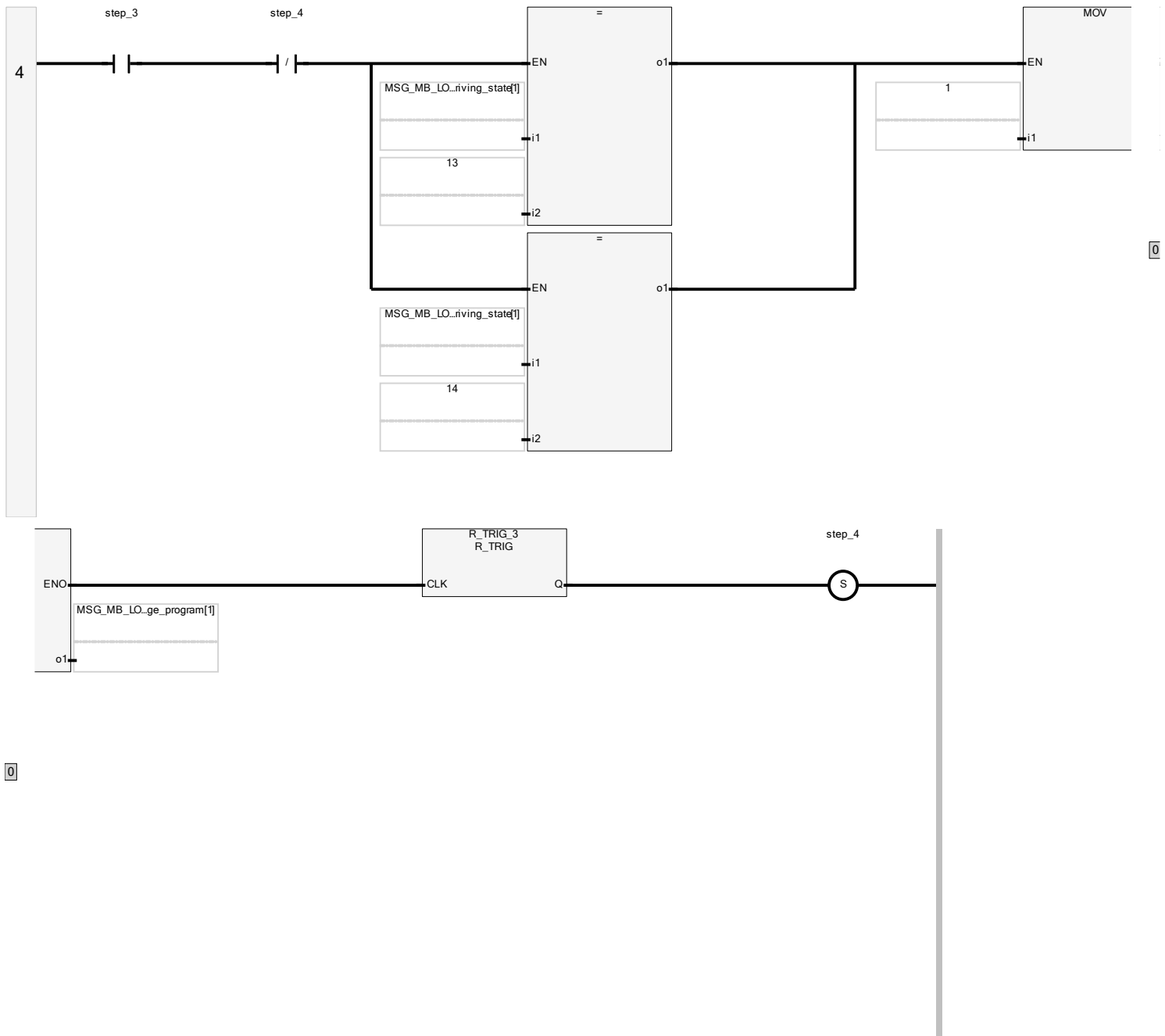


## Rung4 ASCII

**XIC** step\_3 **XIO** step\_4 **BST** = MSG\_MB\_LOCADDR\_screwdriving\_state[1] 13 **NXB** =  
 MSG\_MB\_LOCADDR\_screwdriving\_state[1] 14 **BND MOV** 1 MSG\_MB\_LOCADDR\_change\_program[1] **R\_TRIG** R\_TRIG\_3  
**OTS** step\_4

## Rung4 Diagram

**Step 3A: select program 1 if screw state is OK**  
**13 = screw OK (torque control mode) = 000D (hex)**  
**14 = angle OK (angle control mode) = 000E (hex)**



## Rung5 ASCII

**XIC** step\_3 **XIO** step\_4 **<>** MSG\_MB\_LOCADDR\_screwdriving\_state[1] 13 **<>** MSG\_MB\_LOCADDR\_screwdriving\_state[1] 14 **MOV** 2 MSG\_MB\_LOCADDR\_change\_program[1] **R\_TRIG** R\_TRIG\_4 **OTS** step\_4

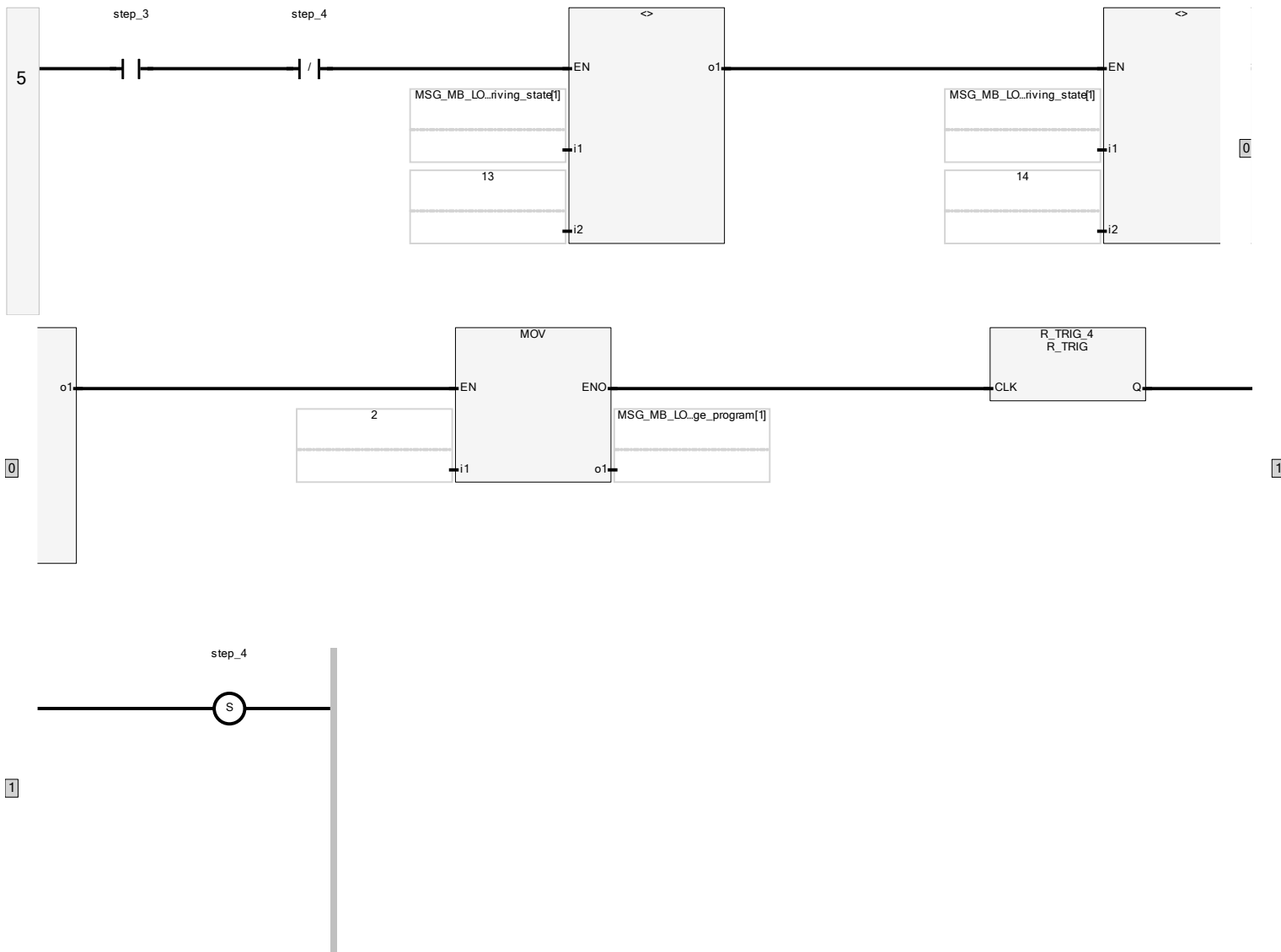
## Rung5 Diagram

**Step 3B: select program 2 if screw state is NOK**

**13 = screw OK (torque control mode) = 000D (hex)**

**14 = angle OK (angle control mode) = 000E (hex)**





## Rung6 ASCII

**XIC** step\_4 **XIO** step\_5 **MSG\_MODBUS2** MSG\_MODBUS2\_change\_program step\_5 MB\_LOCPAR\_change\_program MB\_TARPAR\_change\_program MSG\_MB\_LOCADDR\_change\_program **???** **R\_TRIG** R\_TRIG\_5 **OTS** step\_5

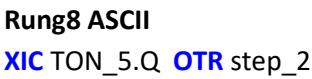
## Rung6 Diagram

**Step 4: write selected program (1 or 2) to register 7373 to change current selected screwdriving program**

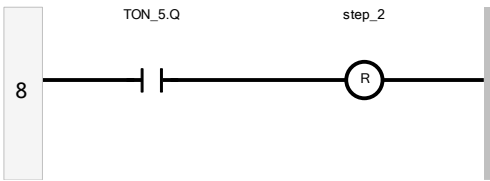


**XIC** step\_5 **XIO** TON\_5.Q **TON** TON\_5 T#500ms ?

### Step 5: wait then reset and restart PLC program



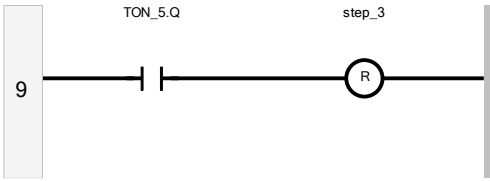
**XIC** TON\_5.Q **OTR** step\_2



Rung9 ASCII

XIC TON\_5.Q OTR step\_3

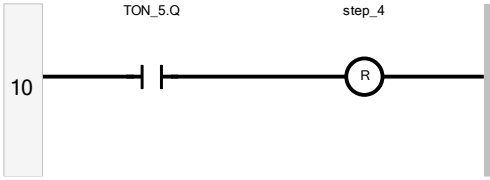
Rung9 Diagram



Rung10 ASCII

XIC TON\_5.Q OTR step\_4

Rung10 Diagram



Rung11 ASCII

XIC TON\_5.Q OTR step\_5

Rung11 Diagram

