Integration of Comau robot into the Siemens Sinumerik 840D sl control system

Nikita Kuprin[1*]

[1] *CTU in Prague, Faculty of Mechanical Engineering, RCMT, Horská 3, 128 00 Prague 2, Czech Republic*

Abstract

This article deals with the integration of Comau robotic arm into the Sinumerik system. The idea behind is to integrate a robotic arm into a machine tool as an exchangeable milling head. Such a solution would allow increasing a workspace of a machine tool by performing light machining operations with a robotic arm in places otherwise unreachable for conventional tools.

Keywords: Comau, Sinumerik, robotic arm, PLC

1. Introduction

Advancements in automation technologies allow deeper integration of robots into machine tools. Whereas previously a robot required a standalone control unit, now it is possible to integrate the robot into the machine's tool control system. Thus, significantly simplifying hardware configuration and allowing seamless synchronization between the machine tool's and the robot's axes. Such integration brings many advantages. For example, it makes possible to use the robot as active support for machining operations of thin-walled parts or even to use the robot as a milling head for light operations.

This article deals with the implementation of one of such solutions in a form of Siemens Run MyRobot /Direct Control. By using the system, a robot can be operated independently of its own controller – axes are directly connected to the Siemens Sinamics Double Motor Modules as illustrated in Figure 1.

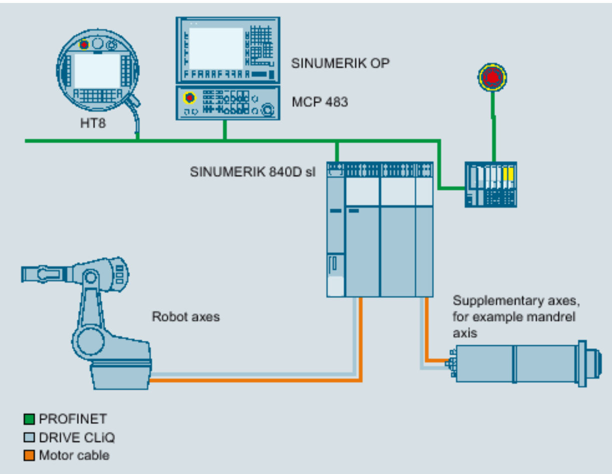


Figure 1. *Illustration of Run MyRobot / Direct Control* ***[1]***

2. Setup

Implementation was done on the testing stand illustrated in Figure 2. The following components were used:

- control system – Siemens Sinumerik 840D sl
- servo drives – Sinamics Double Motor Module 2x9A, Sinamics Double Motor Module 2x5A, Sinamics Double Motor Module 2x3A
- robotic arm – Comau Racer 7 1.4.

Figure 2. *Testing stand*

As already mentioned in the introduction, the robot will be directly controlled by the Siemens' motor modules. The robot's base has two connectors X1 and X2 – X1 is an encoder signal connector and X2 is motor and brake signals connector. The connection scheme is illustrated in Figure 3.

* Corresponding author: nikita.kuprin@fs.cvut.cz

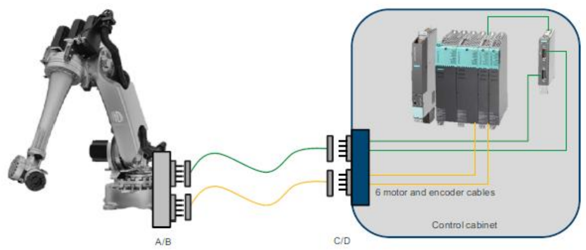


Figure 3. *Connection scheme* ***[1]***

3. Integration procedure

The integration process can be divided into the following steps:

1. network topology
2. robot configuration
3. modification of default PLC project
4. commissioning of PLC project
5. activation of robot configuration and calibration. [1]

3.1. Network topology

The network topology is essential for robot configuration. In this step, the *.utz file is created. There, the topology of DriveCliq network is defined – communication network between NCU, drive modules, encoder modules and any other component.

For this project, it was necessary to create topology which included the main control unit (NCU), 3 double motor modules and 3 SMC40 units. The SMC40 is responsible for connecting encoders to the DriveCliq network. The configuration was created with Sinumerik Create MyConfig (CMC) software and is illustrated in Figure 4.

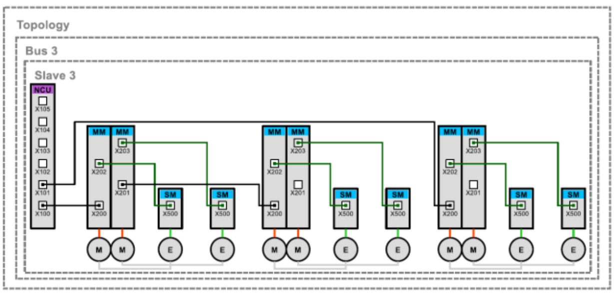


Figure 4. *DriveCliq Network topology*

Several conditions must be met when designing DriveCliq network:

- There cannot be connected more than 4 drives on a single branch.
- All the 6 axes must be connected to the same NCU/NX control unit.

Also, it is important to assign axes names to each drive as illustrated in Figure 5.

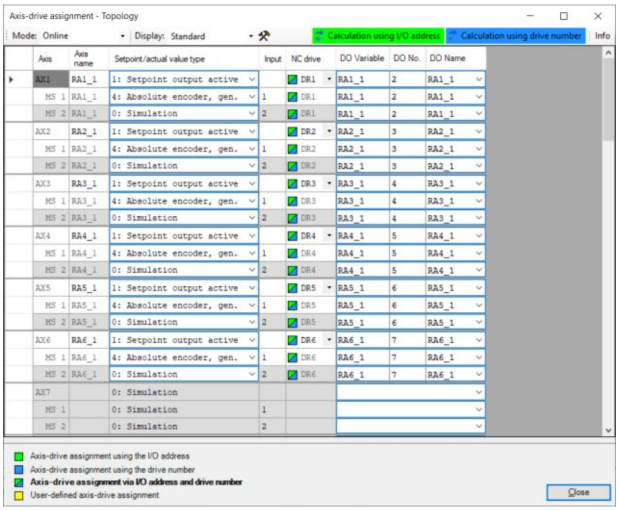


Figure 5. *Axes name assignment*

DO numbering starts at 2 because 1 is reserved for the NCU. Each axis is assigned a name RAx_1 where x stands for axis number.

The final network topology is exported to the *.utz file which will be used in the later steps of integration.

3.2. Robot configuration

There are two ways how to create robot configuration: either manually by setting up parameters in the control unit or using the Robot Configurator software. The later way was chosen for this project and is explained below.

The user interface of the Robot Configurator software is illustrated in Figure 6. Firstly, it is necessary to choose general parameters: mode, robot manufacture, number of robots and system units. For this project, all the parameters were left as default.

After, it is necessary to choose a robot from the database – Comau Racer-7 1.4 is used for the project.

Then, individual axes are assigned to the corresponding drive modules. The following default assignment was used:

- axes 1+2 – Double Motor Module 9A
- axes 3+4 – Double Motor Module 5A
- axes 5+6 – Double Motor Module 3A.

Kinematic structure of the Comau Racer 7-1.4 robot can be seen in Figure 7.

The next step is to choose orientation during programming. For robots, the convention is orientation A – Z, B – Y, C – X.

After, it is necessary to import the *.utz file with network topology (explained in the previous chapter). This is done in the tab labelled DRV.

Finally, the robot configuration can be exported. The output of this step is RobotInstallation folder and a CMC package (*.usz file). All the files must be exported to the root directory of USB flash drive that will be used to install the robot. Contents of the RobotInstallation folder are illustrated in Figure 8.

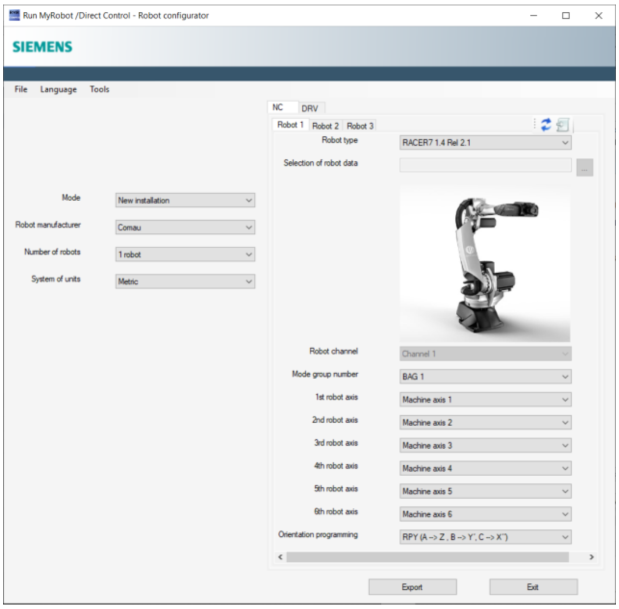


Figure 6. *Robot Configurator software*

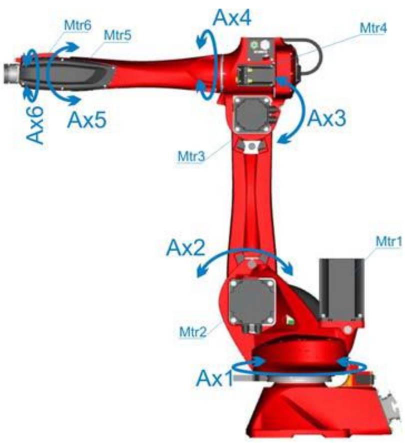
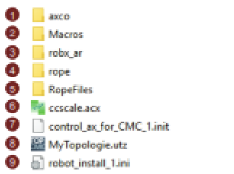


Figure 7. *Kinematic structure* **[2]**



① Optional: Folder "acxo"
If the compile cycle AXCO is needed for the selected robot, this folder is created. Copy the downloaded compile cycle ("*.elf") into this folder.

② Folder "Macros"
The files for configuring the drives are located in this folder.

③ Folder "robx_ar"
The compile cycle ROBX_AR is located in this folder.

④ Folder "rope"
The compile cycle ROPE is located in this folder.

⑤ Folder "RopeFiles"
The encrypted cycle ("*.cpf") located in this folder is relevant for data protection and is needed by the compile cycle ROPE.

⑥ File "ccscale.acx"
This file is needed for data protection.

⑦ "control_ax_for_CMC_1.init" file
This file is needed for setting the NC-side controller data.

⑧ Optional: File "MyTopologie.utz"
This file is needed for the DRIVE-CLiQ topology (user-specified topology). It is created using CMC-Topo.

⑨ File "robot_install_1"
This file contains the configuration that was created in the robot configurator. This file must **not** be edited or manipulated, because it controls the sequence of the CMC scripts! The number of robots determines the number of configuration files. A "robot_install_xxx.ini" file is created for each robot.

Figure 8. *Contents of RobotInstallation folder* **[1]**

3.3. Modification of PLC project

The sample project from Siemens served as the basis for the PLC project. For integration, it was necessary to set up the hardware configuration and to modify standard function blocks. The list of standard function blocks can be seen in Figure 10. The modifications were concerned with the system setup – MCP panel address, HMI panel settings, override step, alarms, etc.

Next, it was necessary to set up safety – reaction on the emergency stop button. For the testing purposes, the emergency stop button was connected in the following way:

- One channel is connected to the NCU port X122.2 – default input for a fast stop.
- The second channel is connected to the NCU port X142.3 – PLC input for emergency stop, available in DB10.DBX60.0.

Press of the emergency stop button must set the bit DB10.DBX56.1. This is realized by linking it to the bit DB10.DBX60.0 with normally closed contact as illustrated in Figure 9. As a result, the press of the emergency stop button triggers error 3000 which blocks any movement with the robot. Error acknowledgement is done by setting the bit DB10.DBX56.2. In this case, the bit is permanently set and for acknowledgement press of the RESET button is sufficient.

Figure 9. *Implementation of safety*

Block	Designation	Description
OB1	CYCL_EXC	Cyclic program processing
OB100	COMPLETE RESTART	Start-up routine
FB210	FbSelJogMode	Mode evaluation for JOG traversing in WCS, TCS, MCS and BCS
FC70	FcAlm	Controlling the ALM (optional)
FC218	FcOvrMcpHpu	Switch to MCP/HPU override handling
FC219	FcMcpHpu	Main block for MCP/HPU evaluation and switchover
FC220	FcOp	Operator panel signals
FC222	FcModeGroup	Evaluation of the reset for MCP/HPU
FC223	FcChannel	Channel-specific signals
FC224	FcUserIcon	Display of the user icons
FC230	FcAxTables	Axis assignment
FC231	FcAxes	Configuration of the active axes
FC232	FcAxis	Axis enable signals
FC1031	FcTeleg370	Data transfer interface DB/telegram 370
DB100	DbUserVers	PLC user program version
DB101	DbUserIcons	Display of the user icons
DB102	DbVar	Global project variables (non-retentive)
DB111	DbMcpHpu	Interface DB for MCP/HPU
DB210	DbInstSelJogMode	Instance DB of the FB210
DB1031	DbAlm	ALM interfaces DB

Figure 10. *Standard function blocks* **[1]**

3.4. Commissioning of PLC project

Firstly, it was necessary to commission the new PLC project. For that, the following steps were performed:

1. Reset PLC and NCK.
2. Compile PLC project and load to the NCU.
3. Complete commissioning by performing a warm reset of the NCK to synchronize the PLC and NCK.

The last step is important for establishing communication between the NCU and MCP.

3.5. Activation of the robot configuration

This step can be divided into two parts: execution of CMC script and manual steps.

For the execution of CMC script, it was necessary to plug in the flash drive with robot configuration (explained in 3.2) into the USB port X125 of the NCU and to perform cold start of the NCU. Next, CMC script was executed and the following tasks were performed automatically by the script:

- Setting the machine data – general, channel, axis and CC
- Copying the drive macros
- Copying and activating the compile cycles with associated machine data and alarms [1]

After successful execution of the CMC script, the following manual steps were performed:

- Starting up the infeed of SLM
- Starting the drive macros – it automatically carries out the commissioning of the drive and sets the optimized controller data for the respective robot axis
- Confirming the encoder serial number for all axes
- Determining the commutation angle for all axes
- Checking the current controllers – 0 dB line must not be exceeded and bandwidth must be greater than 600 Hz (optimal frequency response can be seen in Figure 11)
- Calibrating all the axes – the robot was brought to the adjustment positions using the original calibrated Comau control unit.

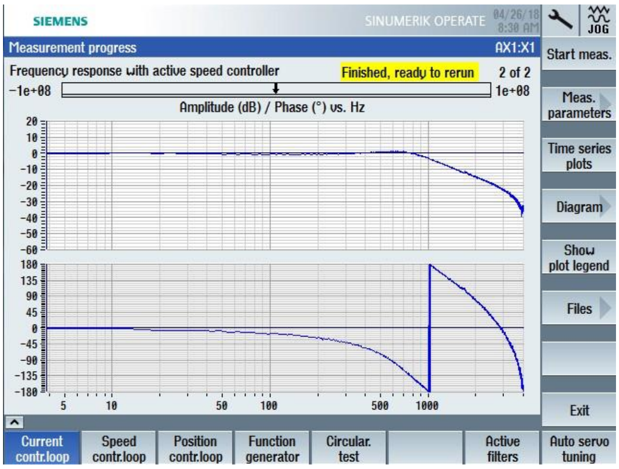


***Figure 11.** Optimal frequency response* **[1]**

4. Conclusion

The Comau Racer 7-1.4 robot was successfully integrated into the Siemens Sinumerik 840D sl control system. As a result, the robot's axes were seamlessly synchronized with the control system. It is now possible to use all the Sinumerik control system's functions to control the robot. For example, in the WC\s Jog mode, the robot can be navigated in the cartesian coordinate system.

Acknowledgement

This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS19/165/OHK2/3T/12.

Abbreviations

CC	Central Controller
CMC	Create MyConfigTopo software
DB	Data Block
DO	Digital Output
MCP	Machine Control Panel
NCK	Numerical Control Kernel
NCU	Numerical Control Unit
PLC	Programmable Logic Controller
SLM	Smart Line Module
SMC	Sensor Module Cabinet-Mounted

References

[1] AG, Siemens. *SINUMERIK Run MyRobot /Direct control* [online]. 2018 [cit. 2020]. Available from: https://support.industry.siemens.com/cs/ww/en/view/109758486

[2] COMAU. *Racer7 - 1.4 V2.1 Technical Specifications*. CR00758092-en_00/2016.07. 2016.