2 Basic Motion Concepts

Chapter Topics:

- Structure of Rockwell and Siemens motion control programs
- Basic configuration steps
- Network connection to servo drives
- Installation considerations

OBJECTIVES

Upon completion of this chapter, you will be able to understand:

- Organization of axes and drives in Rockwell and Siemens motion control programs
- Basic steps to start a motion project
- Communication network connection and configuration
- Installation best practices

SCENARIOS

Scenario 1: Rotary knife control fault

A rotary knife application (Figure 2.1) has two servo axes, one for the conveyor and one for the knife. The processor is a Siemens S7-1515T-2 PN with a 30 ns bit operation time. The knife motion is a position cam with the conveyor as the leading axis. The Send Clock is 1 millisecond, which is the default value. The processor stops approximately one second after the cam starts. The processor fault indicator is flashing. The diagnostics buffer contains the following three errors:

> Temporary CPU error: Acceptable number of pending OB 92 events exceeded 4 times
>
> Temporary CPU error: Buffer overflow for OB 91 events
>
> Stop cycle time exceeded through workload. MotionControl OB execution stopped, new startup inhibit set

which indicate the Send Clock value is too small.

Solution: The performance of the motion control is slowed by increasing the Send Clock to 2 milliseconds. The system does not stop with this Send Clock value. The profiling library

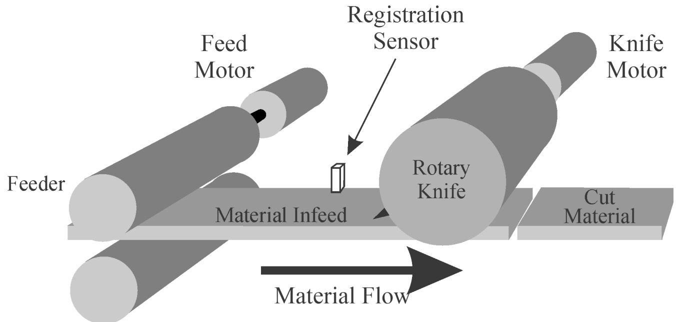


Figure 2.1. Rotary knife system.

function blocks (Siemens, 2025c) are employed to analyze the processor performance. The execution time of the MC-Servo block (OB 91, the core of the motion control) ranges from 290 to 590 microseconds. The MC-Servo block execution is synchronized to the Send Clock. So, with the default Send Clock of 1 millisecond, the execution of the motion control sometimes exceeds half of the processor workload, leading to the excessive number of pending OB 92 events and eventually an OB 91 buffer overflow.

To prevent future errors, the performance of the motion control is further slowed:

Increased the Send Clock to 4 milliseconds (the maximum)

Set the MC-Servo Cycle time factor to 2, which sets the MC-Servo application cycle to 8 milliseconds

The default values for the Send Clock and MC-Servo application time are really for the high-end processors which execute at least 10 times faster than the particular processor used for this application.

Scenario 2: The label-maker for a tire-making machine was randomly printing extra labels.

Solution: Since it was a random occurrence, the problem is suspected of being related to electrical noise. The bonding and grounding of the panels is checked and no issues are discovered. The routing of the conductors in the dirty/very-dirty/clean zones is checked and no issues are found. The shields of the encoder cables and motor cables are checked. For one of the drives, the motor cable shield was not properly secured. The motor power cable connector for the particular drive was on top of the drive and was not easily visible since the drive was mounted above normal line-of-sight. The shield was secured in the clamp attached to the drive. A 0.1μF capacitor is also placed across the 24-volt power supply output. The label-maker stopped randomly printing extra labels.

2.1 INTRODUCTION

Before starting on the programming aspects of a motion control system, one must understand the fundamental concepts of a project:

- Organization
- Networking
- Installation

Rockwell and Siemens projects have different methods of representing the axes and drives in a motion system. The general process of creating the axes and adding drives to the project is also outlined. The network connection from the processor to the servo drives is described next. The aspects of the communication protocol relevant to the high-speed synchronized operation required of motion control is also covered. Lastly, installation considerations such as bonding, grounding, panel layout, and electrical noise reduction are covered.

2.2 ROCKWELL MOTION CONTROL STRUCTURE

For motion control with a Rockwell ControlLogix or CompactLogix system, one must have an appropriate processor, communication module, and servo drive. Current servo drives only support Ethernet/IP communications and that is the priority of this text. Older servo drives used the SERCOS network for communication, and these are briefly described. Some processors have embedded Ethernet ports and do not need a separate communication module to communicate to the servo drives.

A project involving servo motion consists of one or more *axes* and one or more *drives*. An axis is the element of a motion control system that manifests motion and the processor program interacts with an axis. The processor communicates with the drive that in turn, interacts with the servo motor that does the actual motion. The axis is basically a tag consisting of configuration that defines how an axis is to be operated. The valid types of axes are:

AXIS_CIP_DRIVE – Kinetix drives on Ethernet/IP

AXIS_CONSUMED – link for axis motion data produced by a motion axis on another Logix controller. Supported by CompactLogix 5380, CompactLogix 5480, and Compact GuardLogix 5580

AXIS_GENERIC – ControlLogix 5570 and GuardLogix 5570 only. Represents an axis with full motion planner functionality, but not associated with any device that requires motion configuration support (for example, the 1756-DM).

AXIS_GENERIC_DRIVE – SERCOS non-Rockwell drive

AXIS_SERVO – older analog servo modules (for example, 1756-M02AE). Axis execution is done in a separate module and not in the processor

AXIS_SERVO_DRIVE – SERCOS drives

AXIS_VIRTUAL – virtual axis

Within the processor, the motion control axes are grouped in a Motion Group, shown conceptually in Figure 2.2*a* and in the project tree as part of a motion group in Figure 2.2*b*. Currently, Rockwell processors support only one motion group. The processor may also contain other axes not part of the motion group. These axes are shown in the project tree in the “Ungrouped Axes” folder.

The axes in the motion group (that are not virtual axes) conceptually communicate with a drive, Figure 2.2*a*. In reality, the processor motion task does the communicating with the drive using the axis configuration data. The drives are shown in the project tree as part of the I/O Configuration (Figure 2.2*b*), which also shows the communication network path to the drives. Figure 2.2*b* shows the drives connected to an Ethernet module in the chassis. The particular processor has an embedded Ethernet port and the drives could be connected to that Ethernet port.

The major steps to the initial configuration of a motion system are shown in Figure 2.3. The steps are described as:

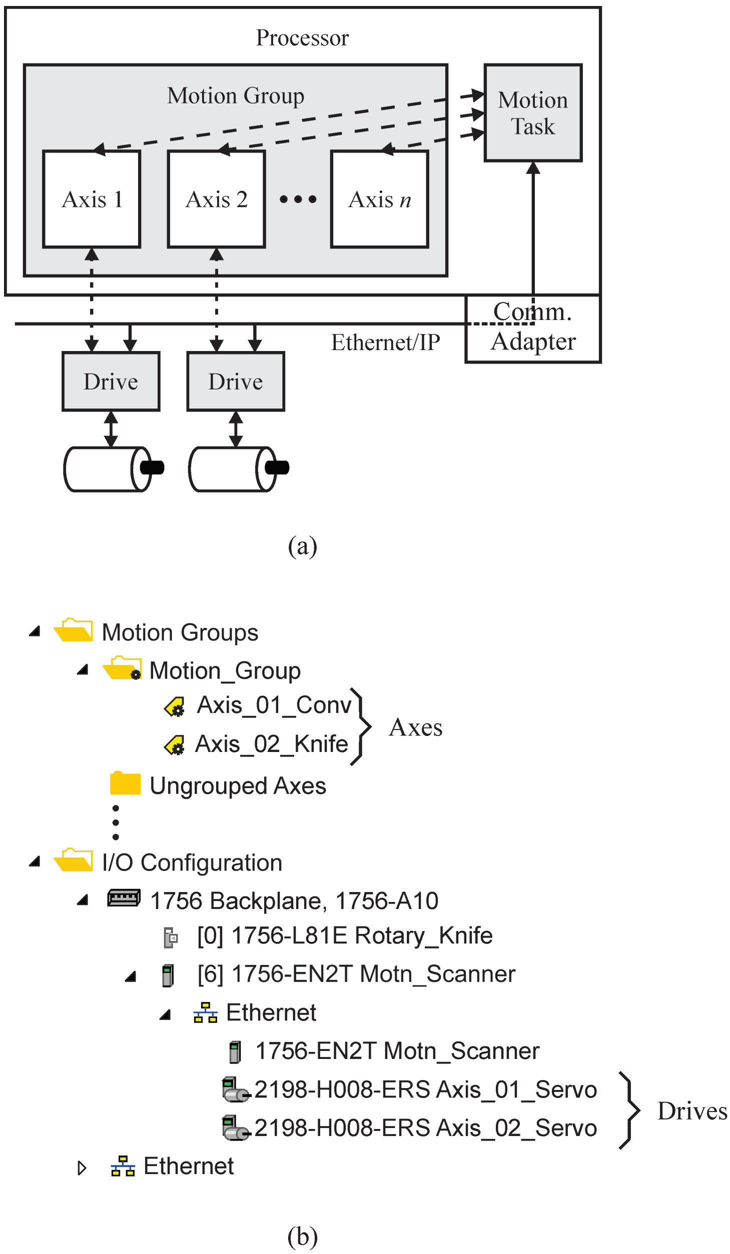


Figure 2.2. Rockwell motion control model: *(a)* conceptually; *(b)* as project tree.

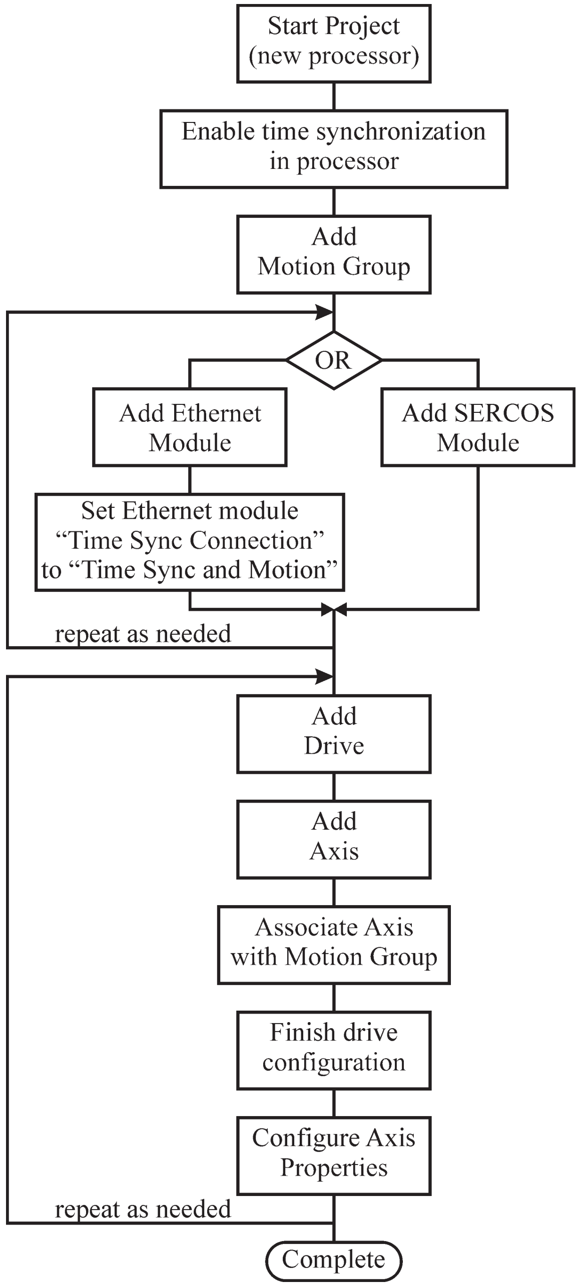


Figure 2.3. Rockwell motion control project overall steps.

Start project – Select a processor capable of motion.

Enable time synchronization in processor – This parameter is part of the processor configuration.

Add Motion Group – Current processors only support one motion group. One must also set up the Base Update Period (BUP), which is the period of the motion task. Up to two additional slower motion periods may also be configured.

Add Ethernet module (as needed) – The Ethernet module must be capable of handling CIP motion and is added to the I/O configuration. Set the IP address and other network parameters.

Set Ethernet module "Time Sync Connection" to "Time Sync and Motion"

Add SERCOS module (as needed) – Kinetix 2000, Kinetix 6000, Kinetix 6200, and Kinetix 7000 drives use the SERCOS network

Add drive – Add drive to the I/O configuration.

Add axis – Axis associated with drive. Done as part of the initial drive configuration

Associate axis with Motion Group – The axis is added to the existing motion group. Done as part of the motion group configuration.

Finish drive configuration –

- Configure the general settings, including the network IP address and mask.
- Configure power settings such as input voltage, input phasing, and bus configuration
- Configure drive digital inputs and outputs
- Configure safety connections, if the drive has integrated safety

Configure axis properties –

- General category – Type of application and response speed
- Motor – If Rockwell motor, use the catalog number. Otherwise enter motor nameplate data such as power, rated voltage, rated current, and maximum speed.
- Motor Feedback – If Rockwell motor, determined by catalog number. Otherwise, enter the encoder characteristics
- Scaling – Scaling to convert motion from controller units to user-defined units. Also sets load type, actuator type, and travel limits.
- Polarity – Motor, motion, and feedback polarity
- Load – Load coupling, load ratio, backlash and friction compensation. Also configures the load observer if it exists in the drive.
- Position Loop – Position loop tuning and error limits
- Velocity Loop – Velocity loop tuning and error limits
- Acceleration Loop – Acceleration limits
- Torque/Current Loop – Gains and limits
- Planner – Motion planner characteristics: maximum velocity, acceleration, and jerk.
- Homing – active/passive homing. If active homing, set sequence.
- Actions – Action for drive stop, motor overload, and inverter overload
- Exceptions – Action for exception (fault) conditions i.e. faults.
- Cyclic Parameters – Parameters to be read and written every cycle.

2.3 SIEMENS MOTION CONTROL STRUCTURE

For motion control with a Siemens S7-1500 system, one must have an appropriate processor and servo drives. The processor has an embedded PROFINET port.

A project involving servo motion consists of one or more *technology objects* and one or more *drives*. A technology object represents a real object, such as an axis. The technology object data block corresponds to the properties of the real object. The motion control blocks in the user program invoke the functions of the technology objects. These functions are executed in the motion control organization blocks (MC-Servo, MC-Interpolator, plus others) that are independent of the user program. The technology objects control the motions of the real objects and report back status information. Through the organization blocks, the processor communicates with the drive that in turn, interacts with the servo motor that does the actual motion.

Possible technology objects (with the data type), are:

Speed axis (TO_SpeedAxis) – All motions are speed-controlled motions.

Positioning axis (TO_PositioningAxis) – Control the position of an axis with closed-loop control. Speed control is also possible.

Synchronous axis (TO_SynchronousAxis) – A positioning axis that can be interconnected with a leading value such that this axis, as the following axis, follows the position of the leading axis during a synchronous operation, such as gearing and camming.

External encoder (TO_ExternalEncoder) – Detects position and makes it available to the controller. Used for an encoder that is not integrated in the servo motor.

Measuring input (TO_MeasuringInput) – Accurate, event-dependent detection of the actual position of an axis or external encoder in response to a signal change at the measuring input.

Output cam (TO_OutputCam) – Generates discrete switching signals dependent on the position of an axis or external encoder.

Cam track (TO_CamTrack) – Generates a discrete switching signal sequence dependent on the position of an axis or external encoder. A cam track can consist of up to 32 individual output cams.

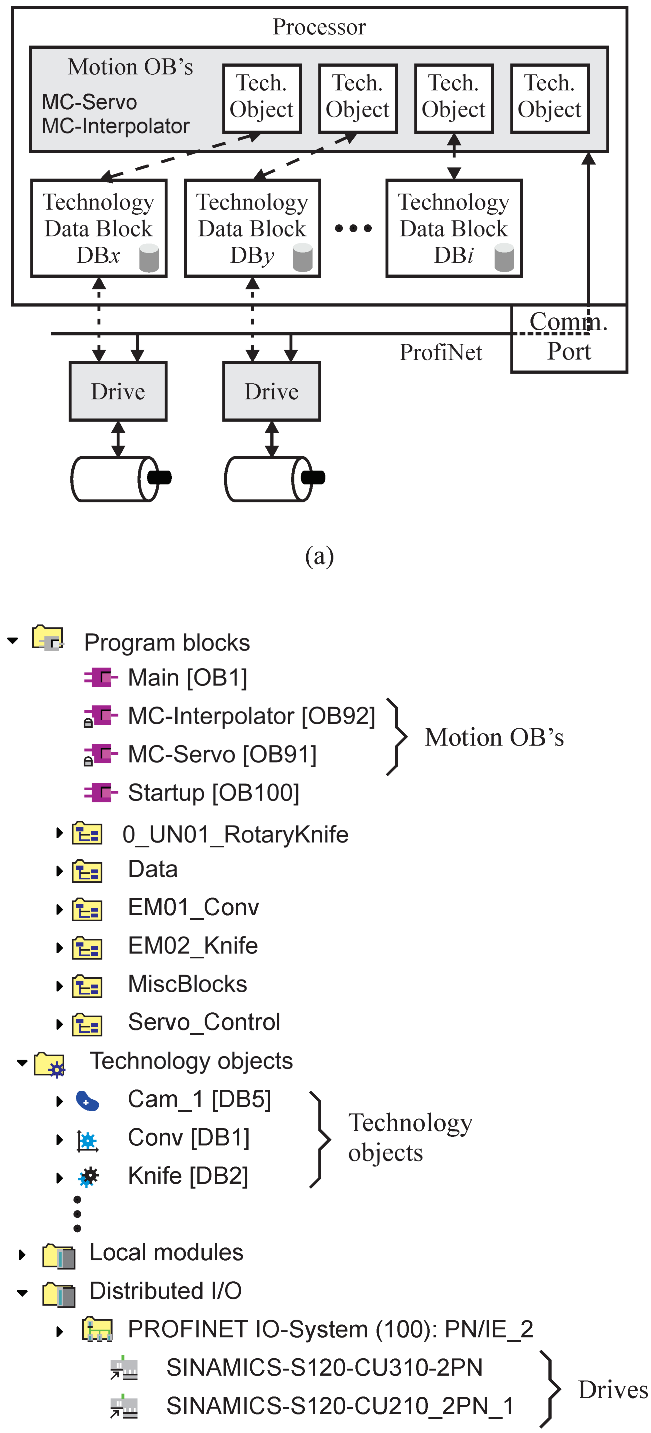
Cam (TO_Cam, TO_Cam_10K, TO_Cam_600Seg, or TO_Cam_6kSeg) – Defines the dependency of a following axis on the position of a leading axis as a function. The function is defined as data points and/or segments. The types are further defined in Section 4.6.

Leading axis proxy (TO_LeadingAxisProxy) – When the leading axis for a synchronous operation is in another processor, this object receives the telegram from the other processor and provides the leading value for the following axis in this processor.

Kinematics (TO_Kinematics) – Provides the framework for a mechanical system consisting set of interconnected axes, for example, a robot arm. Motion commands to the kinematics system are executed on the individual axes in the system.

Interpreter (TO_Intepreter) – Processes commands from the interpreter program and sends motion commands to the technology objects.

Interpreter program (TO_InterpreterProgram) – Program that can be loaded into and executed by the Interpreter.



(b)

Figure 2.4. Siemens motion control model: *(a)* conceptually; *(b)* as project tree..

Interpreter mapping (TO_InterpreterMapping) – Defines which technology objects and processor tags that an Interpreter program can access.

The interpreter technology objects are not covered by this text.

Within the processor, the technology objects are shown conceptually in Figure 2.4*a* and in the project tree as part of the "Technology objects" folder in Figure 2.4*b*. The example in Figure 2.4b has a positioning axis (Conv), a synchronous axis (Knife) and a cam profile (Cam_1).

The technology data blocks (that are not virtual axes) conceptually communicate with a drive, Figure 2.4*a*. In reality, the processor motion organization blocks communicate with the drive using the technology data block configuration data. The drives are shown in the project tree as part of the Distributed I/O (Figure 2.4*b*). A pictorial representation of the network configuration is shown as part of the "Network view" in the "Device configuration."

The major steps to the initial configuration of a motion system are shown in Figure 2.5. The steps are described as:

- Start project – Select a processor capable of motion.
- Add drive – major steps are:
  - Add drive to the distributed I/O configuration.
  - Set initial network connection to processor
  - Add DO_SERVO module object to drive
  - Add additional telegram objects as needed. At a minimum, one will need Standard Telegram 5
  - Set up IP address and PROFINET device name
  - Enable isochronous operation
- Complete network interconnections – partner port settings between the processor and the drives. An example interconnection is described later in the chapter.
- Set isochronous mode - Configure processor as PROFINET for isochronous mode and as sync master, set send clock period and communications load
- Add axis technology object – major steps are:
  - Configure basic parameters – real or virtual axis, linear or rotary axis, units of measure (for position, velocity, and torque), and whether simulation is activated.
  - Specify drive for the axis – select drive from the available drives in the I/O
  - Configure encoder(s) – encoder can be part of the motor, or an external encoder. Up to four encoders can be defined.
  - Configure data exchange with drive and encoder(s).
  - Leading value interconnections – synchronization between axes in processor.
  - Leading value settings – synchronization between axes in different processors.
  - Configure mechanics – encoder mounting (on motor shaft, load side, or external), gearing, and drive direction inversion.
  - Dynamic default values – default values of the velocity, acceleration, deceleration and jerk motion block inputs.
  - Emergency stop – maximum velocity and deceleration if axis is emergency stopped.

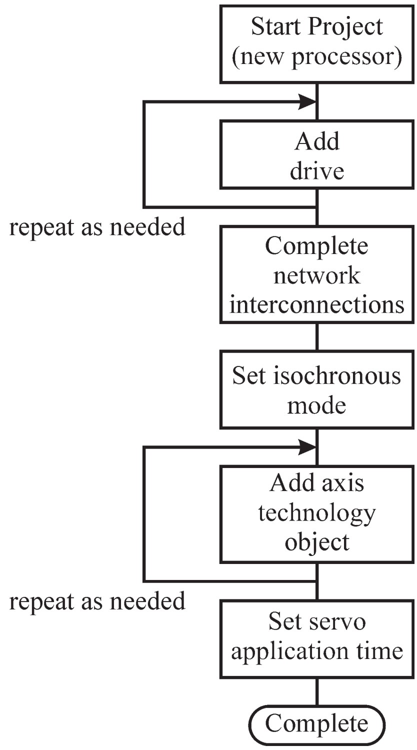


Figure 2.5. Siemens motion control project overall steps.

Limits – hardware and software position limits, maximum velocity, maximum acceleration, maximum deceleration. Torque limits can be set, but only if the drive is configured for Telegram 10x.

Homing – active/passive homing. If active homing, set sequence.

Position Monitoring – sets the time and position window that determines when in position. Also set the time and velocity window that determines if the axis is not moving. If following error monitoring is enabled, an error can be generated if the following error is above a threshold and a warning can be generated if the following error is a percentage of the threshold. The threshold can be dynamically set, that is, the value can be changed to a larger value for larger velocities.

Control loop – Set gain of position loop and precontrol of velocity loop. Position control can be accomplished in the drive or in the processor.

Actual value extrapolation – Set parameters for extrapolation of position and velocity values to compensate for communication network delays.

Set servo application time – May need to set the servo application time to a multiple of the Send Clock (part of MC-Servo OB properties).

Finish drive configuration

Drive configuration is accomplished with the StartDrive or Starter software packages.

2.4 NETWORK CONNECTION TO PLC

The connection between the PLC processor and the servo drives is via a communication network. This section covers three communication networks: Ethernet/IP, SERCOS, and PROFINET. The new IEEE time-sensitive network (TSN) protocol (IEEE, 2020) will likely be adopted by Rockwell Automation and Siemens in the future.

This section is focused on the communications between the processor and the drives. For information about a comprehensive network for the manufacturing enterprise, the interested reader is referred to Cisco and Rockwell Automation (2011) and Siemens (2025a).

2.4.1 Ethernet/IP

Ethernet/IP is the communication protocol used by current Rockwell servo drives and variable-speed drives. Ethernet/IP uses standard Ethernet and TCP/IP (transport control protocol/Internet protocol) protocols to transport messages. The Common Industrial Protocol (CIP) is the messaging protocol at the application layer in Ethernet/IP.

Real-time performance on Ethernet/IP is accomplished with an update mechanism called CIP Sync (Cisco, Panduit and Rockwell Automation, 2019) and has the following characteristics:

Each drive has its own internal clock, accurate to the nanosecond level

The internal drive clocks are synchronized and tuned to a master clock (typically the processor) over the network once per second.

CIP Sync is part of the IEEE 1588 PTP standard (IEEE, 2019) for time synchronization and accounts for real-time latencies in the network and master clock changes.

Motion control needs real-time synchronization to the network, but a managed Ethernet switch with Quality of Service (QoS) and CIP Sync on each machine is not always needed when there are no more than 4 drives. However, when a motion system links multiple automation cells or machines together, a managed switch with QoS and CIP Sync must be used. Detailed guidelines are in the ‘CIP Sync’ and ‘CIP Sync Sequence of Events’ chapters of Cisco and Rockwell Automation (2011).

General guidelines (Rockwell, 2022a) for the Ethernet/IP connections between the processor and the servo drives are shown in Figure 2.6 for drives with dual-port Ethernet connections and shown in Figure 2.7 for drives with single-port Ethernet connections. These diagrams show a separate Ethernet communication module with one or two ports. On some processors, the Ethernet ports are embedded in the processor and separate communication modules may not be used.

The preferred connection is a direct linear connection with the drives daisy-chained together as shown in Figure 2.6*a*. If there are other remote devices (for example, distributed input/output modules), then a switch must be used, as shown in Figure 2.6*b*. The switch should be a managed switch that includes CIP Sync/PTP (for example, Stratix 5200). If the system has an unmanaged switch, then it should be connected to the last drive in the daisy-chain shown in Figure 2.6*a*. An alternative to the daisy-chain of Figure 2.6*a* is the device-level ring (DLR) configuration shown in Figure 2.6*c*. The communication module in the PLC must be capable of this configuration.

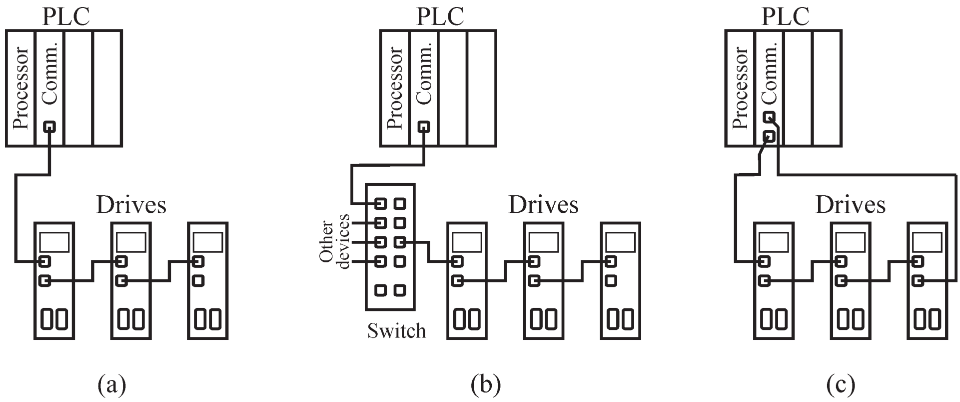


Figure 2.6. Rockwell Ethernet/IP recommended networks for drives with dual-port Ethernet: *(a)* direct linear; *(b)* with switch; *(c)* device-level ring (DLR).

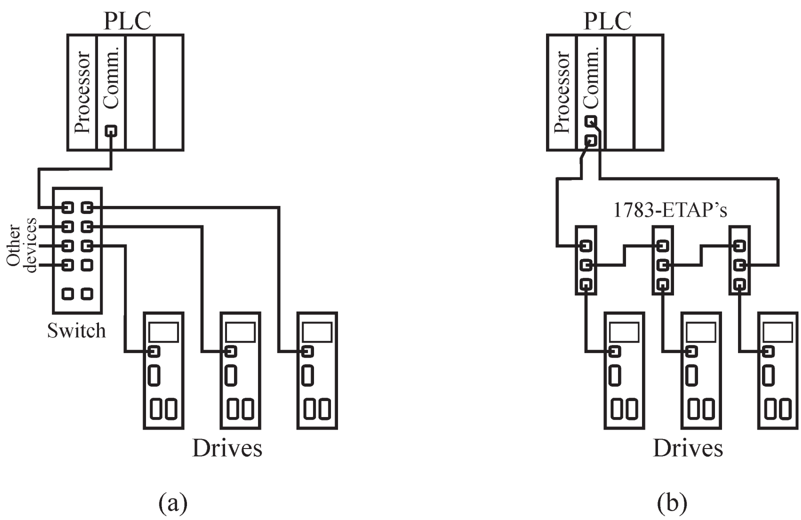


Figure 2.7. Rockwell Ethernet/IP recommended networks for drives with single-port Ethernet: *(a)* with switch; *(b)* device-level ring (DLR)..

For drives that have a single-port Ethernet connection, a switch must be used, as shown in Figure 2.7*a*. If there are no more than four drives, then the switch can be an unmanaged switch (for example, Stratix 2000). For five to eight axes, one should use a managed switch such as a Stratix 5700. If there are more than eight axes, a full managed switch with CIP Sync, such as a Stratix 5700 must be utilized. If a DLR configuration is desired, then each drive is connected to a 1783-ETAP module and the ETAP modules form the ring (Figure 2.7*b*).

One important aspect of the Ethernet/IP network is the configuration of the periodic communication between the processor and the drives. This configuration is part of the Motion Group properties.

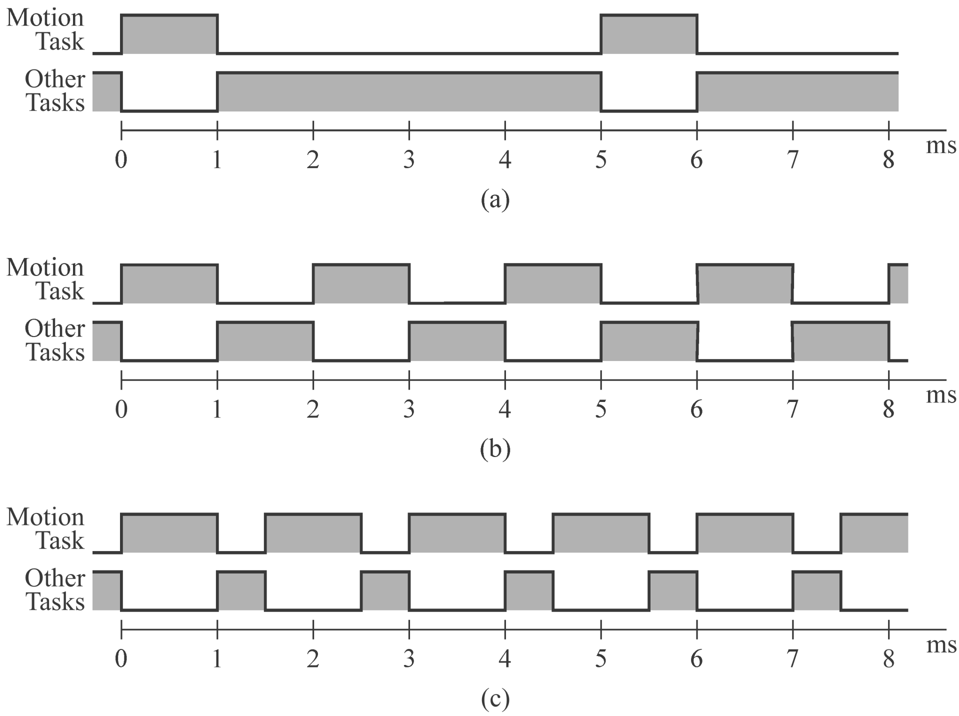


Figure 2.8. Base Update Period (BUP) examples when Motion Task scan time is 1 ms: (a) BUP = 5 ms; (b) BUP = 2 ms; (c) BUP = 1.5 ms.

The Base Update Period (BUP) is the request packet interval (RPI) for the Ethernet communication between the processor and the drive, a Unicast connection, and also the period at which the Motion Task runs. The Motion Task runs at a higher priority than most other tasks. The BUP is a trade-off between updating the axis positions and scanning the other tasks. A low BUP means more dynamic axis position control, but uses a larger percentage of the processor scan. A system with more axes will have a longer Motion Task scan time. The maximum scan time of the Motion Task should be less than 50% of the BUP.

For example, if the scan time of the Motion Task is 1.0 ms, setting the BUP to a value less than 2.0 ms is not recommended. Figure 2.8 shows the amount of time available for the other controller tasks. When the BUP is 5.0 ms, the Motion Task uses about 25% of the controller execution time. When the BUP is 2.0 ms, the Motion Task requires 50% of the controller execution time. When the BUP is 1.5, the Motion Task uses about 67% of the controller execution time, leaving little time for the other tasks.

The Rockwell recommended minimum BUP per axis when using a Kinetix CIP drive (Kinetix 350, 5300, 5500, 5700, or 6500; PowerFlex 527 or 755) are as follows (Rockwell Automation, 2025a):

Processor	**BUP per axis**
1769-L1x, 1769-L2x, 1769-L3x	0.75 ms
1756-L7x	0.5 ms
5069-L3xxERM	0.03125 ms
1756-L8xE (embedded Enet port)	0.03125 ms
1756-L9xx (embedded Enet port)	0.02 ms

The minimum BUP is 1.0 ms. For example, if using a 1756-L81E processor and 50 drives, the BUP should be at least 1.56 ms, which would be changed to 2.0 ms, since one can only specify the BUP in increments of 0.5 ms. As a second example, if using a 1756-L9xx processor with 50 Kinetix drive axes, the BUP would be 1.0 ms, the minimum.

However, if using PowerFlex drives, the minimum BUP is greater. For a PowerFlex 527 drive, the minimum BUP is 4 ms and for a PowerFlex 755 drive, the minimum BUP is 3 ms.

ControlLogix controllers also provide two alternative update periods, slower than the BUP and an integer multiple of the BUP. One should use the BUP for the axes that require dynamic positioning accuracy such as high-speed coordinated axis positioning, aggressive position cams, and registration position correction. Axes that do not require a fast rate use one of the alternative update periods. Applications such as non-coordinated motion, slow point-to-point moves, and less aggressive position cams fall into this category. The alternate update periods also allow one controller to control a larger number of axes.

The general communication timing is described in Rockwell Automation (2024). There are two models:

One Cycle – for ControlLogix 1756-L7x and CompactLogix 1769-L1x/L2x/L3x and earlier processors

Two Cycle – for ControlLogix 1756-L8x and CompactLogix 5069 and later processors

The Two Cycle Timing doubles the number of maximum axes that a processor can handle.

In **One Cycle Timing** the entire exchange of drive-to-controller (D-to-C) packet, motion task execution, and controller-to-drive (C-to-D) packet occurs within one controller base update period (BUP). The C-to-D connection packets are sent periodically according to the configured BUP. The drive scan time, the period at which the device performs its control calculations, is typically much faster than the BUP (0.125 ms for most CIP drives). The timing of basic motion on the EtherNet/IP network with one-cycle timing is shown in Figure 2.9. The steps are as follows:

1. Drive scan runs, triggered by start of BUP.
2. During scan, device initiates D-to-C message with actual position.
3. Processor receives D-to-C message.
4. Motion Task runs, calculating a command position.
5. Processor initiates C-to-D message with command position.
6. Drive receives C-to-D message before end of BUP.

The Motion Task is started 330 μs after the start of the BUP, allowing little time for the D-to-C message to be received before the start of the Motion Task. There is also relatively little time for the C-to-D message to be received by the drive. The timing is relatively “tight” for small BUPs.

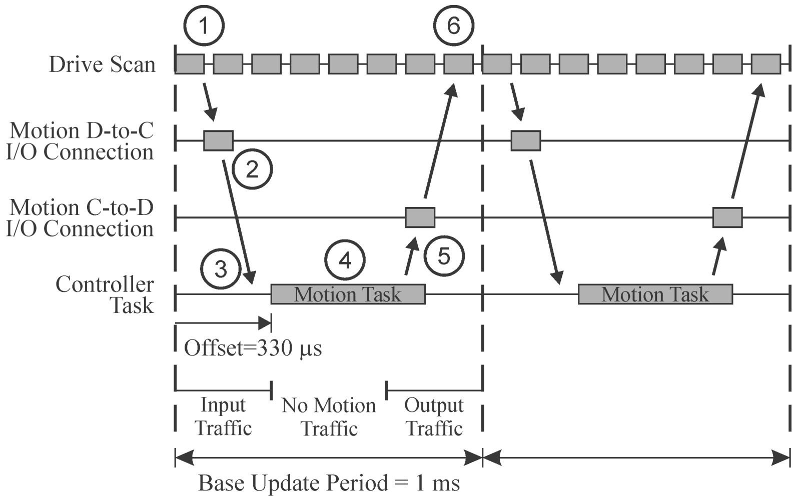


Figure 2.9. Rockwell Ethernet/IP one cycle timing model**.**

In **Two Cycle Timing** the exchange of drive-to-controller (D-to-C) packet, motion task execution, and controller-to-drive (C-to-D) packet occurs within two controller update periods (BUP). The C-to-D connection packets are sent periodically according to the configured BUP. The Motion Task performs its calculation, but does not send the command until the next BUP scan. The timing of basic motion on the EtherNet/IP network with two-cycle timing is shown in Figure 2.10. The steps are as follows:

1. Drive scan runs, triggered by start of BUP.
2. During scan, device initiates D-to-C message with actual position.
3. Processor receives D-to-C message.
4. Motion Task runs, calculating a command position. The dashed line shows that it is sent to the drive in the next cycle.
5. On the next BUP, at the start of the Motion Task, processor initiates C-to-D message with command position, which was calculated during the previous scan.
6. Drive receives C-to-D message before end of BUP.

In this scan model, the Motion Task is started later in the BUP, allowing increased time to receive the D-to-C message. Also, since the C-to-D message is initiated on the next BUP cycle, more time is allowed to receive the C-to-D message. Note that the motion task in the first cycle generates a C-to-D message, but it uses the motion task control actions from the previous cycle. The D-to-C message generated by the drive on the second cycle will be used by the next motion task execution.

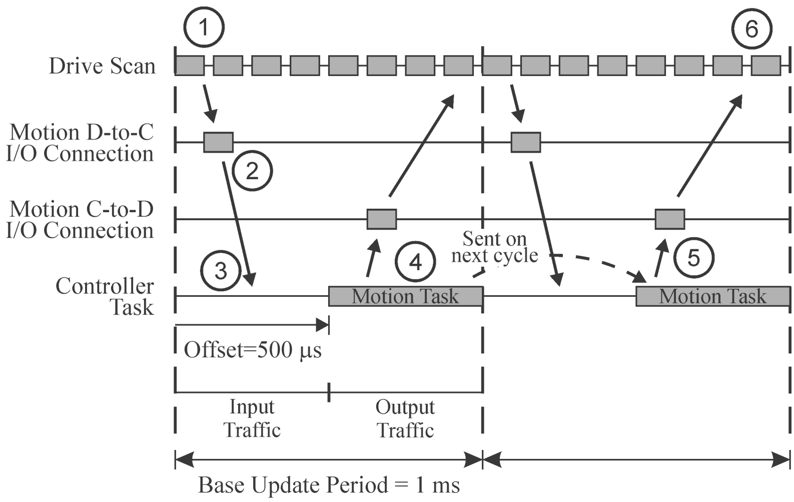


Figure 2.10. Rockwell Ethernet/IP two cycle timing model.

2.4.2 SERCOS

The IEC 61491 SErial Real-time COmmunication System (SERCOS) is the communication protocol used by the first-generation Rockwell servo drives (Kinetix 6000, 6200, 7000). The original standard IEC 61491 has been combined with other communication standards and is profile CPF 16 in IEC 61784 (IEC, 2013, 2023). An example network connection to five drives is shown in Figure 2.11. The communication adapter is a 1756-M*xx*SE Sercos interface module. The network is a fiber-optic ring with one path around the ring.

As for the Ethenet/IP network, one must configure the periodic communication between the processor and the drives. This configuration is part of the Motion Group properties and is identical to that of the Ethernet/IP drives.

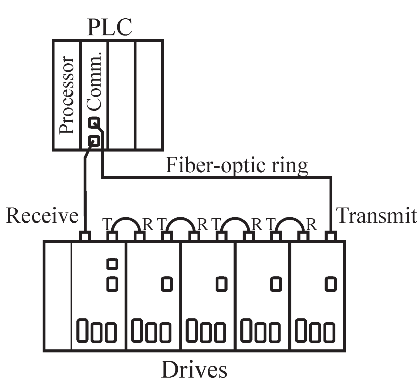


Figure 2.11. Rockwell Sercos network connections.

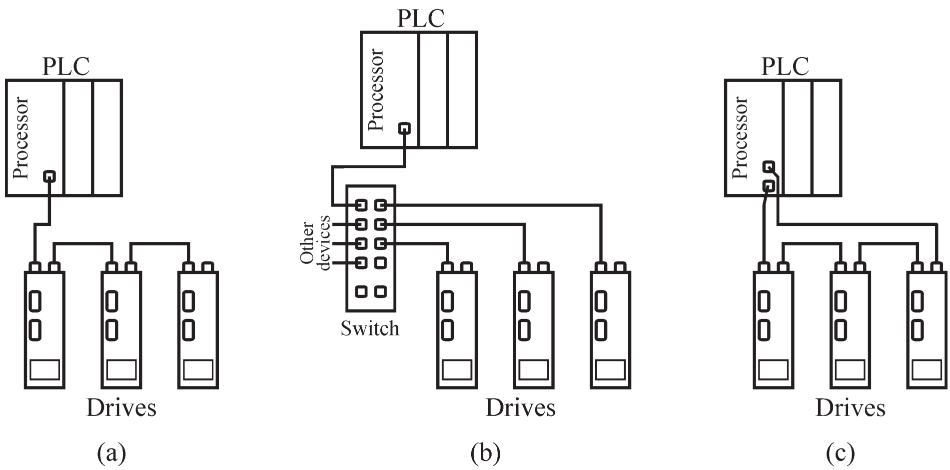


Figure 2.12. Siemens PROFINET recommended networks: *(a)* linear; *(b)* star; *(c)* ring.

2.4.3 PROFINET

PROFINET is the communication protocol used by Siemens S7-1500 processors with current servo drives and variable-speed drives. Like Ethernet/IP, PROFINET uses standard Ethernet and TCP/IP protocols to transport messages. At the application layer, functions are added to support the needs of motion control. For real-time synchronization, the isochronous real-time (IRT) mode provides prioritized, deterministic data transmission. PROFINET also defines a real-time (RT) mode that does not provide synchronization. The RT mode can be used for a low-performance motion control system, but IRT is assumed by this text.

General guidelines (Siemens, 2025a) for the PROFINET connections between the processor and the servo drives are shown in Figure 2.12 for drives with dual-port Ethernet connections and shown in Figure 2.12. All S7-1500 processors currently available for motion control have an X1 interface which is a dual-port PROFINET interface that supports IRT with isochronous mode, so a separate communication module is not needed.

The preferred connection is a linear connection with the drives daisy-chained together as shown in Figure 2.12*a*. A star configuration with a switch may be used, as shown in Figure 2.12*b*. The switch should must be IRT-compatible. An alternative to linear connection of Figure 2.12*a* is the MRP ring configuration shown in Figure 2.12*c*. This topology provides redundancy, but requires extra configuration steps for the processor X1 interface.

One different aspect of configuring the communications between the processor and the drives is setting up the port connections for the processor and the drives. If the network connection is a star, then the connections between the switch ports and the drives must also be configured. As an example, Figure 2.13 shows the connections between a processor and four SINAMICS S120 CU310-2 PN drives. Figure 2.13*a* shows the network view of the configuration which basically shows which network connects the devices. The PN/IE_1 network is for programming and the HMI. The PN/IE_2 network is for the drives. The actual physical port connections between the processor and the drives are shown in Figure 2.13*b* and the device port configuration parameters are as follows:

Processor (Name: Robot)
Port Partner port
[X1 P1 R] J1_Axis\PN-IO [X150]\Port 1 [X1 P1 R]

Drive 1 (Name: J1_Axis)
Port Partner port
[X1 P1 R] Robot\PN-IO [X1]\Port 1 [X1 P1 R]
[X1 P2 R] J2_Axis\PN-IO [X150]\Port 1 [X1 P1 R]

Drive 2 (Name: J2_Axis)
Port Partner port
[X1 P1 R] J1_Axis\PN-IO [X150]\Port 2 [X1 P2 R]
[X1 P2 R] J3_Axis\PN-IO [X150]\Port 1 [X1 P1 R]

Drive 3 (Name: J3_Axis
[X1 P1 R] J2_Axis\PN-IO [X150]\Port 2 [X1 P2 R]
[X1 P2 R] J4_Axis\PN-IO [X150]\Port 1 [X1 P1 R]

Drive 4 (Name: J4_Axis
[X1 P1 R] J3_Axis\PN-IO [X150]\Port 2 [X1 P2 R]
[X1 P2 R] not activated

One important aspect of the IRT mode is the configuration of the periodic communication between the processor and the drives. This configuration is part of the processor configuration. The processor must have the "Sync master" synchronization role and both RT and IRT must be enabled for the X1 interface. The "Send Clock" period defines the time between two consecutive communication cycles, and its value is between 0.25 and 4.0 milliseconds. The runtime of the MC-Servo OB must never be longer than the Send Clock of the isochronous PROFINET IO system. To determine an appropriate Send Clock value, set it to the maximum value (4.0 ms) and then use the function blocks in the profiling library (Siemens, 2025c) while running the motion controller program to determine the execution time of the MC-Servo OB. Set the Send Clock to a value greater than this execution time, allowing time for communications and the execution for the main OB and any other OB's. If the Send Clock value is already at the maximum value, then a faster processor must be purchased.

For applications with coordinated (kinematics) motion and/or position cams, the cycle time of the OB_Servo task may need to be increased to some multiple of the Send Clock (called the Send Clock Factor property of OB-Servo). This factor does not mitigate the requirement that the MC-Servo execution time be less than the Send Clock value, but allows more time for the execution of the multi-axis coordination and camming parts of the PLC program. The scenario at the beginning of the chapter illustrates that point. Note that the Send Clock Factor also influences the axis control performance. A Send Clock Factor that is set too high may not have satisfactory performance.

The drive-to-controller (D-to-C) telegram and controller-to-drive (C-to-D) telegram are both scheduled at the start of a data cycle. The drive scan time, the period at which the

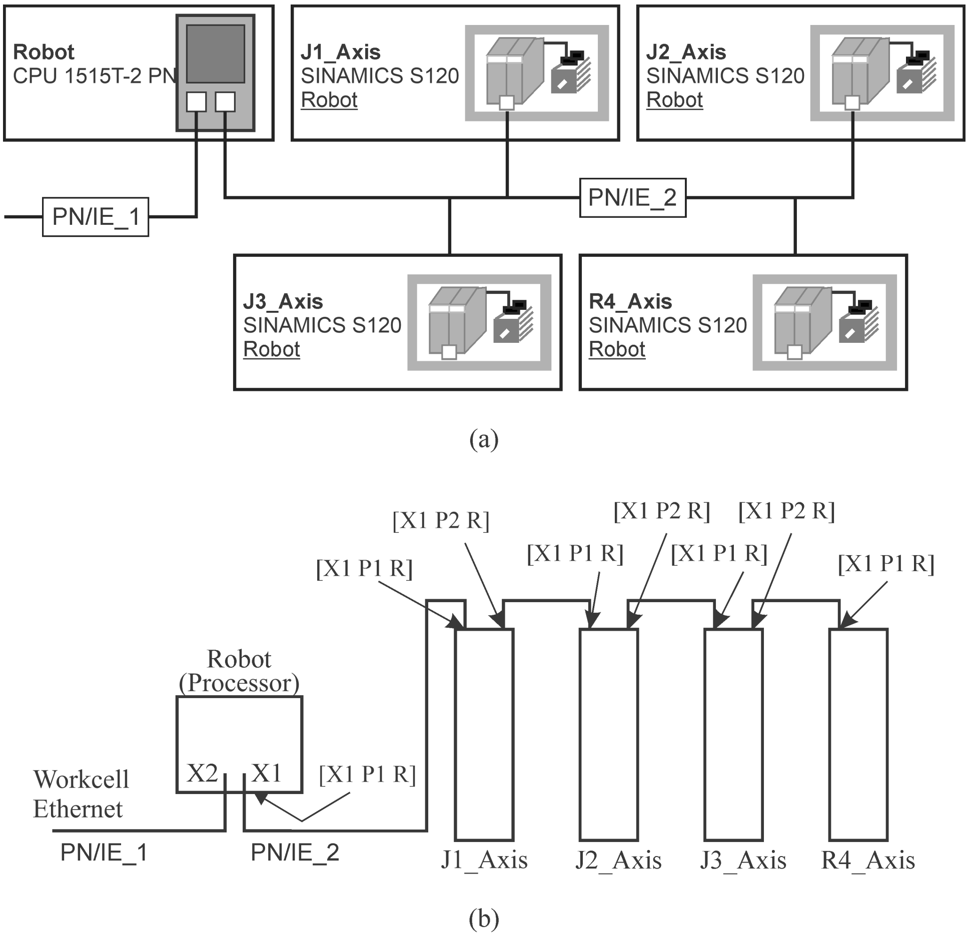


Figure 2.13. Siemens drive connection example: (a) network view; (b) device network connections.

drive performs its control calculations, is typically much faster than the Send Clock time (0.125 ms for most servo drives) and is synchronized to the network Send Clock. The timing of motion on the PROFINET network is shown in Figure 2.14. The steps are as follows:

1. Drive scan runs, calculating actual position and placing it in send telegram.
2. At start of Data Cycle, D-to-C telegram with actual position is transmitted
3. Processor receives D-to-C telegram
4. MC-Servo OB runs, calculating command position and placing in telegram
5. At start of next Data Cycle, C-to-D telegram with command position is transmitted
6. Drive receives C-to-D telegram

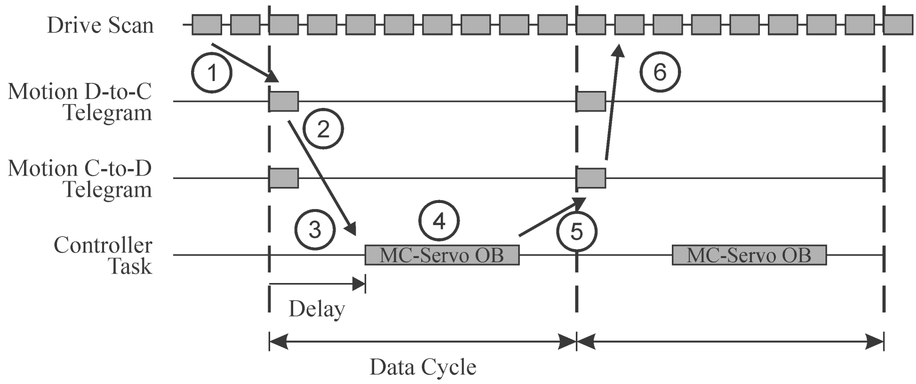


Figure 2.14. Siemens PROFINET data cycle timing model.

The MC-Servo OB execution is synchronized to Data Cycle, starting after a delay that allows the D-to-C telegrams for all drives to be received before the start of MC-Servo. Note that the C-to-D telegram is sent at the start of the next Data Cycle. In IRT mode, both D-to-C and C-to-D telegrams are transmitted at the start of a cycle. Each device on the network is allocated a "slot" to transmit/receive its telegrams.

2.5 INSTALLATION CONSIDERATIONS

This section outlines best practices that minimize the possibility of noise-related failures and covers the concept of high-frequency bonding, ground planes, and electrical noise reduction. For more information, see Rockwell Automation (2001) and Siemens (2024).

Bonding Equipotential bonding is the method of connecting metal chassis, assemblies, frames, shields, and enclosures to reduce the effects of electromagnetic interference (EMI). Bonding creates a ground plane that provides a noise shield in addition to acting as a reliable return path.

Firstly, most paints are insulators. For a good bond between metal components, surfaces must be paint-free or plated. Proper bonding between metal surfaces creates a low-impedance return path for high-frequency energy. Improper bonding blocks the direct return path, allowing high-frequency energy to travel elsewhere in the cabinet and affect the operation of the servo drives.

Secondly, the bonding cables must be wire braid. High-frequency signals travel on the outer surface of a conductor. Thus, a cable braided with several small conductors has a lower impedance than normal conductor cables.

Figure 2.15 shows three subpanels connected with wire braid. Note that both the top and bottom of each panel is connected with a wire braid. Rockwell (2011) and Siemens (2024) also recommend that the cabinet panel should have a high-frequency compatible connection to a foundation ground or meshed network.

Safety Grounding. The safety ground ensures that all metalwork is at the same ground (or Earth) potential at power frequencies. The safety ground circuit and the noise current return circuit may share the same path and components, but they should be considered as