



Machine Control Traceability Interface

Global Common

SD-1034

ISSUED	January 13, 2016
REVISED	January 13, 2016

© 2016 Nexteer Automotive

All rights reserved.

Table of Contents

1.	Scope and Purpose	4
1.1	Scope.....	4
1.2	Purpose and Objectives.....	4
1.3	General System Information and Requirements	4
2.	Logic Routine Requirements (associated routine name)	6
2.1	ASCII Character Development (<i>R23_ASCIIChar</i>).....	6
2.2	Communication Buffer (<i>R27_Trace_VI_Queue</i>).....	6
2.3	Traceability Interface and Initiation (<i>R26_Trace_VI_Station</i>)	7
2.4	Get Request (Permission to Run).....	9
2.5	In Process Request (Ownership):.....	10
2.6	Send Request (Send Results):	10
2.7	Serial Number Generation (<i>R28_Trace_VI_SerialGen</i>)	11
3.	PC Based Control Systems	14
3.1	CSV File Creation – Nexteer Data Transfer	14
3.2	CSV File Creation – File Share Server	15
A.	Traceability Timing Chart.....	16
B.	Traceability Application Functions.....	17
C.	Traceability PLC Tag Descriptions	18
D.	Traceability Field Guidelines.....	22
E.	Status Code Examples.....	24
F.	PC Time Synchronization	25
G.	Process Examples.....	26
H.	Traceability Import Process	29
I.	PLCID Naming Conventions.....	39
J.	StationID Naming Conventions.....	39
K.	PLC Routine Structure.....	40
L.	Data Flow within PLC	40
M.	Network Architecture.....	41

List of Figures

Figure 1:	<i>Trace_VI_Queue</i> Routine Relocation (Controller Organize)	29
Figure 2:	ID_PLC Addon Instruction Modifications	30
Figure 3:	<i>Trace_VI_Queue</i> JSR Instruction	30
Figure 4:	<i>Trace_VI_Station</i> Import Configuration – Routine Properties	30
Figure 5:	<i>Trace_VI_Station</i> Configure Routine Properties	31
Figure 6:	<i>Trace_VI_Station</i> Import Configuration – Configure tag References	31
Figure 7:	Configured tag References	32
Figure 8:	Controller Organizer after importation.....	33
Figure 9:	Station ID Setup.....	34
Figure 10:	Traceability Function Setup.....	34
Figure 11:	Function Number Setup	35
Figure 12:	LookupID Setup.....	35
Figure 13:	Model Setup	35
Figure 14:	Activate Traceability.....	36
Figure 15:	Establish Part Permission Conditions	36
Figure 16:	Enable Part In Process	36
Figure 17:	Enable Part Processed	37
Figure 18:	Results Data Name Array Setup	37
Figure 19:	Results Data Array Setup.....	37
Figure 20:	Data Length Setup.....	38

1. Scope and Purpose

1.1 Scope

- 1.1.1 This specification describes the PLC logic and PC based scripting design requirements and format for Nexteer Automotive facilities utilizing Nexteer's Traceability System. This specification shall be used by the Original Equipment Manufacturers (OEM) in their design of PLC and PC based systems.
- 1.1.2 This specification applies to the equipment requiring Traceability communication for process flow, electronic error proofing, and data collection. Refer to the Manufacturing Engineer's written specification for details regarding traceability requirements.
- 1.1.3 This specification has associated PLC logic and HMI templates that reflect the requirements of this specification. In addition, the templates provide logic routines, screens, and examples that may be applied to new equipment designs. All templates are available at www.nexteerdatabexchange.com.
- 1.1.4 The use of the word "shall" indicates requirements and the use of the word "should" indicates recommendations. The use of the word "may" indicates permission or allowance and the use of the word "can" indicates a possibility.

1.2 Purpose and Objectives

- 1.2.1 The purpose of this specification is to provide Nexteer requirements and guidance to Original Equipment Manufacturers (OEM) for use in their PLC logic and PC based system designs to interface with Nexteer's Traceability System.
- 1.2.2 The objective of this specification is to provide common, maintainable, and cost effective traceability controls systems that enhance both the productivity and ease-of-use of the system, plus ensure the quality of Nexteer products produced. The application of this specification will results in common traceability controls systems software.

1.3 General System Information and Requirements

- 1.3.1 The Nexteer traceability systems are integrated at the machine, cell (group of machines), or asynchronous assembly line level. Depending on the configuration of the traceability system, it may cover multiple cells and / or multiple asynchronous assembly lines.
- 1.3.2 The communication between the control system and the traceability PC is by Ethernet.
- 1.3.3 The Traceability computer, often referred to as the Trace PC, is the central computer for a traceability system. The Traceability PC runs the Nexteer Traceability Application which contains an SQL Server Express database, and any other software packages required for the traceability system.
 - 1. The Nexteer Traceability Application is the interface between the Machine Control System and the SQL Server Express database located on the traceability computer.
 - 2. The SQL Server Express database format is set up the same on all Traceability computers.

1.3.4 A PLC control system interfaces with the Nexteer traceability application by utilizing the required logic routines listed below. These routines are available in the logic templates, and meet the communication tag naming and structure described in this specification.

1. ***R23_ASCIIChar***
2. ***R26_Trace_V1_Station***
3. ***R27_Trace_V1_Queue***
4. ***R28_Trace_V1_SerialGen***

*Note: **R28_Trace_V1_SerialGen** is only required on equipment that is generating serial numbers.*

1.3.5 The OEM is expected to replace all PLC tags that begin with “REPLACE” with the appropriate PLC tag and data.

1.3.6 The Nexteer Traceability Application is not used for PC based machines. PC based control systems interface with the traceability SQL Server Express database directly using SQL commands to read and write records. PC based control systems may also generate CSV files that contain the part status and process results of the part.

2. Logic Routine Requirements (associated routine name)

The following routine requirements ensure the compatibility and functionality of PLC based control systems interfacing with the Nexteer Traceability system.

2.1 ASCII Character Development (*R23_ASCIIChar*)

The *R23_ASCIIChar* routine includes structured text to populate specific PLC tags with ASCII character constants which are used throughout the traceability routines.

Requirements:

- 2.1.1 The *ASCIIChar* routine shall be present within the *MainProgram* under the *MainTask* of the Controller.
- 2.1.2 The Main routine shall call *ASCIIChar* unconditionally through a JSR instruction.
- 2.1.3 Only one *ASCIIChar* routine is required per PLC.

Guidance:

- 2.1.4 When importing the *ASCIIChar* routine, all tags / UDTs created during the import process do not require modification and should be kept as Controller Scope tags.

2.2 Communication Buffer (*R27_Trace_VI_Queue*)

The *Trace_VI_Queue* routine buffers communications between the PLC and the traceability PC. This routine is required to be modified by the OEM to meet the required design needs of each application (reference Annex H for the Traceability Import Process).

Requirements:

- 2.2.1 The *Trace_VI_Queue* shall be present within the *MainProgram* under the *MainTask* of the Controller.
- 2.2.2 The Main routine shall call *Trace_VI_Queue* unconditionally through a JSR instruction.
- 2.2.3 Only one *Trace_VI_Queue* routine is required per PLC.
- 2.2.4 The PLC ID, stored in the tag *Trace.PLCID*, shall be configured in the *Trace_VI_Queue* routine according to the machine's SD Number. A unique PLC ID is used to identify each PLC communicating with the traceability PC. The PLC ID is constructed as follows:
 - 1. Char1 & Char2: ASCII characters "S" and "D" indicating the beginning of the machine asset number.
 - 2. Char3 – Char8: ASCII characters 0-9 conclude the identity of the machine asset number.
 - 3. Char9: ASCII character to identify the machine controller letter code. A single PLC Controller utilizes the letter "X". An example of a PLC ID for a single controller is "SD123456X".
- 2.2.5 The *Trace.HeartbeatTimeout* status shall be part of initial conditions to enter a machine cycle and trigger a fault to indicate a loss of communication between the PLC and the traceability application.

- 2.2.6 The **Trace_VI_Queue** shall not be edited below the phrase “DO NOT EDIT ANYTHING BELOW THIS RUNG.”

Guidance:

- 2.2.7 When importing the **Trace_VI_Queue** routine, all tags / UDTs created during the import process do not required modification and should be kept as Controller Scope tags.
- 2.2.8 When multiple PLC's exist for the same SD number, a unique letter code is required. This is accomplished by using letters “A, B, C...” in place of the “X”. An example PLC ID for multiple controllers is “SD123456A” and “SD123456B”.
- 2.2.9 The logic template **R08_Fault_CycleStop** routine contains an example fault using **Trace.HeartbeatTimeout** status. The heartbeat timeout fault is typically a Cycle Stop fault.

2.3 Traceability Interface and Initiation (R26_Trace_VI_Station)

The **Trace_VI_Station** routine(s) communicate with the traceability PC through requests. There are different types of requests that are detailed below. This routine is required to be modified by the OEM to meet the required design needs of each application (reference Annex A for the Traceability Timing Chart, reference Annex D for Trace Tag Naming Guidelines, and reference Annex H for the Traceability Import Process).

Requirements:

- 2.3.1 **Trace_VI_Station** routine shall be present within the **MainProgram** under the **MainTask** of the Controller.
- Note: On multiple station machines (i.e. dial tables and asynchronous assembly lines), the location of the **Trace_VI_Station** routine(s) may be located in the programs dedicated for each station.*
- 2.3.2 The Main routine shall call **Trace_VI_Station** unconditionally through a JSR instruction.
- 2.3.3 When multiple **Trace_VI_Station** routines are required in a single PLC, each routine shall:
1. Have a uniquely named routine (other than **Trace_VI_Station**). For example: **Trace_VI_PulleyPress**, **Trace_VI_Housing**, **Trace_VI_Motor**, etc...
 2. Have a uniquely named **Station** UDT PLC tag. This should be the same or similar to the routine name. For example: If the routine was named **R26_Trace_VI_Motor**, the **Station** tag could be named **Trace_Motor**.
 3. Have a unique Station ID (“02, 03, 04...”)
 4. Have a unique Station Name.
- 2.3.4 The Station ID, stored in the tag **Station.StationID**, shall be configured as a combination of the PLC ID and a two-digit number as follows:
1. The PLC ID is constructed in the **Trace_VI_Queue** routine.
 2. Num1 & Num2: two-digit numerical value to identify the Station communicating with the traceability PC.

2.3.5 The Station Name, stored in the tag **Station.StationName**, shall be programmed by the OEM. The preferred format is detailed below:

1. A 3-digit department number
2. Area description, such as: BSI, Gray Room, Rack, or Ball Nut
3. Cell or Line number (when applicable)
4. Operation number
5. Machine or station description

For example: 060-Motor Assembly-Line 1-OP220-Hall Calibration

2.3.6 The current running model description shall be copied into the **Station.Model** tag. The Plant should provide a standardized list of model descriptions. These descriptions should be referenced on a manufacturing sequence chart. Model descriptions should be descriptive and be consistent from one station to another.

2.3.7 The **Station.TraceActive** OTE instruction enables the traceability functions in the **Trace_VI_Station** routine(s). The logic that controls the **Station.TraceActive** OTE instruction shall:

1. Remain enabled throughout the entire part processing sequence and until all traceability operations are complete.
2. Not use a part present or sensor signal that may transition during the machine in cycle.

*Note: Disabling the **Station.TraceActive** tag will abort the Trace function requests and clears all Trace tag data.*

3. Turn off between cycles.

2.3.8 The **Station.SerialValidated** OTE instruction shall be enabled once the serial number for the part being processed has been scanned and validated (data content and length).

2.3.9 The scanned serial number shall be copied to the **Station.Serial** tag. The serial number should be copied at the same time or prior to turning on the **Station.SerialValidated** OTE instruction.

2.3.10 The **Trace_VI_Station** shall not be edited below the phrase “DO NOT EDIT ANYTHING BELOW THIS RUNG.”

2.4 Get Request (Permission to Run)

The Get Request is used to request information from the traceability application (SQL database) on a specific part serial number (reference Annex B for details on available Get Request functions).

Requirements

- 2.4.1 Enable the **Station.EnableGet** OTE instruction.
- 2.4.2 The **Station.GetFunction** tag shall be loaded with the Get Request function decimal value being performed.
- 2.4.3 The **Station.SerialValidated** OTE instruction triggers the Get Request function to the Nexteer Traceability Application.
- 2.4.4 The **Station.StationLookupID[1] ... [9]** tag array is primarily used to store Station ID values used to compare against returned Station ID values for permission to run.
 - 1. The **Station.StationLookupID[0]** is loaded with the current Station ID by default and shall not be changed.
 - 2. The **Station.StationLookupID[1]** tag value is used by Get Request functions 2 and 4. The **Station.StationLookupID[1]** shall contain the specific Station ID for the desired record and/or data.

*Note: The **Station.StationLookupID[1]** may contain multiple Station ID's separated by a comma, to search multiple stations at once.*

For example: SD123456X01,SD987654X01,SD765432X01
- 2.4.5 Data is returned to the PLC when the **Station.Seq_GetResultsReturned** OTE is turned on. The data returned is located in the **Station.GetResults.xxxxxx** tags in the PLC.
- 2.4.6 The logic shall be programmed to appropriately control the **Station.PartOKConditions** OTE instruction for the application.
 - 1. There may be several compare branches for the allowed conditions to enable the **Station.PartOKConditions** OTE instruction.
 - 2. The EQU instruction for the status compare may contain any of the preloaded tags Trace.StatusGoodPart, Trace.StatusRejectPart, or Trace.StatusInProcess tag.

Note: Unique reject codes may also be utilized if suitable for the application.
- 2.4.7 The **Station.PartOK** tag shall be used in the sequence routine to allow the sequence of the machine to continue processing the part. The **Station.PartOK** tag shall also be used to control the part status message display to indicate the part is ok to run.
- 2.4.8 The **Station.PartNotOK** tag shall be used in the sequence routine to prevent processing the part and complete the sequence as needed. The **Station.PartNotOK** tag shall also be used to control a fault condition and the part status message display to indicate the part is not ok to run.

2.5 In Process Request (Ownership):

The in process request function sends a part status of 9000 to the SQL Database when enabled. This function is typically used to prohibit the reprocessing of parts. It shall be enabled by a sequence step before the machine begins to alter the part.

Requirements:

- 2.5.1 Enable the *Station.EnableInProcess* OTE instruction.
- 2.5.2 The *Station.InProcessFunction* tag shall be loaded with the In Process Request function with a decimal value of 15.
- 2.5.3 The *Station.PartInProcess* OTE instruction triggers the part In Process Request function to the Nexteer Traceability application. The *Station.PartInProcess* OTE shall be enabled by a sequence step before the machine begins to alter part.
- 2.5.4 When the in process request is complete the *Station.Seq_InProcessRequestComplete* tag will be enabled. The *Station.Seq_InProcessRequestComplete* tag shall be used in the machine sequence steps to verify the traceability communications for the part have completed prior to continuing to processing the part.

2.6 Send Request (Send Results):

The send request is used to send the part status results and data to the Traceability Application (SQL database) on a specific part serial number (reference Annex B for details on available Send Request functions).

Requirements:

- 2.6.1 Enable the *Station.EnableSend* OTE instruction.
- 2.6.2 The *Station.SendFunction* tag shall be loaded with the send function decimal value being performed.
- 2.6.3 The *Station.Status* tag shall be loaded with the quality status of the processed part. This value shall be “9999” for a good part or a unique “1xxx” value representing a reject code (reference Annex E for Status Code examples).
- 2.6.4 The *Station.PartProcessed* OTE instruction triggers the Send Request function to the Nexteer Traceability application. The *Station.PartProcessed* OTE instruction shall be enabled by the machine sequence step that indicates the part has been processed and the quality status of the part is known.
- 2.6.5 When the Send Request function is complete the *Station.AllCommunicationsComplete* tag will be enabled. The *Station.AllCommunicationsComplete* tag shall be used in the machine sequence steps to verify that the traceability communication has been completed, prior to releasing the part to be unloaded.
- 2.6.6 The *REPLACETrace_Name[x]* and *REPLACETrace_Data[x]* result name and data arrays shall be updated to reflect the appropriate name and data for the processed part to be collected.

Note: As previously mentioned, the PLC tag descriptor “REPLACE” for each array is intended to be updated by the OEM to reflect a better description of the data.

1. **REPLACETrace_Name[x]**: the results data names shall be hard coded in this array and be aligned with the data stored in the data array element.
2. **REPLACETrace_Data[x]**: the results data shall be copied into this array each cycle once the part status is known, prior to triggering the **Station.PartProcessed** OTE instruction.
3. All information shall be of STRING data type.

Note: If DINT data type is used for data collection, a DTOS instruction shall be used to convert the raw data into STRING format. If REAL data type is used for data collection, an RTOS instruction shall be used to convert the raw data into STRING format.

- 2.6.7 The COP instruction that copies the results data into **Station.SendRequest.Name[0]** and **Station.SendRequest.Data[0]** tags, shall be set to a length equal to the **REPLACETrace_Name[x]** and **REPLACETrace_Data[x]** data array sizes.
- 2.6.8 The result name and data arrays are aligned as follows
- **REPLACETrace_Name[0]** shall describe the data in **REPLACETrace_Data[0]**
 - **REPLACETrace_Name[1]** shall describe the data in **REPLACETrace_Data[1]**
 - **REPLACETrace_Name[...]** shall describe the data in **REPLACETrace_Data[...]**
 - **REPLACETrace_Name[24]** shall describe the data in **REPLACETrace_Data[24]**

Guidance:

- 2.6.9 The array element size should be limited to 25 as to not affect communication speeds with the traceability application and PLC memory usage.
- 2.6.10 When it is required to send more than 25 pieces of data, multiple Name and Data values may be used in a single array element by concatenating them together with a comma separating the values. For example: **Screw #1 Torque [N],Screw #1 Angle [deg], Screw #1 Count**. The Nexteer Traceability application interprets the comma delimited data as separate pieces of data and uploads them to the database accordingly.

Note: Default STRING data types are limited by a character length of 82.

2.7 Serial Number Generation (R28_Trace_VI_SerialGen)

The **Trace_VI_SerialGen** routine is to be utilized when an Operation must generate a product Serial Number. It is constructed using the standard 20 character Serial Number format PPPPPPPYYJJSSSSUU where:

- PPPPPPPP = 8 digit part number (supplied by the PLC)
- YY = 2 digit year (calculated by the Trace PC)
- JJJ = 3 digit Julian day (calculated by the Trace PC)
- SSSSS = 5 digit sequence (calculated by the Trace PC)

- UU = User specified characters (supplied by the PLC)

This routine works in conjunction with the **R23_ASCIIChar** routine.

Requirements

- 2.7.1 The **Trace_VI_SerialGen** routine shall be present within the **MainProgram** under the **MainTask** of the Controller.
- 2.7.2 The Main routine shall call **Trace_VI_SerialGen** unconditionally through a JSR instruction.
- 2.7.3 The **Trace_VI_SerialGen** routine shall be provided with a unique **SerialGen.ID_Station** which is used to identify each PLC communicating with the traceability PC. The **SerialGen.ID_Station** shall be configured according to the machine SD Number with a unique identifier. The **SerialGen.ID_Station** is constructed as follows:
1. Char1 & Char2: ASCII characters “S” and “D” indicating the beginning of the machine asset number.
 2. Char3 – Char8: ASCII characters 0-9 conclude the identity of the machine asset number.
 3. Char9: ASCII character to identify the machine controller letter code.
 4. Char10 & Char11: two-digit ASCII character to identify the Station communicating with the Trace PC.
- 2.7.4 Multiple **Trace_VI_SerialGen** routines may exist in a single PLC, however, each routine shall:
1. Have a uniquely named routine (other than **Trace_VI_SerialGen**).
 2. Have a uniquely named “**SerialGen**” PLC tag.
 3. Have a unique **SerialGen** Station ID (“02, 03, 04, ...”).
 4. Have a uniquely named Station Name.
- 2.7.5 The Station Name shall be programmed in the **SerialGen.StationName** PLC tag.
- 2.7.6 **SerialGen.PartNumber**: shall be loaded with the part number of the product.
- Note: A STRING data type containing more than 8 characters will result in an error.*
- 2.7.7 **SerialGen.UserCharacter1** and **SerialGen.UserCharacter2**: shall be loaded with a user specified values as needed.
- Note: A STRING data type containing more than 1 character in either of these PLC tags will result in an error.*
- 2.7.8 The **SerialGen.RequestSerial** OTE instruction triggers the Serial Number Generation request to the Nexteer Traceability application. The **SerialGen.RequestSerial** OTE instruction shall be enabled by the machine sequence step that requests a serial number be generated.

- 2.7.9 When the Serial Number Generation request is complete the ***SerialGen.RequestComplete*** tag will be enabled. The ***SerialGen.RequestComplete*** tag shall be used in the machine sequence steps to verify the traceability communications for the serial number generation have completed, prior to continuing on with the cycle.
- 2.7.10 The ***SerialGen.Data.Serial*** tag will contain the generated serial number, when the ***SerialGen.RequestComplete*** tag is enabled.
- 2.7.11 The ***SerialGen.HeartbeatTimeout*** status shall be part of initial conditions to enter a machine cycle and shall trigger a fault to indicate a loss of communication between the PLC and the traceability application.
- 2.7.12 The ***SerialGen.RequestError*** status shall trigger a fault to indicate an unsuccessful response from the traceability application. This shall be an immediate stop fault.
- Note: Error message text will be copied to the ***SerialGen.Data.Message*** tag to be used as needed in the PLC program.*
- 2.7.13 The ***Trace_VI_SerialGen*** routine shall not be edited below the phrase “DO NOT EDIT ANYTHING BELOW THIS RUNG.”

3. PC Based Control Systems

Where permissions are required by the PC Based Control System, the PC program can interface with the SQL Database on the Traceability PC directly without utilizing the Nexteer Traceability application.

Storing process results data from a PC Based control system can be handled in two ways:

- If the results of the data are relatively small, a Comma Separated Variable (CSV) file can be stored in a specific file format and folder on the PC. Depending on the machine and or application, these files are inserted into the database using the Nexteer Data Transfer Utility application.
- If the results data contains a large amount of data, a Comma Separated Variable (CSV) file can be stored in a specific file format and folder on the PC. Depending on the machine and or application these files are inserted into the database through a File Share Server.

Requirements:

3.1 CSV File Creation – Nexteer Data Transfer

Requirements for inserting a record into the SQL Database using the Nexteer Data Transfer utility application.

- 3.1.1 CSV files shall contain part result data from a single machine cycle. The CSV file shall have a single row that contains the comma separated data results values. These files are inserted into the database through the Nexteer Data Transfer Utility. The format shall be as follows:

Timestamp,PLCID,StationID,DataFunction,SerialNumber,Status,Model,StationName,Name000,Data000,Name001,Data001,etc...

1. *Timestamp* is the date and time the data was recorded. It shall be formatted as YYYY-MM-DD hh:mm:ss.fff.
2. *PLCID* is the unique identifier of the Controls System as described in Section 2.2.4 of this document. It shall match the Machine Asset Number.
3. *StationID* is the unique Station identifier as described in Section 2.3.4 of this document. The StationID consist of the PLCID with a unique Station identifier.
4. *DataFunction* is the reserved function number for PC Control System CSV File creation. The DataFunction shall be set to '14' to indicate the data was received from a PC Control System.
5. *SerialNumber* is the Serial Number of the part in process that has been captured and validated.
6. *Status* is the current status of the part as determined by the process. '9999' indicates a good part while '1000' indicates a reject. Depending on the requirements of the Manufacturing Engineer's Specification, unique Reject Codes may be required.

7. *Model* is the current model of the part being processed at the Station. This can be either a numerical value or written text; whichever is required.
8. *StationName* is the specific text referring to the Operation and Machine Description.
9. *Name(000 – xxx)* is the text that identifies the name of the data stored in the associated value of the paired data in the database.
10. *Data(000 – xxx)* is the value of the data assigned to the associated name in the database.

3.1.2 CSV file names shall conform to the following format:

YearMonthDay-HoursMinutesSeconds-SerialNumber.csv

Example: '20150902-143015-12345678901234567890.csv'

3.1.3 CSV files shall be located in the 'C:\Local Traceability\' directory.

3.1.4 This PC must be synced to a PC time server (see Annex F for instruction on syncing to a time server PC).

3.2 CSV File Creation – File Share Server

Requirements for inserting a record into the SQL Database using a File Share Server application.

3.2.1 CSV files shall contain part result data from a single machine cycle. The CSV file shall contain two rows: the first row shall contain all of the header information with comma separated descriptive names. The second row shall contain the comma separated data values that correspond to the header row.

3.2.2 The CSV result files shall be written to a directory on the target PC entitled "Local Traceability." A subfolder of Local Traceability shall be a year-month folder (i.e. 2015-09). A subfolder of the year-month folder shall be a day folder (i.e. 20150906), that contains all the CSV files for that day's records.

Example: C:\Local Traceability\2015-09\20150906\

3.2.3 Recommended CSV File Name Format:

YearMonthDay-TimeStamp-SN_SD Number.CSV

Example: 20150902-143015-12345678901234567890_SD0123456.csv

3.2.4 The PC Based Control Computer should have an IP address on the Nexteer business network to allow files to be moved to file storage on the Nexteer NexTrace database server. Files will be removed from the target PC on a regular basis (typically every five minutes). This prevents Production staff from having to manage storage on the target PC and also should assure that there is sufficient space to write new measurements. Use of SyncBackPro to move files from the target PC to the Trace PC should be avoided due to operational instabilities that SyncBackPro occasionally displays.

3.2.5 This PC must be synced to a PC time server (see Annex F for syncing to a time server PC).

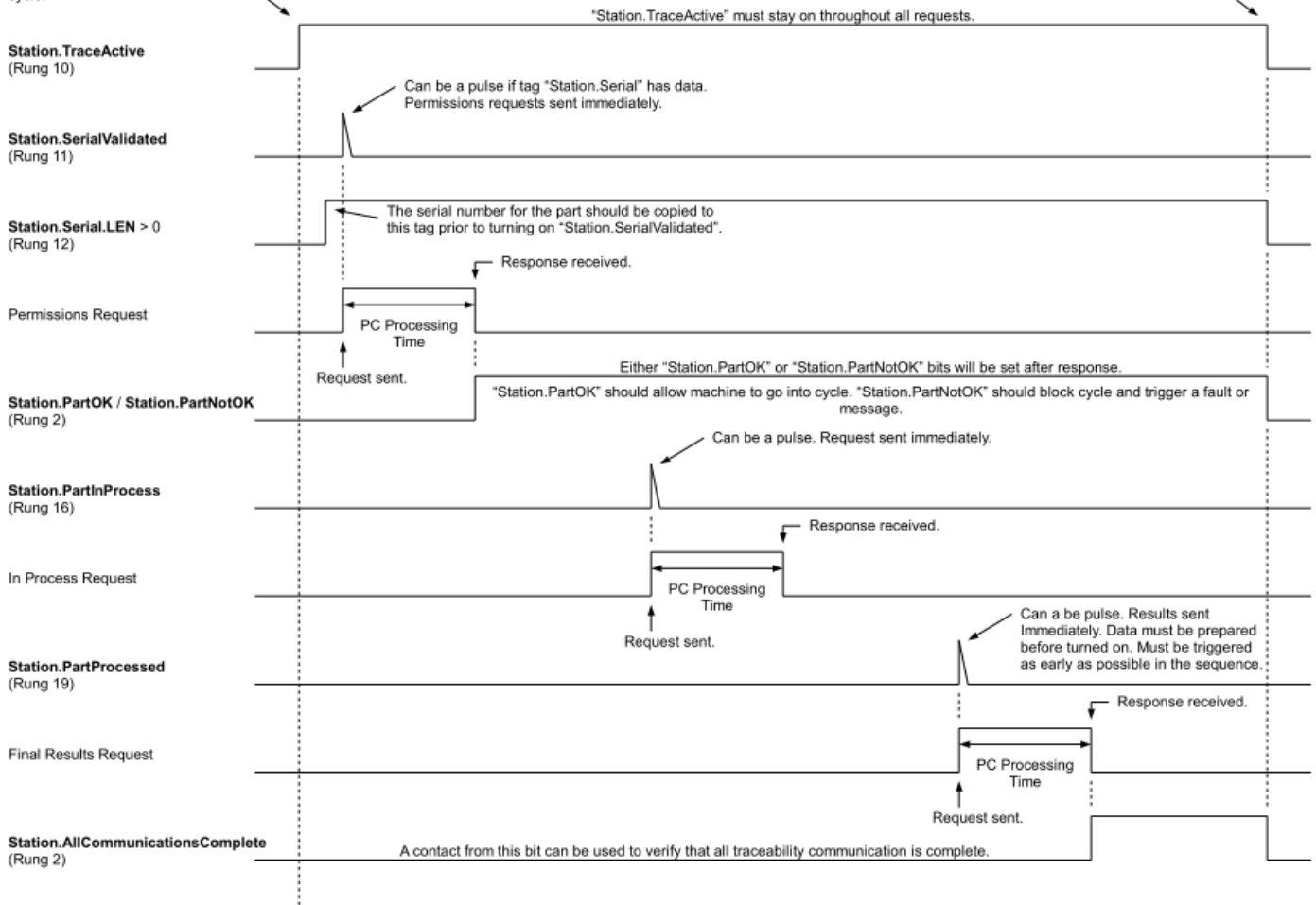
A. Traceability Timing Chart

Trace_V1_Station routine sequence timing guide

Last Updated: February 19th, 2015
By: Adam Romzek

Turning on "Station.TraceActive" enables the traceability sequence. This bit must be driven by a machine sequence bit that will stay on throughout the cycle. Avoid using a PLC input or sensor signal that may flicker during the machine cycle.

Turning off "Station.TraceActive" resets the traceability sequence and clears all data in the Trace_V1_Station routine. Turning this bit off prematurely will result in the loss of data. Note: "Station.TraceActive" must be turned off between cycles.



B. Traceability Application Functions

<u>Function</u>	<u>Description</u>	<u>Function Type</u>	<u>Comments</u>
1	Current Status Lookup	GET FUNCTION	Returns the most recent record in the database for the serial number.
2	Previous Station Lookup + Data	GET FUNCTION	Returns the most recent record in the database from a previous station.
3	Current Status Lookup + Data	GET FUNCTION	Returns the most recent record in the database for the serial number with data for the record.
4	Current Status Lookup + Previous Station Data	GET FUNCTION	Returns the most recent record in the database for the serial number with data from a previous station.
10	Normal Results	SEND FUNCTION	1) Trace App inserts record in Status Table. 2) SQL Trigger on Traceability computer inserts record in Summary Table.
11	Final Assembly Station Results	SEND FUNCTION	Normal results, plus: 3) SQL Trigger on Traceability computer INSERTs record in downstream Traceability computer Status table. If downstream INSERT fails, a RECORD COLLECTION routine in downstream Traceability computer must collect the record.
12	Marriage Data	SEND FUNCTION	Normal results, plus: 3) SQL Trigger on Plant Server INSERTs record in Marriage table on Plant Server. Typically the first 5 Data fields are reserved for marriage serial numbers.
13	Batch data	SEND FUNCTION	Normal results, plus: 3) SQL Trigger performs a query to locate other serial numbers with the same batch number, then INSERTs a record in the Status table for each of the serial numbers with the batch results info.
14	File Record (CSV)	SEND FUNCTION	Results are from a CSV file, and are inserted by another application.
15	In Process	IN PROCESS FUNCTION	Same as Normal results (Function 10) except this function sends a part status of 9000 to the SQL Database.
16	Serial Number Batch Results	SEND FUNCTION	Normal results, plus: 3) SQL Trigger performs a query to locate other serial numbers with the same batch number, then INSERTs a record in the Status Table for each of the serial numbers with the batch results info. Batch number must be located in the Data000 column of the database.
17	Batch Number Results (Batch number is used as the serial number)	SEND FUNCTION	Normal results, plus: 3) SQL Trigger performs a query to locate other serial numbers with the same batch number, then INSERTs a record in the Status Table for each of the serial numbers with the batch results info. Batch numbers must be located in the Serial Number column of the database.
20	Trace App Results Info	SEND FUNCTION	Same as Normal results (Function 10).

C. Traceability PLC Tag Descriptions

- C.1 The following PLC tag requires modification in the *Trace_VI_Queue* routine. It is to be considered an “input” to the Traceability System.

<u>PLC tag</u>	<u>Description</u>
<i>Trace.PLCID</i>	The <i>Trace.PLCID</i> tag is used to build the Station ID in the <i>Trace_VI_Station</i> routine. Only one PLCID should exist in the Controller. At a minimum, ASCII Char's 3 – 8 in the <i>ID_PLC</i> Addon Instruction shall be modified to match the Machine's Asset Number.

- C.2 The following list of PLC tags is provided for use throughout the Controller in the *Trace_VI_Station* routine. They are to be considered “outputs” from the Traceability System.

<u>PLC tag</u>	<u>Description</u>
<i>Trace.HeartbeatTimeout</i>	The <i>Trace.HeartbeatTimeout</i> tag is used to signal a loss of communication with the Trace PC. This tag has been prepopulated in the <i>Fault_CycleStop</i> routine as Fault #134 (<i>Fault_CycleStop[0].5</i>). A loss of communication should prohibit the next cycle from initiating.
<i>Trace.ClearBuffer</i>	Toggling this tag manually will clear all of the content from the <i>Trace.Buffer</i> tags. This is only to be used when bad data has made its way into the buffer which the Trace App cannot process. This includes text strings such as \$, %, ^, &, *, unreadable hex characters, and long string lengths.
<i>Trace.StatusNoRecord</i>	Prepopulated double-integer tag with the value '5000' for use in the <i>Trace_VI_Station</i> routine to indicate a record does not exist in the database. The value '5000' is an error code sent from the Trace PC; <i>Trace.StatusNoRecord</i> can be used as a comparator to the returned error code for part permissions.
<i>Trace.StatusInProgress</i>	Prepopulated double-integer tag with the value '9000' for use in the <i>Trace_VI_Station</i> routine to indicate a part is “In Process”.
<i>Trace.StatusGoodPart</i>	Prepopulated double-integer tag with the value '9999' for use in the <i>Trace_VI_Station</i> routine to indicate a part has been processed successfully and is considered a “Good Part”.
<i>Trace.StatusRejectPart</i>	Prepopulated double-integer tag with the value '1000' for use in the Quality routine to indicate a part has been processed unsuccessfully and is considered a “Reject Part”. This tag is the default value of any Reject Part and should be altered in the Quality routine with the appropriate reject code for the process. A reject code of '1000' typically indicates a general reject but gives no detail to the reason.

C.3 The following PLC tags should be modified within the *Trace_VI_Station* routine. They are to be considered “inputs” to the Traceability System.

PLC tag	Description
<i>Station.StationID</i>	The <i>StationID</i> tag is used to complete the PLCID developed in <i>Trace_VI_Queue</i> . Because each <i>Trace_VI_Station</i> routine requires a unique Station ID, this two character identifier is used to indicate which <i>Trace_VI_Station</i> is communicating with the Trace PC. Num1 and Num2 in the <i>Station.ID_Station</i> Addon Instruction may require changes depending on the application.
<i>Station.StationName</i>	The <i>StationName</i> tag is used to identify the name of the Station in the database. This tag is to be updated with specific text referring to the Operation and Machine Description. Please note that STRING Data Types are limited to 82 characters.
<i>Station.StationLookupID[(1-9)]</i>	The <i>StationLookupID</i> tag can be used to identify the expected Station ID of the previous Operations for permission purposes. Array element [1] can be used for specific Station ID lookup addresses if a Function 2, 3 or 4 are utilized.
<i>REPLACE_CurrentModel</i>	The OEM is responsible for replacing the tag with the appropriate tag from the machine application. The tag should reflect the current model being processed at the Station. This can be either a numerical value or written text; whichever is required.
<i>REPLACE_PartPresentCondition</i>	The OEM is responsible for replacing the tag with the appropriate tag from the machine application. This tag should reflect the parts presence in the Station. A Sequence Step is the preferred input method, however, is not required. NOTE: the <i>TraceActive</i> bit must be enabled during the entire Traceability processes (‘request’, ‘in process’, and ‘send’ functions) for proper functionality.
<i>REPLACE_SerialNumVerified</i>	The OEM is responsible for replacing the tag with the appropriate tag from the machine application. This tag should reflect the Serial Number has been captured and is valid. A Sequence Step is the preferred input method, however, is not required. NOTE: the <i>SerialValidated</i> bit must be enabled during the entire Traceability processes (‘request’, ‘in process’, and ‘send’ functions) for proper functionality.
<i>REPLACE_Barcode</i>	The OEM is responsible for replacing the tag with the appropriate tag from the machine application. This tag is the Serial Number of the part that has been captured and validated.
<i>REPLACE_In_Cycle</i>	The OEM is responsible for replacing the tag with the appropriate tag from the machine application. This tag should reflect the moment in the logic when the part begins the assembly process. A Sequence Step is the preferred input method. Once <i>PartInProcess</i> is active, the logic will send data to the Trace PC indicating a status code of ‘9000’ for the Serial Number of the part.
<i>Seq_Step081SendResults</i>	The OEM is responsible for replacing the tag with the appropriate tag from the machine application. This tag should reflect the moment in the logic when the part has been processed (regardless of good / reject status). A Sequence Step is the preferred input method. Once <i>PartProcessed</i> is active, the logic will send data to the Trace PC indicating the appropriate status code as determined by the Quality routine for the Serial Number of the part.
<i>REPLACETrace_Name[(0-25)]</i>	The OEM is responsible for editing the tag to correspond with the <i>Trace_VI_Station</i> routine. This array is used in conjunction with the Data array. The Name array assigns the description of the paired data in the database. Default size is 10 elements.
<i>REPLACETrace_Data[(0-25)]</i>	The OEM is responsible for editing the tag to correspond with the <i>Trace_VI_Station</i> routine. This array is used in conjunction with the Name array. The Data array assigns the value of the paired name in the database. Default size is 10 elements.
COPY INSTRUCTION Length	The OME is responsible for updating the length of the COP Instructions for <i>SendRequest.Name[0]</i> and <i>SendRequest.Data[0]</i> based on the amount of data sent to the

PLC tag	Description
	Trace PC. Example: if 10 elements of data are being sent [0-9], the length must be updated for both COP Instructions.

C.4 The following list of PLC tags is provided for use throughout the Controller in the **Trace_VI_Station** routine. They are to be considered “outputs” from the Traceability System.

PLC tag	Description
Station.PartOK	When “get request” is enabled and, based on programmed acceptance criterion, the part is deemed ok to run, PartOK will become enabled. PartOK should be used in the Sequence Routine when permissions are required.
Station.PartNotOK	When “get request” is enabled and, based on programmed acceptance criterion, the part is deemed not ok to run, PartNotOK will become enabled. PartNotOK should be used in the Sequence Routine when permissions are required.
Station.InProcessComplete	When “in process” is enabled and part processing has begun, InProcessComplete will become active once the database has recorded the record. This is a handshaking signal returned to the PLC as acknowledgement from the Trace PC.
Station.AllCommunicationsComplete	When “send request” is enabled and part processing is complete, AllCommunicationsComplete will become active once the database has recorded the record. This is a handshaking signal returned to the PLC as acknowledgement from the Trace PC. AllCommunicationsComplete should be used in the Sequence Routine when data collection is required.

C.5 The following PLC tag requires modification in the **Trace_VI_SerialGen** routine. They are to be considered “inputs” to the Traceability System.

PLC tag	Description
SerialGen.StationID	The StationID tag is used to build the Station ID in the Trace_VI_SerialGen routine. The Station ID does not need to be unique from the Trace_VI_Station routine’s Station ID, however, it is recommended that it be changed to lessen confusion. At a minimum, ASCII Char’s 3 – 8 in the ID_StationFull Addon Instruction shall be modified to match the Machine’s Asset Number.
SerialGen.StationName	The StationName tag is used to identify the name of the Station in the database. This tag is to be updated with specific text referring to the Operation and Machine Description. Please note that STRING Data Types are limited to 82 characters.
REPLACE_SerialGenPN	The OEM is responsible for replacing the tag with the appropriate tag from the machine application. The tag should reflect the part number being processed at the Station.
SerialGen.UserCharacter1 SerialGen.UserCharacter2	When required, the OEM is responsible for completing the User Character Setup by altering the final characters of the standard 20 character Serial Number. When not required, the tags should remain the default value of ‘0’.
REPLACE_RequestSerial	The OEM is responsible for replacing the tag with the appropriate tag from the machine application. This tag should reflect the moment in the logic when a Serial Number is to be generated. A Sequence Step is the preferred input method. Once RequestSerial is active, the logic will send data to the Trace PC initiating the request.

C.6 The following list of PLC tags is provided for use throughout the Controller in the *Trace_VI_SerialGen* routine. They are to be considered “outputs” from the Traceability System.

PLC tag	Description
<i>SerialGen.HeartbeatTimeout</i>	The <i>SerialGen.HeartbeatTimeout</i> tag is used to signal a loss of communication with the Trace PC. When utilizing <i>Trace_VI_SerialGen</i> , this tag must be utilized in the <i>Fault_CycleStop</i> routine to indicate a loss of communication. A loss of communication should prohibit the next cycle from initiating.
<i>SerialGen.RequestComplete</i>	When a Serial Number request is sent, <i>RequestComplete</i> will become active once the database has issued answered the request. This is a handshaking signal returned to the PLC as acknowledgement from the Trace PC. <i>RequestComplete</i> should be used in the Sequence Routine when a Serial Number is requested.
<i>SerialGen.RequestError</i>	When a Serial Number request is sent, <i>RequestError</i> will become active if an error occurs while attempting to generate a Serial Number. This is a handshaking signal returned to the PLC as acknowledgement of an error Trace PC. <i>RequestError</i> should be used in the Sequence Routine when a Serial Number is requested.
<i>SerialGen.Data.Serial</i>	Upon a successful generation of a Serial Number, <i>Data.Serial</i> will contain the completed 20 character Serial Number. The Serial Number will be unique and recorded in the database.
<i>SerialGen.Data.Message</i>	Upon an unsuccessful generation of a Serial Number, <i>Data.Message</i> may be used to provide a description of the error that occurred.

D. Traceability Field Guidelines

Timestamp,PLCID,StationID,DataFunction,SerialNumber,Status,Model,StationName,Name000,Data000,Name001,Data001

Cat	Field Name	Example	Comments
Header	Timestamp	2015-06-26 09:06:57.007	<p>“Timestamp” name may not be used for any other field.</p> <p>Time values should differ for each record.</p> <p>All PLC’s and PC’s should be synchronized to a time server to ensure accurate timestamps (reference Annex F for syncing to a timeserver PC).</p>
	PLCID	SD800788X	<p>“PLCID” name may not be used for any other field.</p> <p>Must begin with “SD”.</p>
	StationID	SD800788X01	<p>“StationID” and “StationOP” names may not be used for any other field.</p> <p>Must begin with “SD”.</p> <p>Must be unique – verify after copying PLC Code.</p> <p>Typically, 11 characters in length.</p> <p>StationID is typically referenced on the Manufacturing Sequence chart.</p>
	DataFunction	10	<p>“DataFunction” name may not be used for any other field (reference Annex B for description of all Data Functions).</p>
	SerialNumber	38017975151762436302	<p>“SerialNumber” name may not be used for any other field.</p> <p>Serial Numbers cannot be blank (the text “No Serial” should be used if the serial scan failed).</p> <p>Format depends on the value stream (reference process sheet where format can be retrieved).</p> <p>Some value streams use Direct Part Marking Spec (34000869) to specify the format.</p> <p>The Plant will provide format for master parts, these can be different length and/or format.</p> <p>SerialNumbers may be referenced on manufacturing sequence chart.</p> <p>Component Serial Numbers will not be displayed in this column; they will be one of the early “Data” fields.</p> <p>Only primary and assembly serial numbers will be displayed in this column.</p> <p>Serial Numbers that include Julian dates must have values within range of 1 – 366.</p> <p>Serial Numbers that include shift must have values within 1 – 3.</p> <p>Serial Numbers that include the year must have two digits.</p>
	Status	9999	<p>“Status” and “PLC_Status” names may not be used for any other field.</p> <p>Status Codes may be included on the Manufacturing Sequence Chart (reference Annex E for Status Code Examples).</p>
	Model (Part Number)	L RWD LDLA BASE LHD 05154744	<p>Since part numbers change regularly, it is preferable to use text to describe the Model.</p> <p>“Model” name may not be used for any other field.</p> <p>Plant will provide standardized list of model descriptions. This should be documented on the manufacturing sequence chart.</p> <p>Model descriptions will be consistent from one station to another.</p> <p>Model descriptions should be descriptive.</p> <p>If a Model description is not required, the tag should be populated with “N/A.”</p>

	StationName	084-BSI-Line1-Sta220-OP20 to210 Pallet Data	<p>“StationName” name may not be used for any other field.</p> <p>Should include the Three Digit Department Number indicated on the Manufacturing Sequence Chart.</p> <p>Should include the Cell or Line Name (e.g. BSI, Grey Room, Rack, Ball Nut, etc.).</p> <p>Should include the Cell or Line Designation: “-Cell” or “-Line” plus number (one digit) (e.g. 1, 2, 3, ...) or letter (e.g. A, B, C, ...).</p> <p>Could include a three digit Station Number preceded by “-Sta” (e.g. -Sta220).</p> <p>Could include a three digit Operation Number specified on the Manufacturing Sequence Chart preceded by “-OP”.</p> <p>Could include a range of three digit Operation Numbers specified on the Manufacturing Sequence Chart preceded by “OP” (e.g. – OP20 to 210).</p> <p>Should include a high level description (e.g. Pallet Data).</p>
Name / Data	Name[xxx]	RackSN Magnet Press Distance [mm]	<p>No field name is duplicated.</p> <p>Field names are consistent between lines containing same data.</p> <p>Field names include units in brackets (e.g. [N]).</p> <p>Field names are spelled correctly.</p> <p>Cannot use “FTQ_Results”, “FTQ_Desc”, “FTQ_FirstTime”, “FTQ_PreviousRun”, “FTC”.</p> <p>No field is named a header field (e.g. “Status”).</p>
	Data[xxx]	38010205151761061501	<p>Data content from one filed does not conflict with another, e.g. BSI – cannot have a good status with a Fail Code.</p> <p>Where field content is expected to be variable, such as a press load, indicate this as “Variable Values” on the Manufacturing Sequence Chart.</p> <p>Where field content is supposed to be within a range, such as Pallet Numbers, ensure that the data varies within that range. Indicate this as “Range of Values” on the Manufacturing Sequence Chart.</p> <p>Where field content contains fixed values, such as reading NTC Codes from a part, ensure that the data is consistent. Indicate this as “Fixed Values” on the Manufacturing Sequence Chart.</p> <p>If this part is not required for the current running model, populate this tag with “N/A.”</p> <p>All fields are populated regardless of errors. If there is no data to report, then display “No Data” for text fields and 0 for Numeric fields.</p>

E. Status Code Examples

STATUS CODES FOR SINGLE STATION MACHINES	
Status	Comments
1000-1xxx	General Reject Codes
STATUS CODES FOR MULTISTATION MACHINES	
Status	Comments
1000-1009	General Reject Codes
1010-1019	ST10 – Reject Codes
1020-1029	ST20 – Reject Codes
1030-1039	ST30 – Reject Codes
1xx0-1xx9	STxx – Reject Codes
COMMON STATUS CODES FOR ALL MACHINES	
Status	Comments
2000-2999	Center for Analysis Failures
5000	None or More than 1 Record In Database
5001	Error requesting data from status table
5003	Error inserting record into request log table
5004	Error inserting record into status table
5010	Parameter PLC ID is missing
5011	Parameter Station ID is missing
5012	Parameter Serial Number is missing
5013	Parameter Status is missing
5015	Parameter Function not valid
5020	Multiple parameters are missing
9000	In Process
9999	Good Part

F. PC Time Synchronization

In order to synchronize the Traceability PC time and date to a time server, use the registry editor program 'Regedit' to make the following changes to the PC's registry.

The "NtpServer" IP address listed below is the address for the time server PC at the Saginaw facility. Each Nexteer Site will have their unique time server PC to reference.

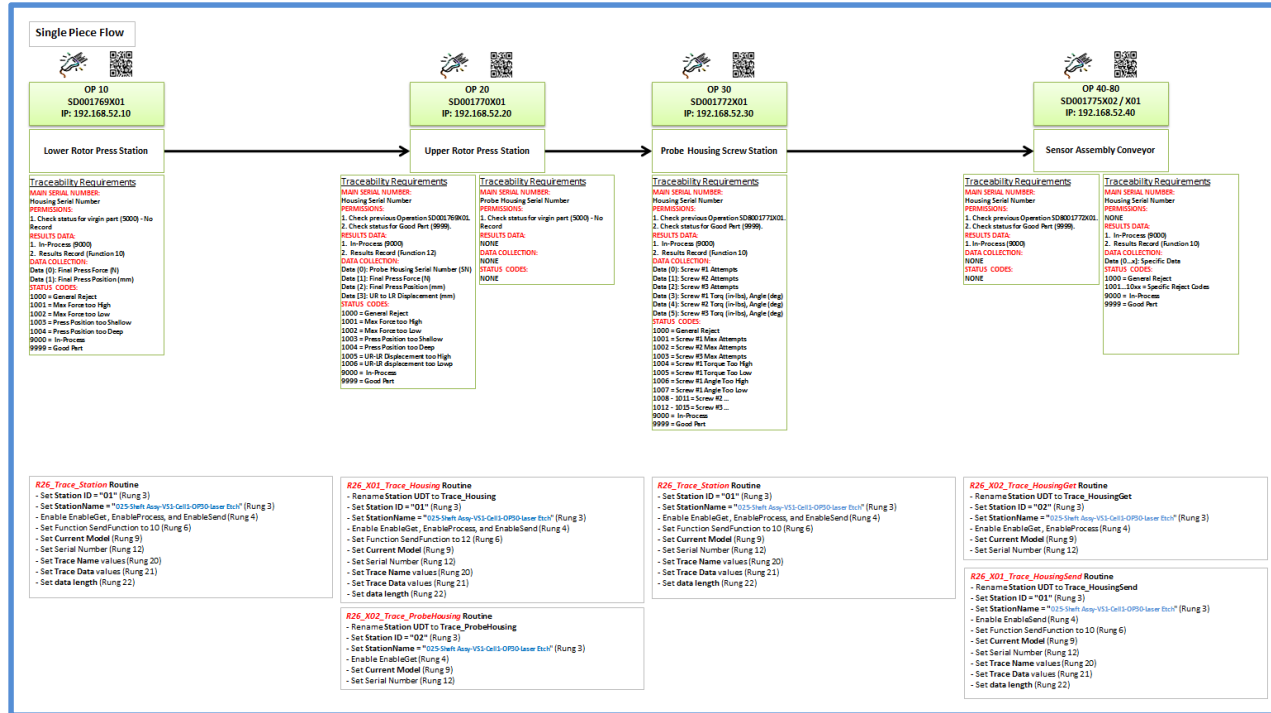
```
; Disable Windows Update
[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU]
"NoAutoUpdate"=dword:00000001

; DISABLE AUTORUN (via .inf files) on all drives
[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer]
"NoDriveTypeAutoRun"=dword:000000FF

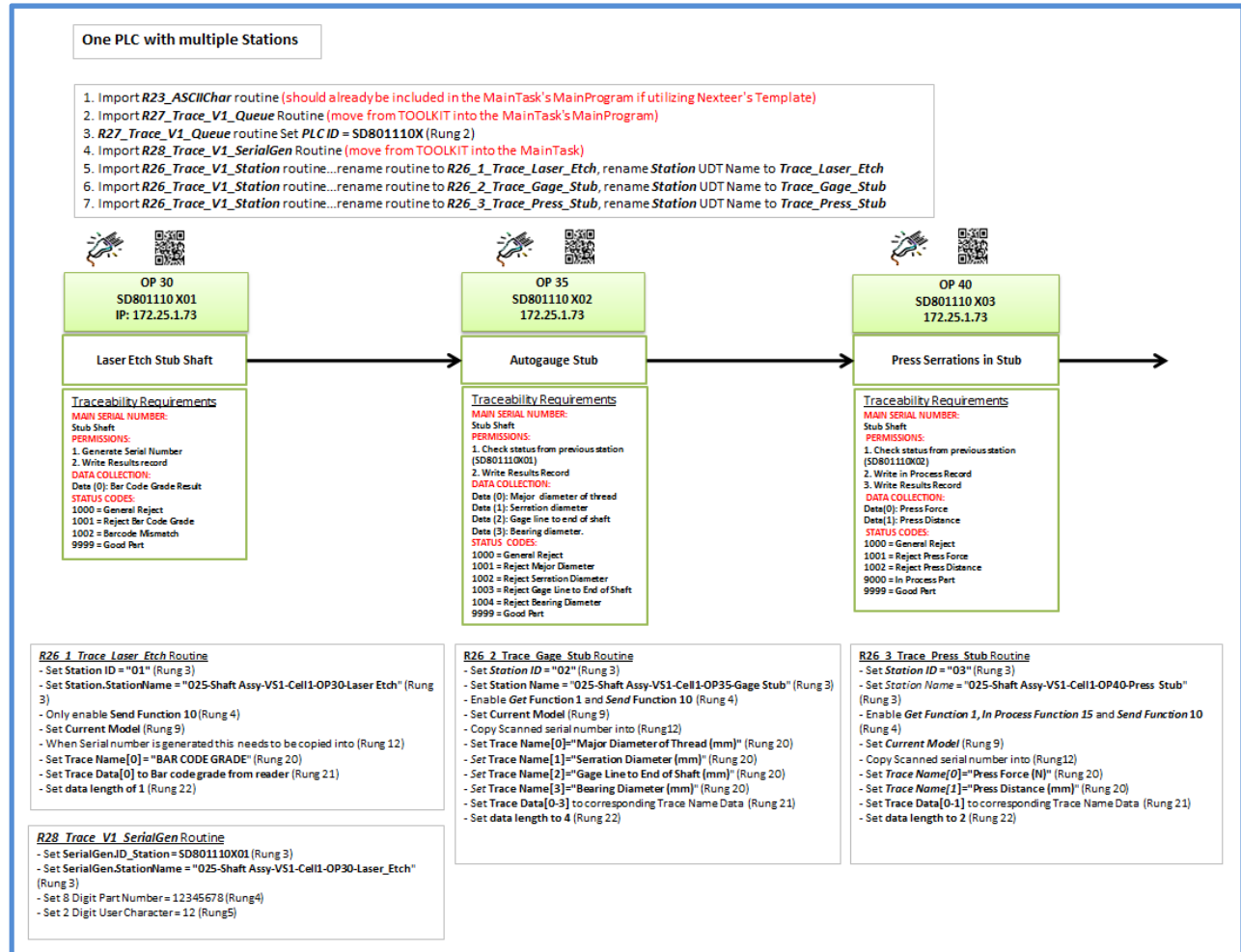
; Use Nexteer's NTP Time server for our own time/date
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\W32Time\Parameters]
"NtpServer"="10.199.100.92,0x9"
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\DateTime\Servers]
@="0"
"1"="time.windows.com"
"2"="time.nist.gov"
"3"="time-nw.nist.gov"
"4"="time-a.nist.gov"
"5"="time-b.nist.gov"
"0"="10.199.100.92"
```

G. Process Examples

G.1 The following is an example of a typical single piece flow Assembly Cell with a Multi-Station asynchronous Assembly Line. Each machine must have a unique **StationID** and have a unique Station UDT PLC tag when multiple **Trace_V1_Station** routines are present.

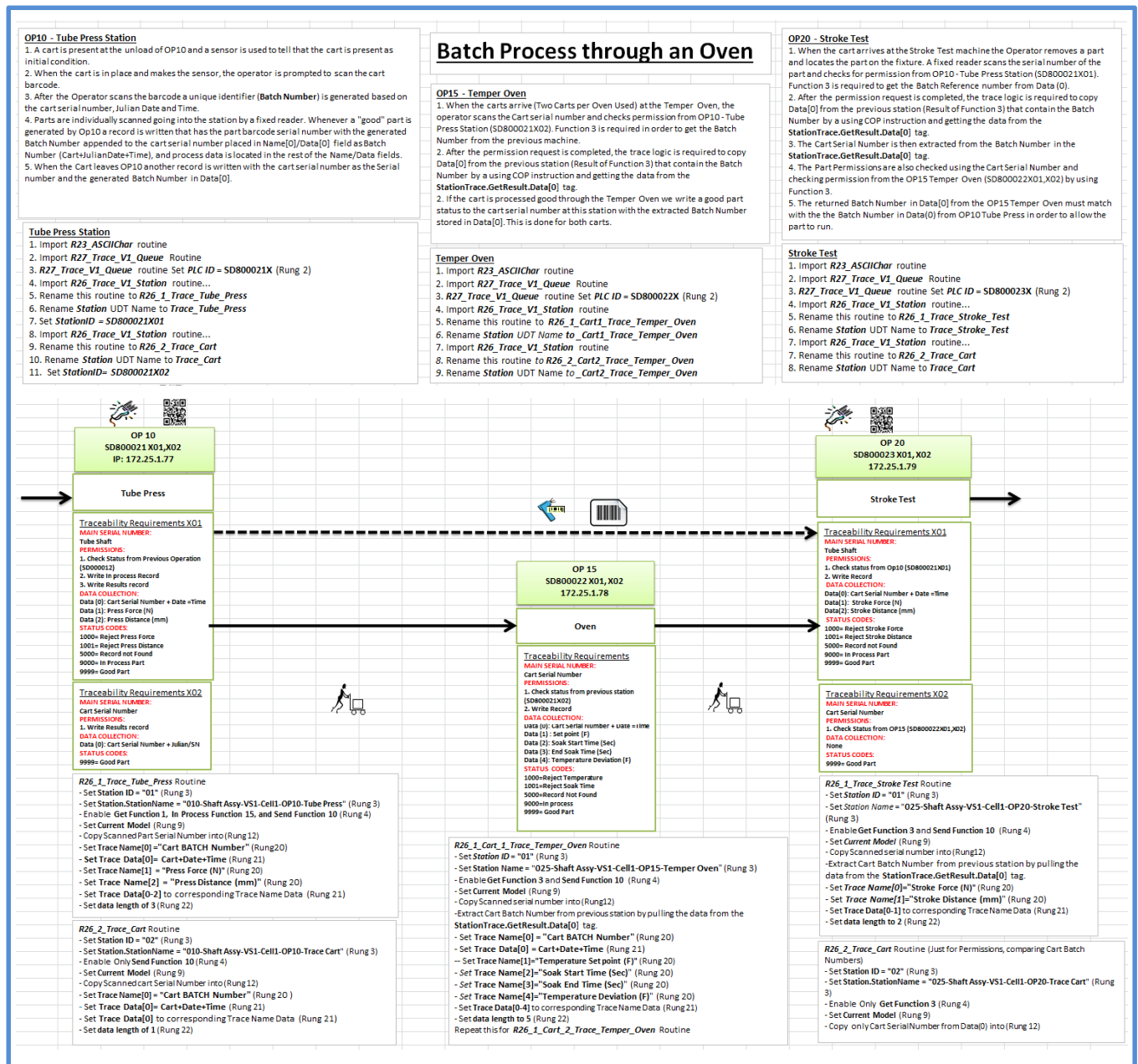


G.2 The following is an example of one PLC controlling multiple Stations. Each machine must have a unique **StationID** and **Station** UDT PLC tag in order to communicate successfully through the **Trace_VI_Queue** routine with the multi-routine complexity of the PLC configuration. In this example, **Trace_VI_SerialGen** is utilized to create a unique Serial Number for the Stub Shaft.



G.3 The following is an example of handling batch information through an Oven. Batch handling is done when individual part tracking is not feasible through a process, such as the Temper Oven in this example. Each machine has a unique PLC controlling only that process. Each machine must have a unique *StationID*.

Note: In cases where the part needs to “cool off” or wait a period of time before the part is allowed to run, The *Trace.Buffer.ElapsedTime* tag can be used. This tag is populated with the elapsed time using the time stamp record in the database and the current time when the GET Request was made. For example...When the parts come out of the Oven below, a record is written for the cart serial number. Every record that is written has a timestamp as part of this record in the Database. When the Stroke Test Machine request permission from the Oven to run the part, the Trace Application will populate the *Trace.Buffer.ElapsedTime* tag with the current time minus the Timestamp from the record that retrieved.



H. Traceability Import Process

This Annex demonstrates the import process and required changes to the Traceability routines for a properly functioning Traceability System. Although the *Trace_VI_Station* routine is located in the Toolkit, for demonstrative purposes, the import process will be shown to indicate all changes required.

- H.1 Begin by moving the *Trace_VI_Queue* routine out of the Toolkit. The *Trace_VI_Queue* routine shall be located in the *MainProgram* of the *MainTask* for all machine types (Single Station and Multi Station designs).

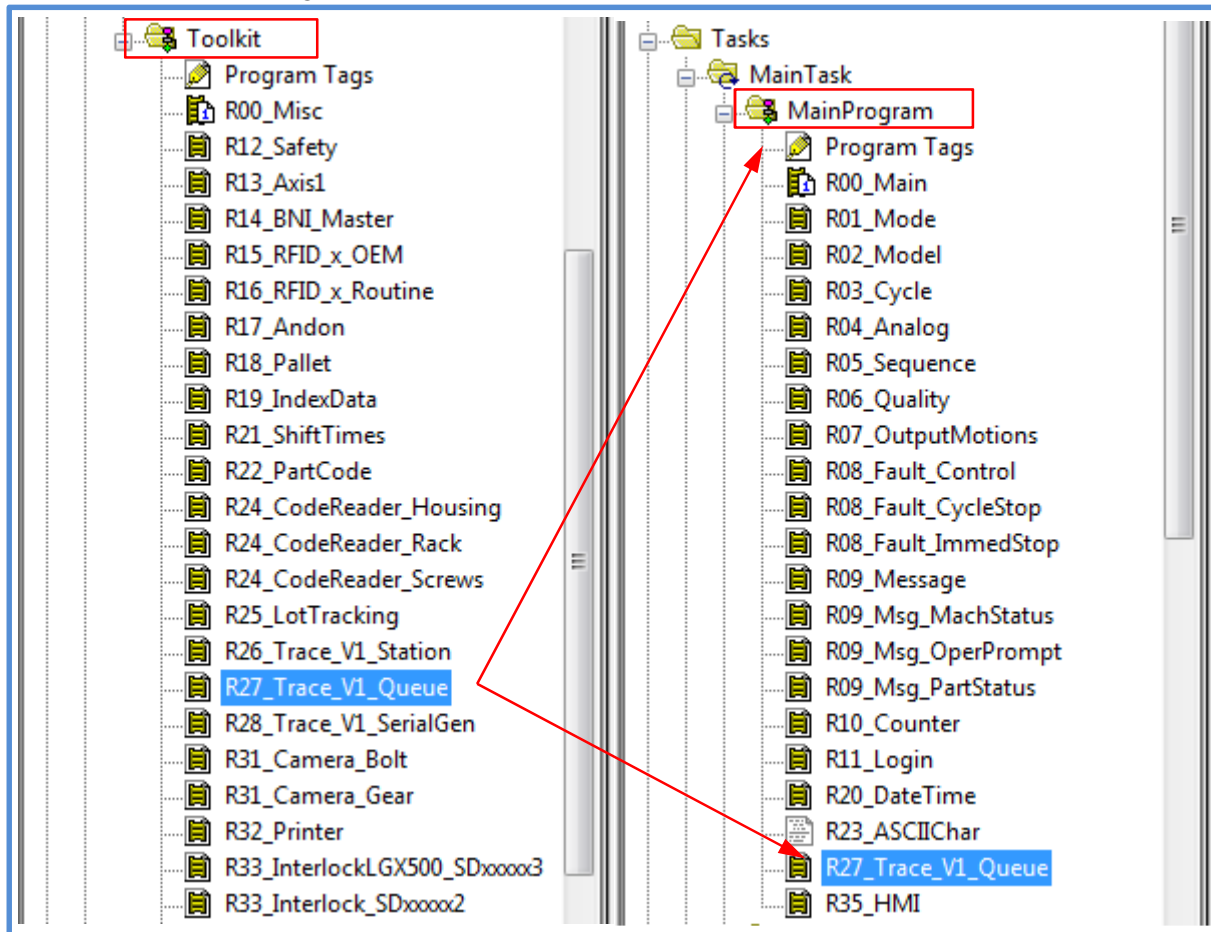


Figure 1: *Trace_VI_Queue* Routine Relocation (Controller Organize)

H.2 Update the ID_PLC Addon Instruction with the appropriate Machine Asset Number. At a minimum, Char3-8 will require updates.

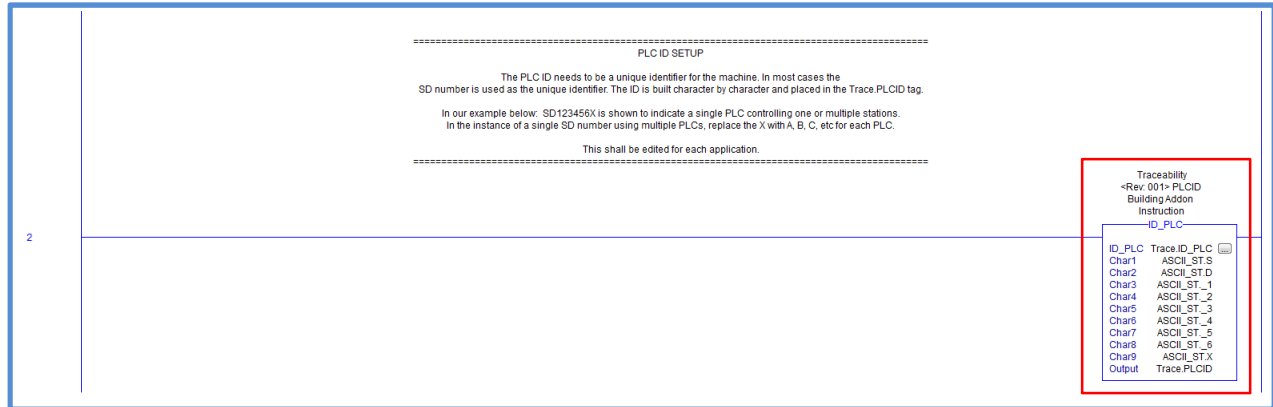


Figure 2: ID_PLC Addon Instruction Modifications

H.3 Add the required JSR Instruction to the Main routine of the **MainProgram**. Refer to SD-1032 for specifics on solving order and placement of the **Trace_V1_Queue** routine.

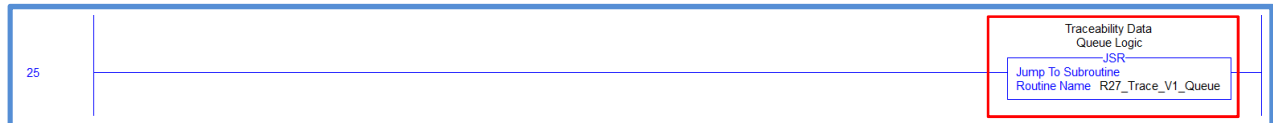


Figure 3: **Trace_V1_Queue** JSR Instruction

H.4 Begin the import process of the **Trace_V1_Station** routine. Depending on Single Station or Multi Station logic designs, the **Trace_V1_Station** routine may not be located in the **MainProgram**. For demonstration purposes, this document will show the **Trace_V1_Station** routine in the **MainProgram**.

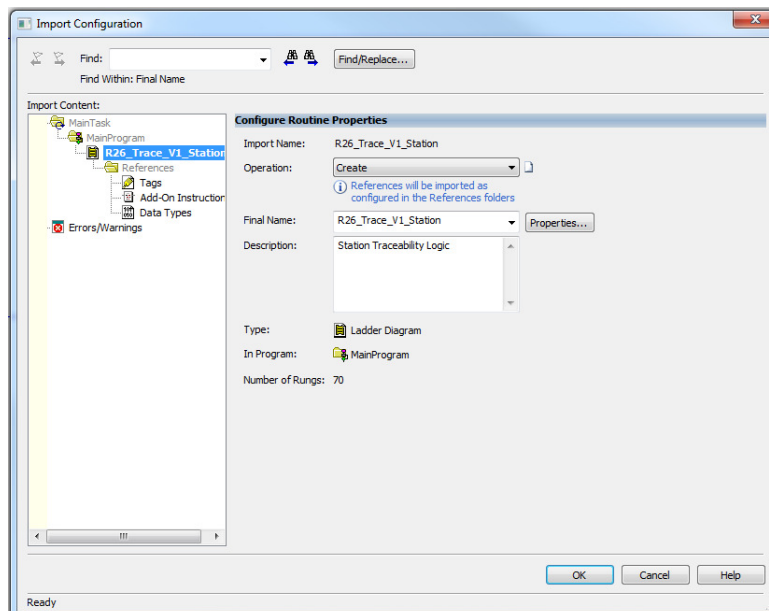


Figure 4: **Trace_V1_Station** Import Configuration – Routine Properties

- H.5 Provide a unique Final Name and Description to the Routine based on the **StationID** unique identifier and machine process. This is especially critical when multiple **Trace_V1_Station** routines are required.

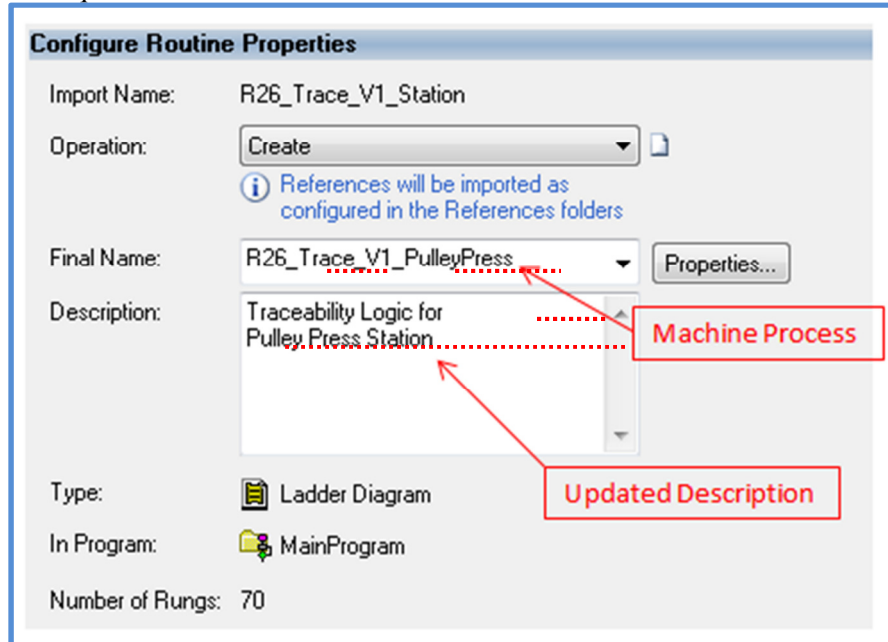


Figure 5: **Trace_V1_Station** Configure Routine Properties

- H.6 Select the Configure tag References to provide unique critical PLC tags.

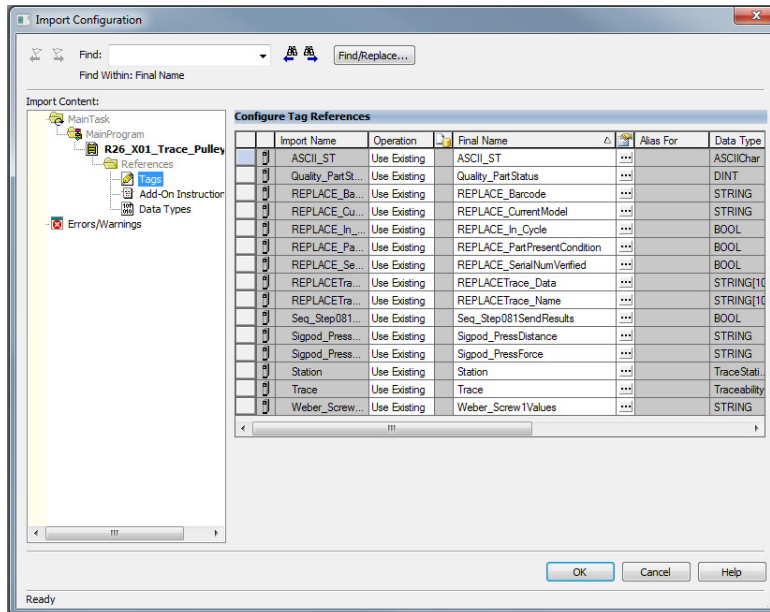


Figure 6: **Trace_V1_Station** Import Configuration – Configure tag References

- H.7 Provide unique PLC tags for the routine. This is especially critical when multiple **Trace_V1_Station** routines are required. In this example, **REPLACeTrace_Data**, **REPLACeTrace_Name**, and **Station** PLC tags have been replaced.

Configure Tag References						
	Import Name	Operation		Final Name	Alias For	Data Type
	ASCII_ST	Use Existing		ASCII_ST	...	ASCIISChar
	Quality_PartSt...	Use Existing		Quality_PartStatus	...	DINT
	REPLACE_Ba...	Use Existing		REPLACE_Barcode	...	STRING
	REPLACE_Cu...	Use Existing		REPLACE_CurrentModel	...	STRING
	REPLACE_In...	Use Existing		REPLACE_In_Cycle	...	BOOL
	REPLACE_Pa...	Use Existing		REPLACE_PartPresentCondition	...	BOOL
	REPLACE_Se...	Use Existing		REPLACE_SerialNumVerified	...	BOOL
*	REPLACETra...	Create		PulleyPress_Trace_Data	...	STRING[10
*	REPLACETra...	Create		PulleyPress_Trace_Name	...	STRING[10
	Seq_Step081...	Use Existing		Seq_Step081SendResults	...	BOOL
	Sigpod_Press...	Use Existing		Sigpod_PressDistance	...	STRING
	Sigpod_Press...	Use Existing		Sigpod_PressForce	...	STRING
*	Station	Create		Trace_PulleyPress	...	TraceStati..
	Trace	Use Existing		Trace	...	Traceability
	Weber_Screw...	Use Existing		Weber_Screw1Values	...	STRING

Figure 7: Configured tag References

H.8 Initiate the import process.

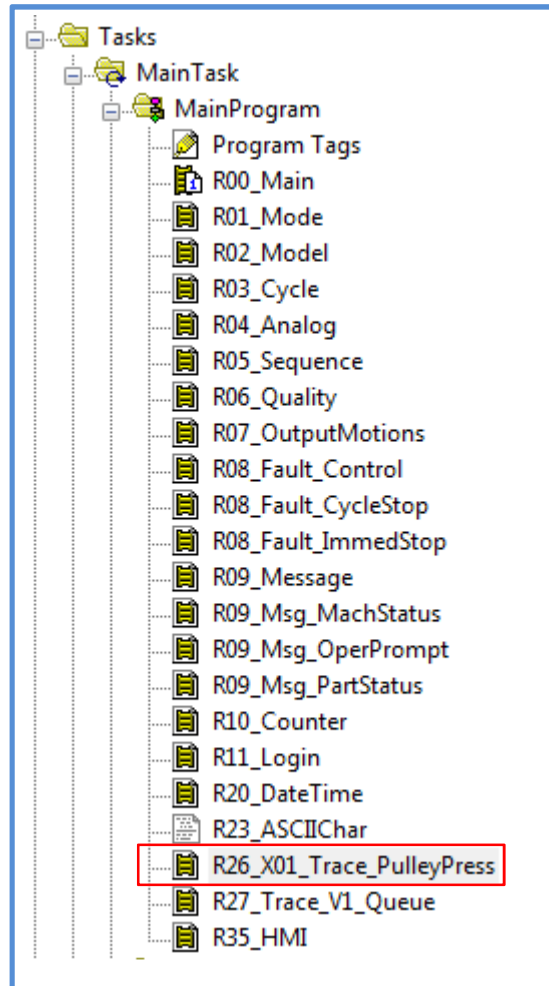


Figure 8: Controller Organizer after importation.

H.9 Update the ID_Station Addon Instruction with the appropriate two digit unique **StationID** identifier. This is especially critical when multiple **Trace_VI_Station** routines are required.

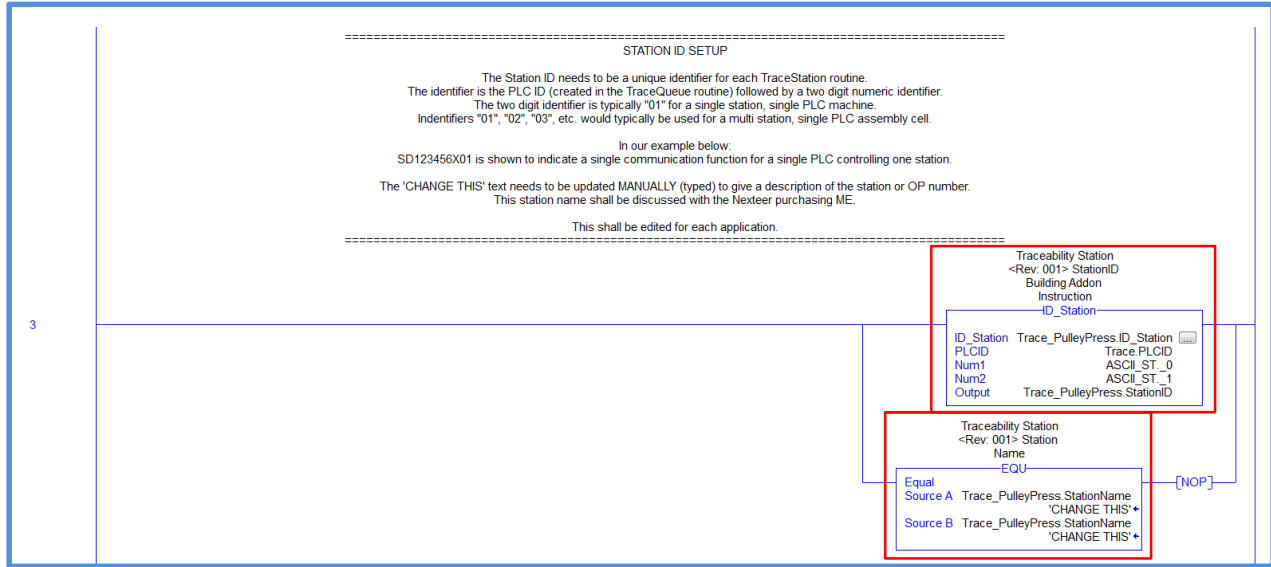


Figure 9: Station ID Setup

H.10 Setup the Traceability Functions that are required for the machine.

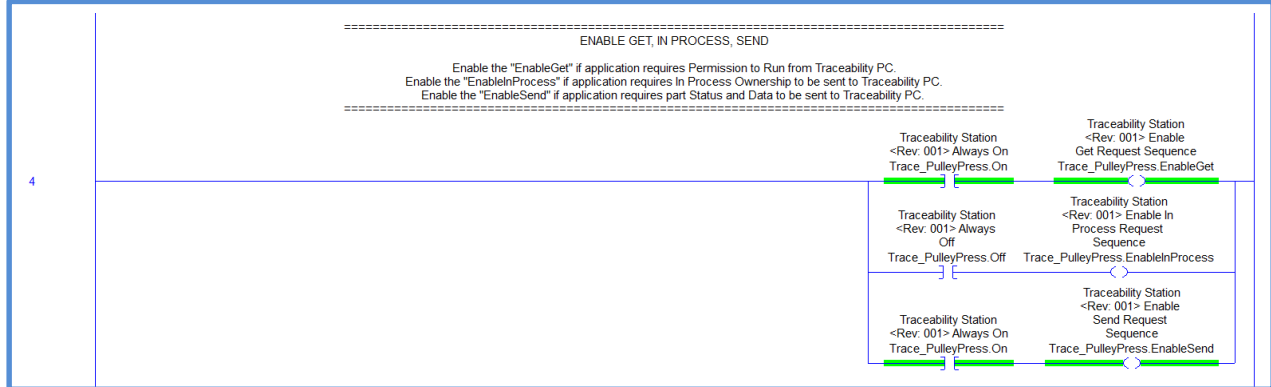


Figure 10: Traceability Function Setup

H.11 Setup the Function Numbers for each Traceability Function.

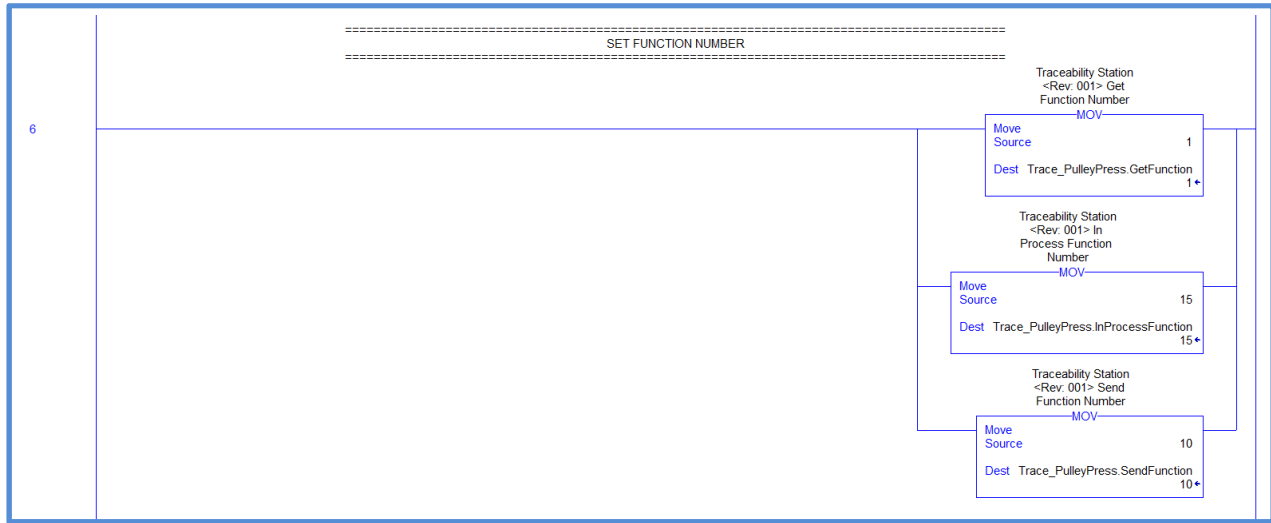


Figure 11: Function Number Setup

H.12 Establish the LookupID's for the expected returned data.

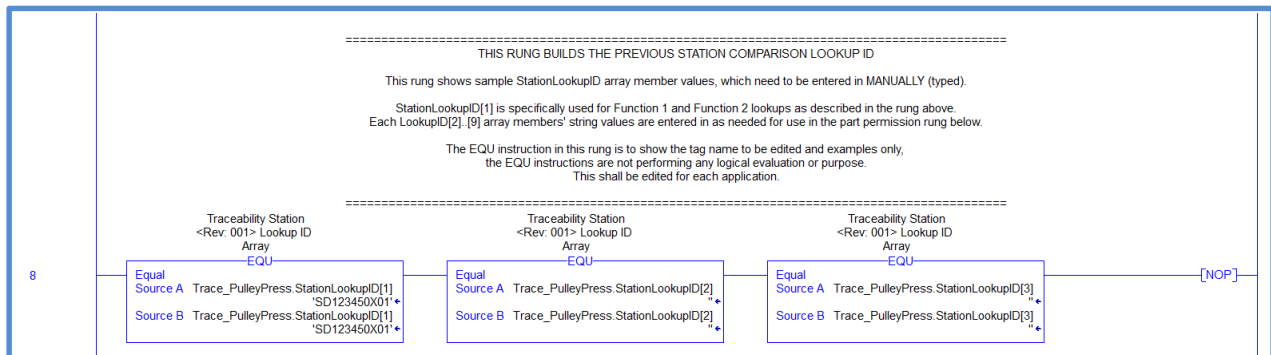


Figure 12: LookupID Setup

H.13 Setup the Current Model.

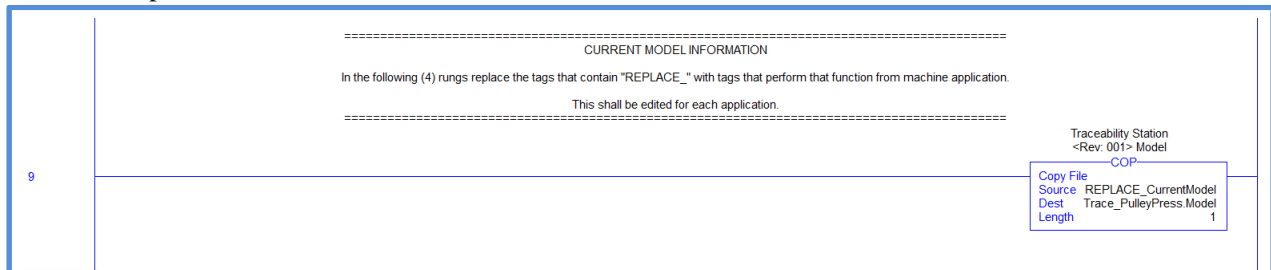
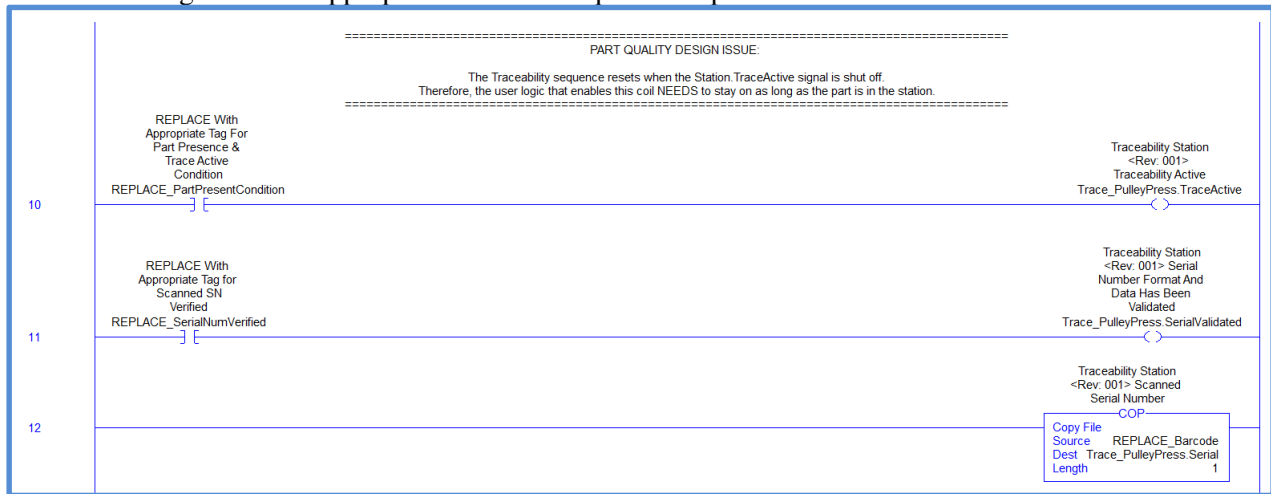
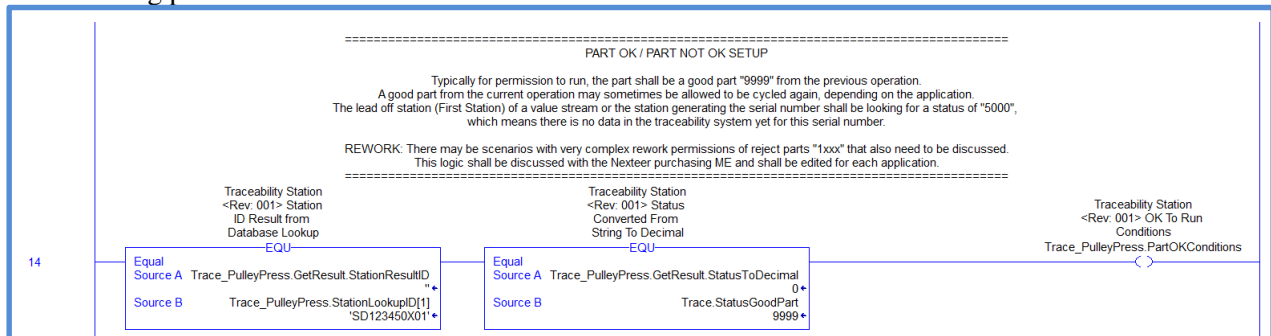


Figure 13: Model Setup

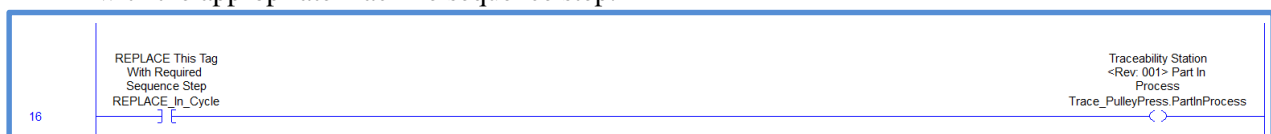
H.14 Setup the conditions to activate the Traceability System. This includes replacing the REPLACE* PLC tags with the appropriate machine sequence step and scanned barcode information.



H.15 Establish the part permission conditions that will allow the scanned barcode to continue the assembly. This rung can become complex depending upon the complexity of the Lookup that is being performed.



H.16 Setup the trigger to establish In Process. This includes the replacing of the REPLACE* PLC tag with the appropriate machine sequence step.



H.17 Setup the trigger to establish Part Processed. This includes the replacing of the REPLACE* PLC tag with the appropriate machine sequence step.

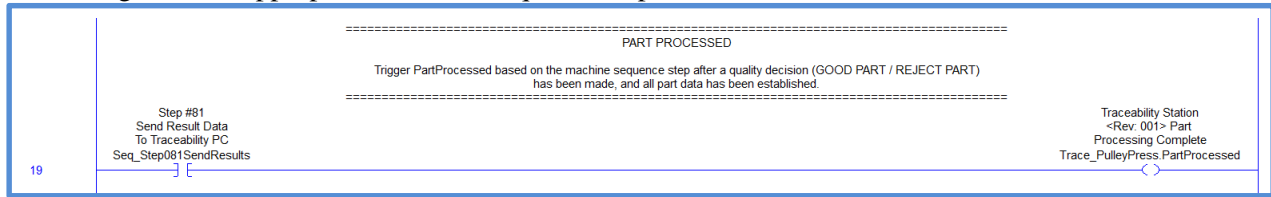


Figure 17: Enable Part Processed

H.18 Setup the Results Data Name Array

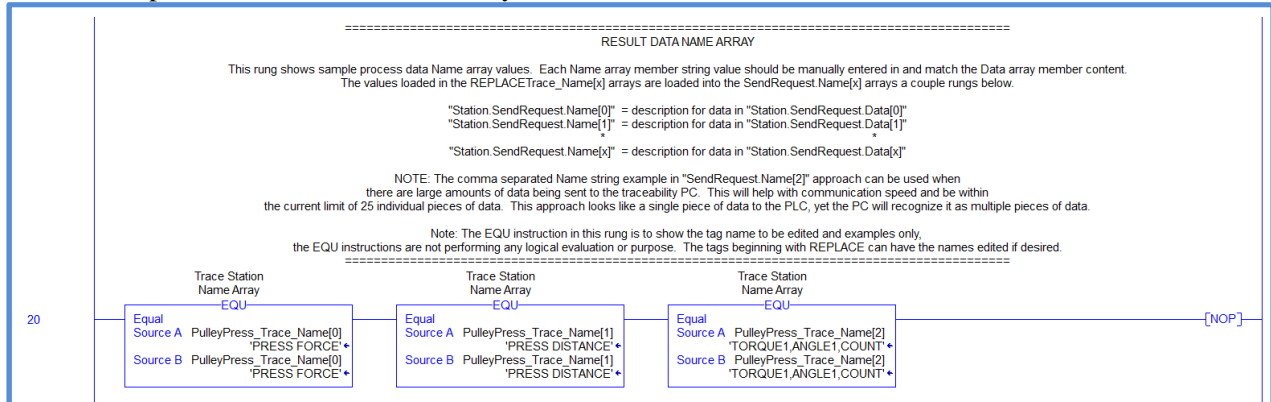


Figure 18: Results Data Name Array Setup

H.19 Setup the Results Data Array

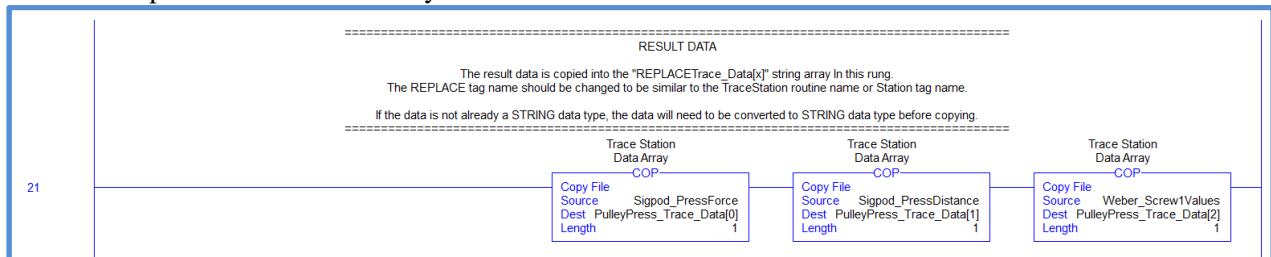


Figure 19: Results Data Array Setup

H.20 Setup the LENGTH of the data being sent to the Trace PC.

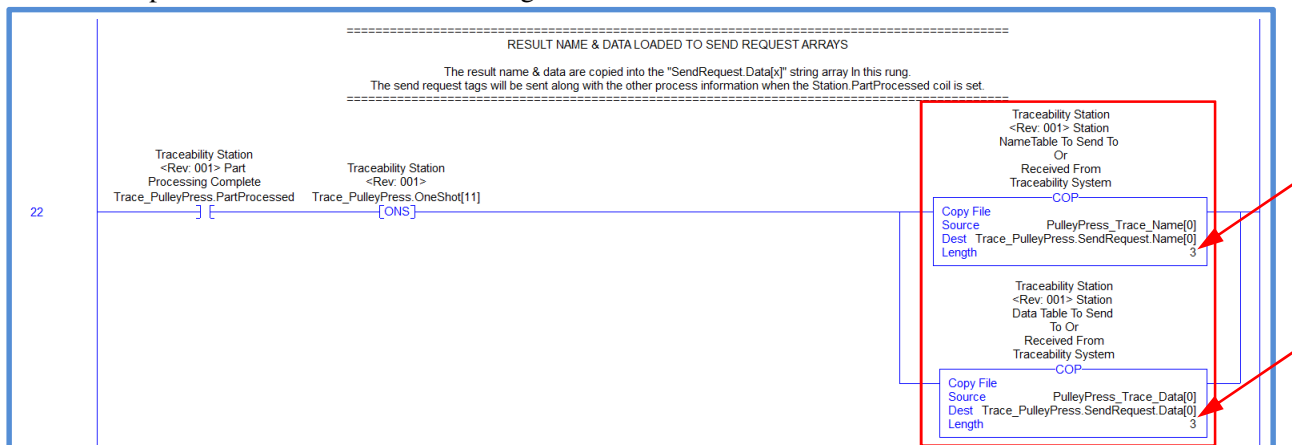
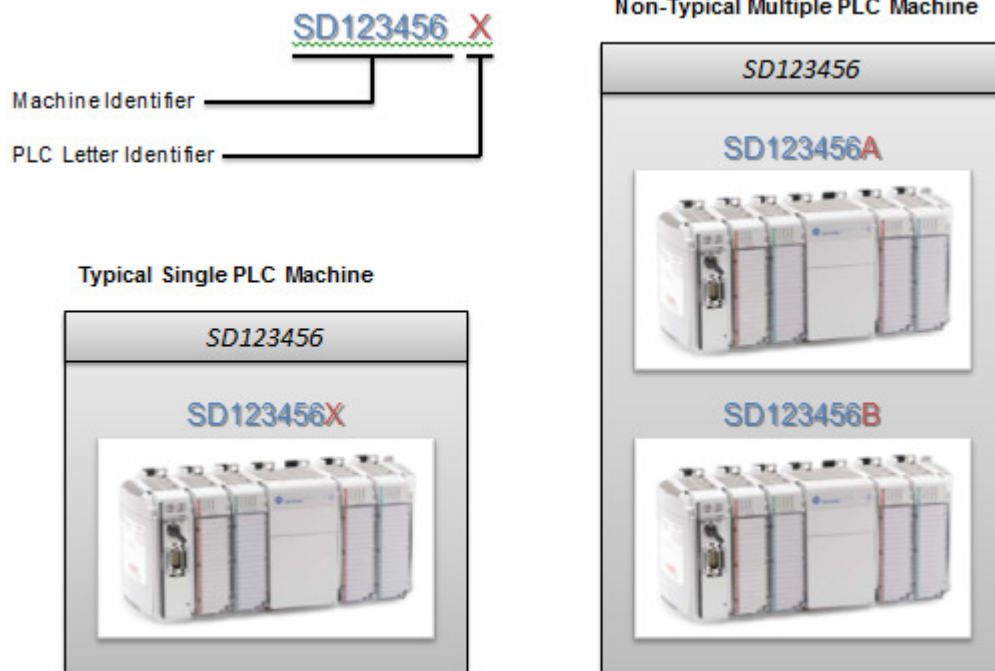


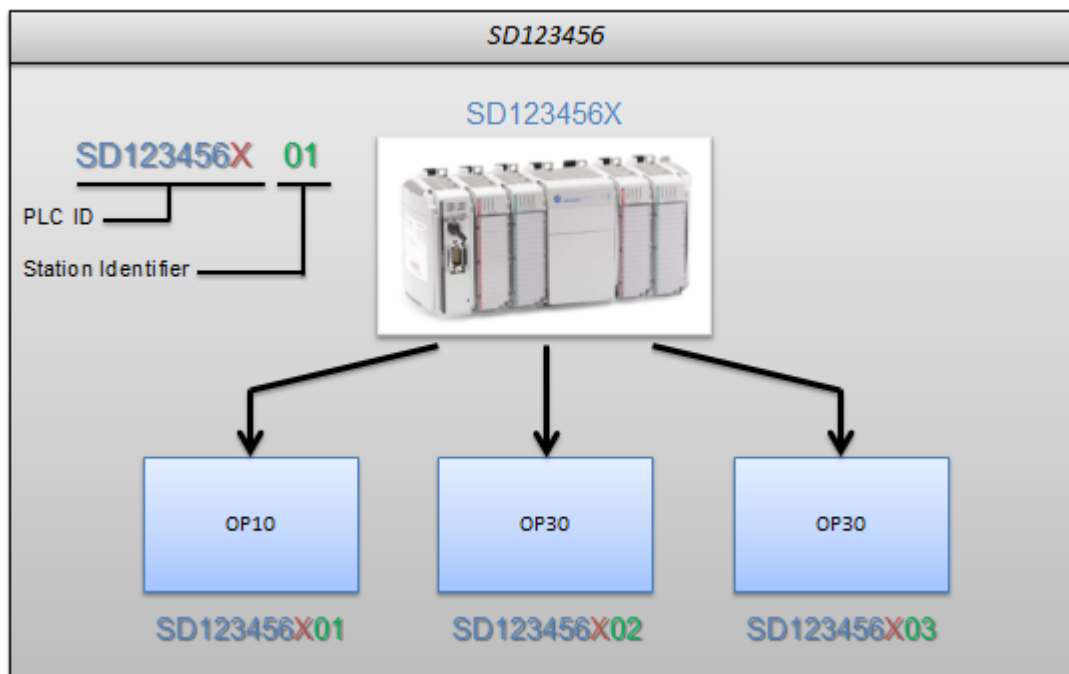
Figure 20: Data Length Setup

I. PLCID Naming Conventions

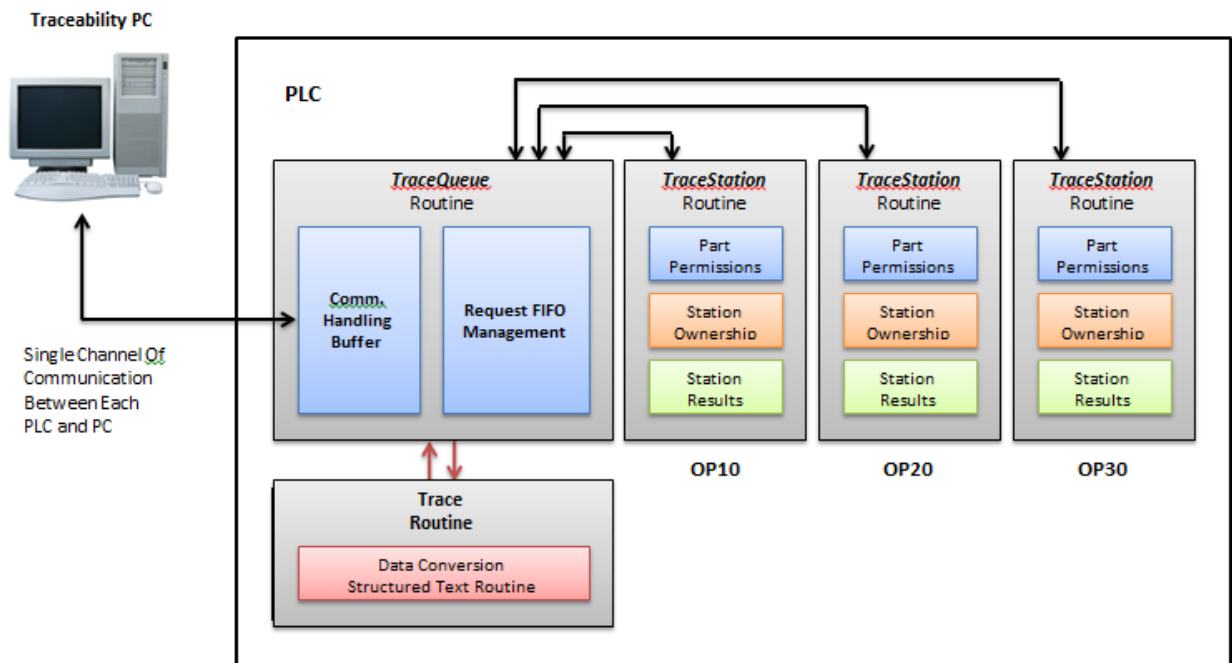


J. StationID Naming Conventions

Single PLC Controlling Multiple Stations

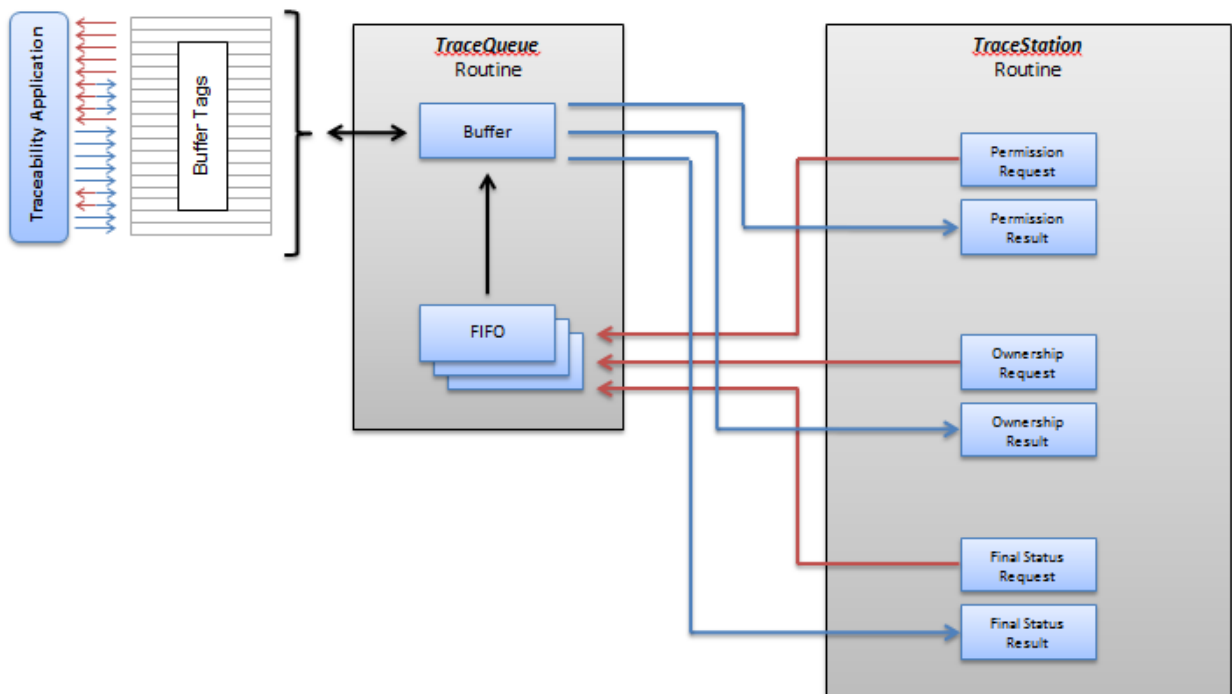


K. PLC Routine Structure



L. Data Flow within PLC

Data Flow Within PLC



M. Network Architecture

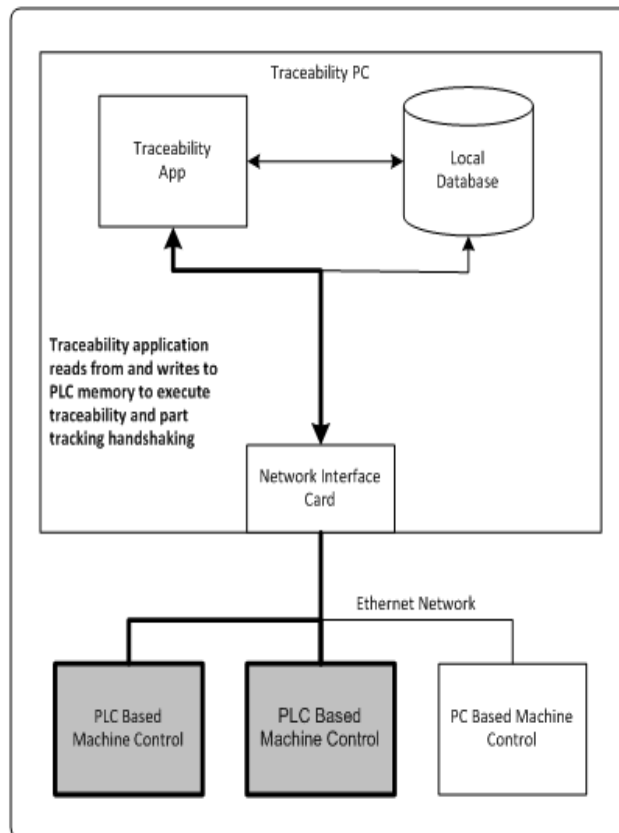


Figure 1 - Communication scheme to PLC based machine control systems

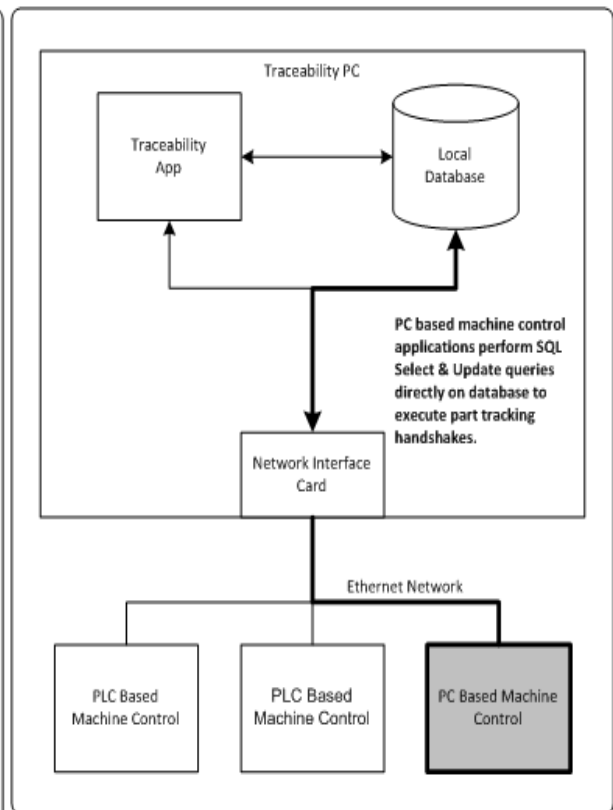


Figure 2 - Communication scheme to PC based machine control systems

RECORD OF REVISIONS

Revision #	Date	Section	Description
001	13JA16	All	Re-write of Specification from previous Guideline
002			
003			
004			
005			
006			
007			
008			
009			
010			
011			
012			
013			
014			
015			
016			
017			
018			
019			
020			