PlantConnect
Device Driver Configuration

Table of Contents

1. Device Driver

1.1. Overview

Each device driver is represented by a dll file. Name of the dll file is same as name of the protocol. So if protocol name is OPC, dll name will be OPC.dll.

There are 2 parts in the configuration of device driver in PlantConnect

1. Set protocol parameters for the driver. This is done on 'Device Configuration' screen
2. Set driver specific information for each variable. This is done on 'Variable Configuration' screen. There are 3 main attributes of a variable that are related to driver – **tag name**, **start address** and **data type**.

1.2. Drivers List

1.2.1. Drivers using Ethernet:

1. OPC
2. Euromap
3. ModbusTCPMaster
4. EtherNet/Ip

For all updated drivers list refer Chapter 2.

1.2.2. Drivers using Serial Communication:

i. ModbusRTUMaster
ii. MC3F
iii. MC4
iv. SelogicaDriver
v. Fanuc
vi. GSI
vii. Titrator

For all updated driver list refer Chapter 3.

1.2.3. Drivers using HTTP protocol for Communication

i. MTConnect

2. Drivers using Ethernet

2.1. OPC

It is DCOM based protocol.
For a number of devices that support OPC protocol. Example, B & R Controller

Protocol Document referred – opcda205A_cust.doc
Version – OPC Data Access (OPC DA) Customer Interface Version 2.05A

Protocol Parameters:
Mandatory Parameters:

1. **DAServer –** Full qualified name of OPC server or CLSID(if OPC server running on a different machine than DAS).
2. **DAServerType –** OPC Server type – LOCAL / REMOTE. 'Local' means the OPC server is running on same machine as DAS. 'Remote' means the OPC server is running on a different machine in the network.
3. **ServerMachine –** OPC Server Machine name or IP address. Needed only if OPC server is REMOTE (running on a different machine than DAS)

Optional Parameters:

4. **Redundant_DAServer –** Full qualified name of Redundant OPC server or CLSID(if OPC server running on a different machine than DAS).
5. **Redundant_DAServerType –** Redundant OPC Server type – LOCAL / REMOTE.
6. **Redundant_ServerMachine –** Redundant OPC Server Machine name or IP address. Needed only if OPC server is REMOTE (running on a different machine than DAS).
7. **DeviceStatus –** It is not mendatory paramteter. It is used to set device status by reading tag name value. Tag name should be provided whose value will decide the device status.
   Status can be either of following.
   If value read is 0 = Connected
   If value read is 1 = Calibration
   If value read is 2 = Maintenance
   if other than above value is received then device status should be Connected.
8. **MaxItems –** Maximum variables to be read from device in one attempt. It should be between 1 and 512. If not defined or value is outside the range, it is taken as 512.

Example -
DAServer = Softing.OPCToolboxDemo_ServerDA.1

DAServerType = REMOTE
ServerMachine = 10.6.10.55
Redundant_DAServer = Softing.OPCToolboxDemo_ServerDA.1
Redundant_DAServerType = REMOTE
Redundant_ServerMachine = 10.6.10.10
DeviceStatus = Channel_1.Device_1.analog
MaxItems = 20

Variable Configuration

1. **Tag Name –** Required (fully qualified parameter name used by OPC server)
2. **Start address –** Not required
3. **Data type** – You can select appropriate data type of variable from drop down list. The data type list display as mentioned below:
   - Short(2 bytes (-32768 – 32767)) canonical type 2 (VT_I2) - [30]
   - Long (4 bytes (-2147483648 – 2147483647)) canonical type 3 (VT_I4) - [31]
   - Float (4bytes) canonical type 4 (VT_R4) - [32]
   - Double (64bit) canonical type 5 (VT_R8) - [33]
   - Byte (1 byte (0-255)) canonical type 17 (VT_UI1) - [34]
   - Word (2 bytes - (0-65535)) canonical type 18 (VT_UI2) - [35]
   - Dword(4bytes (0 – 4294967295)) canonical type 19 (VT_UI4) - [36]

Example -
Tag Name = time.local.second
Data Type = Short(2 bytes (-32768 – 32767)) canonical type 2 (VT_I2) - [30]

2.2. Euromap

File based protocol
For a number of devices that support Euromap protocol. Example, Battenfeld injection moulding machines

Protocol Document referred - eu63.pdf
Version – Euromap 63 version 1.05a

Euromap is a file based protocol. Requests are sent to the machine by creating various files. The machine sends response also by creating various files. All these files reside at a location on file system that is accessible to both the machine and PlantConnect DAS.

In the network, there is one file giving basic information of all the Euromap devices present in that network. Name of this file is **Machine.ini**. This file is used to obtain a list of all devices (machines) along with the location of each machine's session path entry. Keep this file in PlantConnect DAS root folder (where pls.exe is present)

Sample Machine.ini file

```
[MACHINE]
1=MACHINE_1
2=MACHINE_2

[MACHINE_1]
SESSIONPATH =\\SV1\INTERFACE\MACH1
MAXSESSION=5
IPADDRESS=128.123.200.103

[MACHINE_2]
SESSIONPATH =\\SV2\INTERFACE\MACH2
MAXSESSION=6
IPADDRESS=128.123.200.104
```

Machine.ini file explanation
Section [MACHINE] gives names of all machines present in network

The file also contains one section for each machine. This section has following keys –
SessionPath – Network path where all request and response files for this machine should be kept

MaxSession – Maximum number of sessions that this machine will manage simultaneously
IPAddress – IP address of the machine

Protocol Parameters –
Following protocol parameters should be defined for Euromap protocol

1. **MachinName** – Name of the Machine as specified in machine.ini file. (For Example MACHINE_1)
2. **TimeOut** – This is the wait time in milliseconds for the machine to write Response File after a request is received. PlantConnect DAS will wait for this time to get a response file.

Example
MachineName=MACHINE_1
TimeOut=20000

Variable Configuration -

1. **TagName -** Fully qualified parameter name supported by specific machine.
2. **Start address** – Not required
3. **End address** – Not required

Example -
Tag Name = ActTimCyc

2.3. ModbusTCPMaster

Modbus is probably the most widely used protocol in process control industries today. Modbus allows communication to occur between slave devices such as PLCs and Instrumentation and Master devices such as computers or DCSs.

Used for any controller supporting Modbus TCP protocol.

Protocol Parameters:
Mandatory Parameters:

1. **PortNumber -** It is the TCP/IP port for Modbus TCP communications. E.g. 502, 503 etc.
2. **SlaveID** – Modbus slave ID. Starts from 1 to 255.
3. **HostAddress** – It is the IP address of the slave or gateway will connect to. E.g. 10.6.10.255
4. **TimeOut -** Communication timeout in milliseconds.

Optional parameters:

5. **HighIn** – It is required in case analog signals are coming from device & conversion to engineering units is required. It is the Highest Input value.
6. **LowIn –** It is the Lowest Input value. If you provide these values(High In & Low In) you also need to provide High Out & Low Out values on variable add page.
7. **CalculationType –** This is not mandatory parameter. This is used in case we need to apply scaling factors to variables as per requirement. In this case, this parameter should be set to 1. If not set then standard calculation is applied to variable.
8. **ConnectOnPolling –** This parameter is not mandatory. This change is added specifically to resolve issue of device status fluctuation at Atul Valsad site. Value of this parameter is either '0' or '1'. if this value is 1 then it means, Connect socket for each polling frequency in order to read all variable values. If not set then its value will be 0 as default.
9. **IsLossyConnect –** This parameter is not mandatory. Some devices are closing the connection after each request-response. For this behavior, we need to connect the device before each read command. For such devices, we need to set this flag as "1". By default it will keep connection open and will use same connection.
   Note: For lossy connection, This flag should set to all devices which are connected on same Host and port.
10. **IsReadRetry** – If read is failed and this flag is 1, try to read it 3 times.
11. **WriteFunctionCode –** If write on modbus single-single register using function code 6 then set **WriteFunctionCode = 1 otherwise it is by-default 0.**

Below mentioned parameters require only in case of 'Calibration Status' required for device.

12. **ClientName –** This is not mandatory parameter. This is used in case reading calibration status for specific client. It holds client name either of following

"FM", "Thermofisher", "Iceshia", "Spectrum".

13. **DeviceType –** This parameter is not mandatory and used for FM client. This helps for calculating calibration device status. It holds value as Forbes Marshall device type (G=GCEM, D for DCEM)
14. **PlantStatusAddress –** This parameter is not mandatory and used for FM client. This helps for calculating maintenance device status. It holds value as Modbus register address whose value will be used to decide maintenance status.
15. **CalStatusAddress –** This parameter is not mandatory and used for FM/Thermofisher/Iceshia/Spectrum client. Following are the values for this variable when used with different client.

Customer specific details for calibration related parameters is mentioned in below section of 'Calibration Parameter Details'.

General Example -

PortNumber = 502
HostAddress = 10.6.10.6
TimeOut = 2000
SlaveID = 1

Variable Configuration

1. **Tag Name –** Not required for device connection but tag name is a compulsory field on UI, so keep same as variable name.
2. **Start address** – Modbus register address minus offset. e.g. if register address is 40001 and offset is also 40001, then start address should be 0.
3. **Data type** – Here it is the combination of Modbus function code and data type of a variable. User will see the list as shown below -

   On **variable** page '**Type**' as '**Analog**' then it will show below list -

   - 16 bit Unsigned Integer(3,Holding Register)
   - 32 bit Signed Integer(3,Holding Register)
   - 16 bit Unsigned Integer(4,Input Register)
   - 32 bit Signed Integer(4,Input Register)
   - 32 bit Float(3,Holding Register)

- 32 bit Float(4,Input Register)
- 32 bit Float Reverse (little endien format)(3,Holding Register)
- 32 bit Float Reverse(4,Input Register)
- 64 bit Double(3,Holding Register)
- Double(4,Input Register)
- Double Reverse(3,Holding Register)
- Double Reverse(4,Input Register)
- 32 bit Signed Integer Reverse(3,Holding Register)
- 32 bit Signed Integer Reverse(4,Input Register)
- 16 bit Signed Integer (3,Holding Register)
- 16 bit Signed Integer (4,Input Register)
- Data long 3 words (3,Holding Register)
- Data long 3 words (4,Input Register)
- Data long 3 words Reverse(3,Holding Register)
- Data long 3 words Reverse(4,Input Register)

On variable page 'Type' as 'Digital' then it will show below list -

For input and output coil use below 2 option -

- Digital (1,Output Coil)
- Digital (2,Input Coil)

If converting **16 bit Unsigned Integer(3,Holding Register)** register value to digital use below option as per their bit position -

To read specific bit value of register(say 0^{th} bit, 1^{st} bit or 2^{nd} bit and so on from 0 to 15), below data type need to be configured.

- Zeroth Bit(LSB)
- First Bit(LSB)
- Second Bit(LSB)
- Third Bit(LSB)
- Fourth Bit(LSB)
- Fifth Bit(LSB)
- Sixth Bit(LSB)
- Seventh Bit(LSB)
- Eighth Bit(LSB)
- Ninth Bit(LSB)
- Tenth Bit(LSB)
- Eleventh Bit(LSB)

- Twelth Bit(LSB)
- Thirteenth Bit(LSB)
- Fourteenth Bit(LSB)
- Fifteenth Bit(LSB)

On variable page 'Type' as 'String' then it will show text filed -

- **Add value in text field of 'Data Type' greater than 100(16 bit Unsigned Integer(3,Holding Register))**

  - To read string from registers, you have to set end address greater than 100 depends on length of string. Each register of modbus is of 16bits(2 bytes) and each register contains 2 characters in it. Thus when you want to read string of say 10 characters, you need to read 5 registers data at a time. Hence End address in this case will be 105. (I.e. 105-100 = 5 registers will be read)

  Note – At a time, variables whose block no and function code is same, one request will be send for these variables.

  Example -
  TagName = Paramter1
  Start Address = 2 (register address = 40003 and offset = 40001)
  Data type = **16 bit Unsigned Integer(3,Holding Register)**

4. High Out and Low Out

These fields are required only if protocol parameter 'High In' and 'Low In' are defined. These values are nothing but the converted High & Low Output.

Formula To calculate factors:
EU_Reading = ((ModbusReading – LowIn) / (HighIn – LowIn)) * (HighOut – LowOut) + LowOut.

Formula To calculate factors when CalculationType = 1:
EU_Reading = (LowOut * ModbusReading) + HighOut

'Calibration Parameter Details':

A. ClientName = FM

#Protocol Parameters

ClientName=**FM**

CalStatusAddress = <StartAddress> (Value on this address is converted to hex. If last two bits(RHS) of hex value are **not** '00' then set calibration flag

else if Device type is "G" or "D" AND first two bits of hex value are **not** "00" set purging flag(M flag))

PlantStatusAddress= <Start address> (Value on this address is converted to hex.

1) If Device type is "D" AND hex value is **not** "0005" AND last two bits of hex value are **not** "05" then set M flag)

2) If Device type is "G" AND hex value is **not** "000A" then set M flag)

#Trigger Calibration

ANALOG variable, if write value 1 it starts the calibration.

B. ClientName = Thermofisher

This parameter is added to read calibration status of device. It holds 3 values separated by delimiter "~" as follows

1st parameter is start address to read holding register value.

2nd parameter is zero bit number- read value from given start address and read its zero th bit. If it is 1 then set device is in calibration mode.

3rd parameter is span bit number – read value from given start address and read its span th bit. If it is 1 then set device is in calibration mode.

If one of the bits received is 1 then device is in calibration mode .

C. ClientName = Iceshia

This parameter is used to read calibration status of device. This change in specific to Iceshia client. Its holds two start addresses seperated by delimeter '~'. It reads values of these addresses and if one of the values received is 1 then its sets device status in calibrated mode.

D. ClientName = Spectrum

This parameter is used to read calibration status of device. This change in specific to Spectrum client. Its holds start address. Depends on value received on this start address, device status will be set. If value received is 0 then set device status as 0 . otherwise set its status as 1(Device is in calibration mode).

E. ClientName = ABB

#Protocol Parameters

ClinentName = **ABB**
CalStatusAddress = <StartAddress> (if Value = 1 then set calibration flag othewise 0)

#Trigger Calibration
Digital variable, if write value 1 it starts the calibration.

F. ClientName = HNL
#Protocol Parameters
ClinentName = **HNL**
CalStatusAddress = <StartAddress> (if Value = 1 then set calibration flag othewise 0)

#Trigger Calibration
ANALOG variable, if write value 1 it starts the calibration.

G. ClientName = FMEM
This is used to read calibration status of device in case of Emerson. This change is for FM – Emerson client. It needs start address of 2 variable or key CalStatusAddress and PlantStatusAddress. Depending on the values received on these addresses, device's status for Calibration or Maintenance is set.

For Calibration flag- if CalStatusAddress values are 11 or 12 or 14 or 15 or 1 or 2 or 4 or 5 or 10, then **C** flag is set

For Maintenance flag- For maintenance flag, values on PlantStatusAddress are converted to binary automatically. Note :Analog value is only given on this address(When Maintenance is initiated). If bit0 =0 AND (bit1=1 or bit2=1 or bit3=1) AND CalStatusAddress value =0 then **M** flag is set. For example- Analog values = 8,4,2, etc.

#Protocol Parameters
ClientName = **FMEM**
CalStatusAddress = <Start address> (if Value = 11 or 12 or 14 or 15 or 1 or 2 or 4 or 5 or 10 then set calibration flag othewise 0)
PlantStatusAddress = <Start address> (If Value when converted to binary, bit0 ==0 AND (bit1==1 or bit2==1 or bit3==1) AND Calibration flag is not set then set maintenance flag otherwise 0)

#Trigger Calibration
DIGITAL variables, we require 2 digital variables. if write value 0 on 1st variable and 1 on 2nd variable, then it starts the calibration.

(for ex- After 3 minutes) if write value 1 on 1st variable and 0 on 2nd variable, then it stops the calibration.
#Trigger Maintenance
DIGITAL variable, if write value 1 it starts the maintenance.
(for ex- After 3 minutes) if write value 0 it stops the maintenance.
Note: We have different conditions for calibration and maintenance triggers as mentioned above.

H. ClientName = FMONGC
#Protocol Parameters
ClientName = **FMONGC**
CalStatusAddress = <StartAddress> (if Value = 1 then set calibration flag otherwise 0)
#Trigger Calibration
ANALOG variable, if write value 1 it starts the calibration.

I. ClientName = FMGCEM
#Protocol Parameters
ClientName = **FMGCEM**
CalStatusAddress = <StartAddress> (Value on this address is converted to hex. If last two bits(RHS) of hex value are **not** '00' then set calibration flag
else if Device type is "G" or "D" AND first two bits of hex value are **not** "00" set purging flag(M flag))
PlantStatusAddress= <Start address>(Value on this address is converted to hex. 1) If Device type is "D" AND hex value is **not** "0005" AND last two bits of hex value are **not** "05" then set M flag)
2) If Device type is "G" AND hex value is **not** "0004" then set M flag)
#Trigger Calibration
ANALOG variable, if write value 1 it starts the calibration.

J. ClientName = FMSIEMENS
#Protocol Parameters
ClientName = **FMSIEMENS**
CalStatusAddress = <StartAddress> (if Value = 5 then set Zero calibration, if value = 6 set Span calibration otherwise 4 for normal measuring)
WriteFunctionCode=1 (As we need to write single-single register, I.e, for 0 min C001address, for 0 min C004 address)
#Trigger Calibration
For Span Calibration, use ANALOG variable. If write value 1 it starts the Span calibration.
For Zero Calibration, use DIGITAL variable. If write value 1 it starts the Zero calibration.

2.4. OPC UA

For a number of devices that support OPC UA protocol. Example, B & R Controller (X20 CPU with the Ethernet TCP / IP port)**.**

Protocol Parameters:

1. **ServerURL –** Server endpoint URL of OPCUA Server E.g. opc.tcp://localhost:51510/Server
2. **TimeOut –** maximum time to give the response. In Milliseconds.
3. **SecurityMode –** OPC Server levels of security. None, Sign, SignAndEncrypt
4. **AuthenticationMode -** OPC Server user authentication mode. Anonymous, User
5. **UserName –** User name to authenticate OPC UA server. Required if AuthenticationMode is User
6. **Password –** Password to authenticate OPC UA server. Required if AuthenticationMode is User
7. **NameSpaceIndex –** It is unique identifier node id(A namespace in OPC UA is like a container for node ids).

E.g. Node_ID should like this "**ns=6**;s=0:Channel_1.Device_1.BOD"
ns=6 is a namespace address which is defined in this Protocol Parameter.

Example –
ServerURL= opc.tcp://localhost:51510/Server
TimeOut = 600000
SecurityMode=None
AuthenticationMode=User
UserName=AIPL
Password=AIPL12*
NameSpaceIndex=6

For Variable configuration

Tag Name:
Required. This is the name of the TAG configured in the OPC UA server.

Start Address:
Required. It requires to provide the index number of array like 0,1,...n. If variable is not reading from array then defined it as -1.

Example:
Node_ID should like this "ns=6;s=0:Channel_1.Device_1.BOD" it is not array variable.
ns=6 is a namespace address which is add in Protocol Parameter**.**
s= 0:Channel_1.Device_1.BOD is a unique address of that variable which is add it in Tag Name.
Some times instead of s there should be I.

StartAddress=-1
TagName= 0:Channel_1.Device_1.BOD

Supported Data Types
Analog
String
Digital (Boolean)

Data type for Analog:

- Short(2 bytes (-32768 – 32767)) canonical type 2 (VT_I2) - [30]
- Long (4 bytes (-2147483648 – 2147483647)) canonical type 3 (VT_I4) - [31]
- Float (4bytes) canonical type 4 (VT_R4) - [32]
- Double (64bit) canonical type 5 (VT_R8) - [33]
- Byte (1 byte (0-255)) canonical type 17 (VT_UI1) - [34]
- Word (2 bytes - (0-65535)) canonical type 18 (VT_UI2) - [35]
- Dword(4bytes (0 – 4294967295)) canonical type 19 (VT_UI4) - [36]

Note: No need to provide any data type for String and Digital Type of variable.

2.5. STEP7(Siemens)

Siemens TCP/IP PLC are supported.

Protocol Parameters

1. **PortNumber -** It is the TCP/IP port for STEP7 TCP communications. E.g. 502, 503 etc.
2. **HostAddress** – It is the IP address of the slave or gateway will connect to. E.g. 10.6.10.255
3. **TimeOut –** Communication timeout in milliseconds.
4. **BufferSize –** write size of buffer

Example –

PortNumber = 502
HostAddress = 10.6.10.6
TimeOut = 2000
BufferSize = 2048

For Variable configuration
TagName is same as variable name.

Supported Data Types
Analog
String
Digital (Boolean)

Start Address

- **Analog, Digital, String :**

Set start number of character from given string

End Address

- **Analog:**

30: Integer
31: Float

- **Digital :**

Set 1

- **String :**

Number of characters to be read

For Example :

received String is :: Eáp0CÝ00BúA01AIPL (0 is nothing but null)

ANALOG :

Float:

StartAddress : 0
EndAddress : 31

StartAddress : 4
EndAddress : 31

int:

StartAddress : 8
EndAddress : 30

StartAddress : 10
EndAddress : 30

DIGITAL:

StartAddress : 12
EndAddress : 1

STRING:

StartAddress : 13
EndAddress : 4

2.6. EtherNet/IP

Protocol Parameters:

1. PortNumber - It is the TCP/IP port for TCP communications. E.g. 502, 503 etc.
2. HostAddress – It is the IP address of the slave or gateway will connect to. E.g. 10.6.10.255
3.TimeOut – Communication timeout in milliseconds.
4.PLCType – Type of PLC with whom protocol is connected to get data.
PLC Type: LGX, PLC5, SLC500, Unknown

Example –

PortNumber = 502
HostAddress = 10.6.10.6
TimeOut = 2000
PLCType = 0 [if LGX Type PLC is connected then use 0, for all other types use 1]

For Variable configuration
TagName of variable should be same as TagName configured in PLC to read data from assigned register address.

Supported Data Types
Analog

2.7. DigitalPulseDriverRTU

This driver uses ModbusRTU as a base and added some addiotional functionality on top of it. Thus below is the configuration of same.

Protocol Parameters

1. **CommunicationType –** This parameter decided the communication type to be used. Either TCP or RTU. According to the communication type other protocol parameters are to be set

2. **DigitalPulsePollInMillisec –** This is the polling frequency of dll thread which calls and get data from ModbusRTU dll. This polling freqquency should be in milliseconds and minimum time to be configured as 100ms.

3. **SlaveID –** Modbus Slave ID. starts from 1 to 255.
4. **TimeOut -** maximum time to give the response. In Milliseconds.

5. **PortNumber –** Port number to be used for communication. Com port. e.g. Com1, Com2 etc in case of RTU (serial communication) or actual port number in case of TCP
6. **BaudRate -** No of times/sec a signal changes state. Required for RTU
7. **DataBits -** No of bits used to represent one character of data. Required for RTU
8. **StopBits -** sent with the data but not part of the data. Required for RTU
9. **Parity -** no of bits with value. None = 0 Even = 2 Odd = 1. Required for RTU
10. **HostAddress** - Host Machine Address. Required for TCP'
11. **IsLossyConnect –** This parameter is not mandatory. Some devices are closing the connection after each request-response. For this behavior, we need to connect the device before each read command. For such devices, we need to set this flag as "1". By default it will keep connection open and will use same connection. **Note: For lossy connection, This flag should set to all devices which are connected on same Host and port.**
12. **IsReadRetry** – If read is failed and this flag is 1, try to read it 3 times.
13. **WriteFunctionCode –** If write on modbus single-single register using function code 6 then set **WriteFunctionCode = 1 otherwise it is by-default 0**

Note – make sure if your TimeOut protocol parameter should be less than or equal to DigitalPulsePollInMillisec. Otherwise it won't work properly. Minimum TimeOut value can be set as 100.

Example
PortNumber = COM1
BaudRate = 9600
DataBits = 8
Parity = 0
StopBits = 1
TimeOut = 100
DigitalPulsePollInMillisec = 100
SlaveID = 1

Variables to be configured in PlantConnect

Sr.	Parameter	Type	Start Address	Logging	Remark
1.	MouldOpenPulse	Digital	Register address whose value to be read	Don't log	This will read value from device at given start address. And based on end address configured , it will read bit value. **1)If tag Name = "mouldPulse,30", then modbus request will be send for holding register . In this case we need to set End Addredd field to read specific bit value from that. If end address is 2 then it will read (2-1)= 1st bit of byte(LSB bit starts from 0).** **2) If Tag name= "mouldPulse,42", then modbus request will be sent for output coil. In this it will directly give me pulse value as this is digital type variable.**
2.	Actual Cycle Time	Analog	-1	log	Calculated in dll Time difference in seconds between occurrences of two consecutive 1 values of Pulse parameter.
3.	Cycle Count	Analog	-2	Log all values when this value changes	Calculated in dll. Increment by 1 for every valid cycle.
4.	Std. Cycle Time	Constant			Constant for given product. Will be set thru 'product

Sr.	Parameter	Type	Start Address	Logging	Remark
		(writable)			change' data entry form. Otherwise user should write this value when a new product is setup
5.	Cavities	Constant (writable)		Don't log	Constant for given product. Will be set thru 'product change' data entry form. Otherwise user should write this value when a new product is setup
6.	Parts Produced	Calculated		log	= Cavities in each cycle

Note: - While configuring this driver , please make sure that not to configure standard modbusRTUMaster protocol on same COM port As both the drivers have different way of working.

2.8. MCProtocol

Protocol Parameters

- **PortNumber -** It is the TCP/IP port for MCProtocol TCP communications. E.g. 502, 503 etc.

- **HostAddress** – It is the IP address of the slave or gateway will connect to. E.g. 10.6.10.255
- **TimeOut –** Communication timeout in milliseconds.
- **BufferSize –** write size of buffer
- **PCNo** – Enter PC Number. Specify the PC No. that are set with the parameters for the access target network module. The range is 1 to 120.
- **NetworkNo** -Specify the NetworkNo. Specify the **NetworkNo**. that are set with the parameters for the access target network module. The range is 1 to 239.
- **MonitorTimer -** Set the wait time up to the completion of reading and writing processing. Set value in seconds.
- **RequestDestIO -**Provided By users
- **RequestDestModule -**The range is 1 to 31.

Specification fo Point 8 and 9

Specify these numbers when an access target is as shown below. • Multidrop connection station • CPU module on multiple CPU system • CPU module on redundant system, CC-Link IE Field Network remote head module

Example –

PortNumber = 502
HostAddress = 10.6.10.6
TimeOut = 2000
BufferSize = 2048
PCNo=255
NetworkNo= 00
MonitorTimer = 4
RequestDestIO = 03FF
RequestDestModule =00

For Variable configuration

- **TagName : <Variable Name>,<Device code >,<Block Number>**
- **Example :We read Data Register so Device Code is 'D'**
  - **Variable1,D,1**

Supported Data Types

- Analog
- String
- Digital (Boolean)

Start Address

5. **Analog, Digital, String :**
   Set start number of character from given string

End Address

6. **Analog:**
   30: Unsigned Integer
   31: Signed Integer
   32: Unsigned Long
   33: Signed Long
   34: Float/Double

7. **Digital :**
   Set 0 to 15
8. **String :**
   Number of Word to be read.
   Word is a 16 bit Data Type.

Sample Request and Response Example :

- Request:500000FF03FF000018001004010000D*0000000004
  - 5000 - SubHeadrer
  - 00 - Network numbefr
  - FF - PC number
  - 03FF - Request destination module I/O No
  - 00 - Request destination module station No
  - 0018 - Request data length*1
  - 0010 - Monitoring timer
  - 0401 - Command
  - 0000 - SubCommand
  - D* - Device code
  - 000000 - Head device number(Start Address)
  - 0002 - Number of device points(Number of bytes to read)

- Response :D00000FF03FF00000C0000414211D2

- D000 - SubHeadrer
- 00 - Network numbefr
- FF - PC number
- 03FF - Request destination module I/O No
- 00 - Request destination module station No
- 000C - Response data length*1
- 0000 - END Code
- 414211D2 - Data

Example Of Variable Configuration

ANALOG:

int:

StartAddress : 0
EndAddress : 31

StartAddress : 4
EndAddress : 31

Unsigned int:

StartAddress : 8
EndAddress : 30

StartAddress : 10
EndAddress : 30

Unsigned long:

StartAddress : 0
EndAddress : 32

StartAddress : 99
EndAddress : 32

long:

StartAddress : 44
EndAddress : 33

StartAddress : 55
EndAddress : 33

DIGITAL:

StartAddress : 12
EndAddress : 0
StartAddress : 12
EndAddress : 1
StartAddress : 12
EndAddress : 3

STRING:

StartAddress : 13

EndAddress : 4

3. Drivers using Serial Communication

Common Protocol Parameters
These parameters are needed for all RS232 drivers

1. **PortNumber –** Com port. e.g. Com1, Com2 etc
2. **BaudRate –** Baud rate. e.g. 9600, 19200 etc
3. **DataBits –** Data bits
4. **Parity –** Parity. Possible values 0=none , 1=odd, 2=even, 3=mark, 4=space
5. **StopBits –** Stop bits
6. **TimeOut –** Communication timeout in milliseconds

3.1. ModbusRTUMaster

For any controller supporting Modbus RTU protocol

Protocol Parameters:
Mandatory parameters:

1. All RS232 protocol parameters plus following parameters
2. **SlaveID** – Modbus slave ID. Starts from 1 to 255.

Optional parameters:

3. **HighIn** – It is required in case analog signals are coming from device & conversion to engineering units is required. It is the Highest Input value.
4. **LowIn –** It is the Lowest Input value. If you provide these values(High In & Low In) you also need to provide High Out & Low Out values on variable add page.
5. **CalculationType –** This is not mandatory parameter. This is used in case we need to apply scaling factors to variables as per requirement. In this case, this parameter should be set to 1. If not set then standard calculation is applied to variable.
6. **ConnectOnPolling –** This parameter is not mandatory. This change is added specifically to resolve issue of device status fluctuation at Atul Valsad site. Value of this parameter is either '0' or '1'. if this value is 1 then it means, Connect socket for each polling frequency in order to read all variable values. If not set then its value will be 0 as default.
7. **IsLossyConnect –** This parameter is not mandatory. Some devices are closing the connection after each request-response. For this behavior, we need to connect the device

before each read command. For such devices, we need to set this flag as “1”. By default it will keep connection open and will use same connection.

Note: For lossy connection, This flag should set to all devices which are connected on same Host and port.

8. **IsReadRetry** – If read is failed and this flag is 1, try to read it 3 times.
9. **WriteFunctionCode –** If write on modbus single-single register using function code 6 then set **WriteFunctionCode = 1 otherwise it is by-default 0.**

Below mentioned parameters require only in case of 'Calibration Status' required for device.

10. **ClientName –** This is not mandatory parameter. This is used in case reading calibration status for specific client. It holds client name either of following
11. “FM”, “Thermofisher”, “Iceshia”, “Spectrum”.
12. **DeviceType –** This parameter is not mandatory and used for FM client. This helps for calculating calibration device status. It holds value as Forbes Marshall device type (G=GCEM, D for DCEM)
13. **PlantStatusAddress –** This parameter is not mandatory and used for FM client. This helps for calculating maintenance device status. It holds value as Modbus register address whose value will be used to decide maintenance status.
14. **CalStatusAddress –** This parameter is not mandatory and used for FM/Thermofisher/Iceshia/Spectrum client. Following are the values for this variable when used with different client.

Customer specific details for calibration related parameters is mentioned in above section of 'Calibration Parameter Details' of ModbusTCP protocol.

General Example -

PortNumber = COM1
BaudRate = 19200
DataBits = 8
Parity = 0
StopBits = 1
TimeOut = 6000
SlaveID = 1

Variable Configuration

1. **Tag Name –** Not required for device connection but tag name is a compulsory field on UI, so keep same as variable name.

2. **Start address** – Modbus register address minus offset. e.g. if register address is 40001 and offset is also 40001, then start address should be 0.
3. **Data type** – Here it is the combination of Modbus function code and data type of a variable. User will see the list as shown below -
   On **variable** page '**Type**' as '**Analog**' then it will show below list -
   - 16 bit Unsigned Integer(3,Holding Register)
   - 32 bit Signed Integer(3,Holding Register)
   - 16 bit Unsigned Integer(4,Input Register)
   - 32 bit Signed Integer(4,Input Register)
   - 32 bit Float(3,Holding Register)
   - 32 bit Float(4,Input Register)
   - 32 bit Float Reverse (little endien format)(3,Holding Register)
   - 32 bit Float Reverse(4,Input Register)
   - 64 bit Double(3,Holding Register)
   - Double(4,Input Register)
   - Double Reverse(3,Holding Register)
   - Double Reverse(4,Input Register)
   - 32 bit Signed Integer Reverse(3,Holding Register)
   - 32 bit Signed Integer Reverse(4,Input Register)
   - 16 bit Signed Integer (3,Holding Register)
   - 16 bit Signed Integer (4,Input Register)
   - Data long 3 words (3,Holding Register)
   - Data long 3 words (4,Input Register)
   - Data long 3 words Reverse(3,Holding Register)
   - Data long 3 words Reverse(4,Input Register)

   On variable page 'Type' as 'Digital' then it will show below list -
   For input and output coil use below 2 option -
   - Digital (1,Output Coil)
   - Digital (2,Input Coil)

   If converting **16 bit Unsigned Integer(3,Holding Register)** register value to digital use below option as per their bit position -
   To read specific bit value of register(say 0^{th} bit, 1^{st} bit or 2^{nd} bit and so on from 0 to 15), below data type need to be configured.
   - Zeroth Bit(LSB)
   - First Bit(LSB)

- Second Bit(LSB)
- Third Bit(LSB)
- Fourth Bit(LSB)
- Fifth Bit(LSB)
- Sixth Bit(LSB)
- Seventh Bit(LSB)
- Eighth Bit(LSB)
- Ninth Bit(LSB)
- Tenth Bit(LSB)
- Eleventh Bit(LSB)
- Twelth Bit(LSB)
- Thirteenth Bit(LSB)
- Fourteenth Bit(LSB)
- Fifteenth Bit(LSB)

On variable page 'Type' as 'String' then it will show text filed -

- **Add value in text field of 'Data Type' greater than 100(16 bit Unsigned Integer(3,Holding Register))**

- To read string from registers, you have to set end address greater than 100 depends on length of string. Each register of modbus is of 16bits(2 bytes) and each register contains 2 characters in it. Thus when you want to read string of say 10 characters, you need to read 5 registers data at a time. Hence End address in this case will be 105. (I.e. 105-100 = 5 registers will be read)

Note – At a time, variables whose block no and function code is same, one request will be send for these variables.

Example -
TagName = Paramter1
Start Address = 2 (register address = 40003 and offset = 40001)
Data type = **16 bit Unsigned Integer(3,Holding Register)**

4. High Out and Low Out

These fields are required only if protocol parameter 'High In' and 'Low In' are defined. These values are nothing but the converted High & Low Output.

Formula To calculate factors:

EU_Reading = ((ModbusReading – LowIn) / (HighIn – LowIn)) * (HighOut – LowOut) + LowOut.

Formula To calculate factors when CalculationType = 1:
EU_Reading = (LowOut * ModbusReading) + HighOut

3.2. MC3F

For Krauss Mafai Injection Moulding Machines

Protocol Document referred - zentralrechner[1].pdf
Version – 1/2000

Protocol Parameters -
All RS232 protocol parameters

Example
PortNumber = Com1
BaudRate = 9600
DataBits = 8
Parity = 0
StopBits = 1
TimeOut = 2000

Variable Configuration

1. **Tag Name** – Format of tag name is - **<parameter group><parameter number>**, e.g. Z18, P5 etc.

2. **Start address** – This field indicates the command to be used for reading a parameter. Various commands supported for different parameter types are as below –

I Command – to read an 'actual value' (e.g. Z18) - start address = 0.
Z Command – to read a 'cycle actual value' (e.g. T01, S20) – start address = 1. Maximum 9 cycle actual values can be read together using command Z.
Y Command – to read a 'set value' (e.g. T08) – start address = 2.

3. **End address** – This is the divisor for the parameter value. On MC3F screens, values of some parameter groups are displayed with some divisor. e.g. All parameters of the "T" group are shown either in 1/10 °C (degrees Centigrade, SI) or in 1/10 °F (degrees Fahrenheit, US). Please refer to Appendix B in manual for details. If you want data in PlantConnect to match with this, specify the divisor in end address. So for parameter T01, end address should be 10

Example -
Tag Name = Z18
Start Address = 0

End Address = 0

Tag Name = T01
Start Address = 1
End Address = 10

Tag Name = T08
Start Address = 2
End Address = 10

3.3. MC4

For Krauss Mafai Injection Moulding Machines

Protocol Document referred - mc4_interface.pdf (English translation referred - mc4_interface_english.doc)
Version – MC4 Rechnerschnittstelle Version 1.4

Protocol Parameters

1. All RS232 protocol parameters.
2. **Mode** - The machine can communicate with host computer in 2 modes – 'Standard' mode and 'Extended' mode. In standard mode max. 4 parameters can be read in one command, while in extended mode max. 16 parameters can be read in one command. Valid values for mode – X (extended mode) / S (standard mode)

Example
PortNumber = Com1
BaudRate = 19200
DataBits = 8
Parity = 0
StopBits = 1
TimeOut = 2000
Mode = X

Variable Configuration

7. **Tag Name** – Fully qualified parameter name.

Parameter name format is as below –
<parameter object>.<parameter element >.<parameter value type>
Parameter object identifies a group of parameters, e.g. the parameter object SCRW1_H_BAR_Z01 contains all parameters, which are connected with cylinder1 of injection unit 1.
Through <parameter element> one element of the parameter is described.
Parameter elements can be – SET / ACT / ED / EDACT / XPH / TAH / TNH
Parameter value type can be – VAL / MIN / MAX / LTXT / DIM / etc.
e.g. SCRW1_H_BAR_Z01.SET.MAX

8. **Start address** – not used
9. **End address** – not used

Example
Tag Name = MOLD_S_CLOS_SAFE1.SET.VAL
Start Address = 0
End Address = 0

3.4. SelogicaDriver

For Arburg Injection Moulding Machines

Protocol Document referred - GR_SELOGICA_4x_SPAR.pdf (English translation referred - Host interface of SELOGICA version 3.doc and Host interface of SELOGICA (pg 27 to 46).doc)
Version – Stand 02/2005

Protocol Parameters
All RS232 protocol parameters.

Example
PortNumber = Com1
BaudRate = 9600
DataBits = 8
Parity = 2
StopBits = 1
TimeOut = 2000

Variable Configuration
1. **Tag Name** – Only following parameters can be read from the controller. Please use tag names given in the table below.

Parameters	Tag Name	Range	Format type
Machine number	D001	8-digit	DWORD
Operating mode	D002		WORD
Process status	D003		WORD
Order status	D004		WORD
Program no. 1	D005	15 BYTE ASCII	16 BYTES
Program no. 2	D006	15 BYTE ASCII	16 BYTES
Assignment no.1	D007	15 BYTE ASCII	16 BYTES
Assignment no.2	D008	15 BYTE ASCII	16 BYTES
Follow-up order flag	D009	0 no subsequent order in Layer 2 1 subsequent order in Layer 2	BYTE
Download block	D010	0 MPS 'download' allowed 1 MPS 'download' blocked	BYTE
Cavity	D011		WORD
Target number of units	D012		DWORD
Actual number of units	D013		DWORD

Parameters	Tag Name	Range	Format type
Defective number of units	D014		DWORD
Current time	D015	dd:mm:yy:hh:mm:ss	6 BYTE
Time of last shutdown	D016	dd:hh:mm:ss	4 BYTE
Tool code WKZ 1	D017		WORD
Tool code WKZ 2	D018		WORD
Alarm text no. 1	D019	0 no alarm	WORD
Alarm text no. 2	D020	0 no alarm	WORD
User code	D021		WORD
BDE code	D022		WORD
Dialog flag	D023	0 no change 1 Change in Layer 1 2 Change in Layer 2 3 Change in Layer 1 and 2	WORD
Position of material supply	D024	1,2,3,4,5	WORD
Tool identifier 1	D025	15 BYTE ASCII	16 BYTE
Tool identifier 2	D026	15 BYTE ASCII	16 BYTE
Cylinder code 1	D027		WORD
Cylinder code2	D028		WORD
Average total cycle time (t4032)	t4032	[msec] max.999.9s	DWORD
Cycle time (t4012)	t4012	[msec] max.999.9s	DWORD
External monitoring active	D031	0 no 1 yes	WORD
No. of OK parts produced	D032		DWORD
No. of defective parts produced	D033		DWORD
Injection time	t4018	[msec] - max. 99.99s	DWORD
Dosing time	t4015	[msec] - max. 99.99s	DWORD
Temperature of the cylinder 1	T801I	[1/10 grad C]	WORD
Temperature of the cylinder 2	T802I	[1/10 grad C]	WORD
Temperature of the cylinder 3	T803I	[1/10 grad C]	WORD
Temperature of the cylinder 4	T804I	[1/10 grad C]	WORD
Temperature of the cylinder 5	T805I	[1/10 grad C]	WORD
Temperature of the cylinder 6	T806I	[1/10 grad C]	WORD
Temperature of the cylinder 7	T807I	[1/10 grad C]	WORD
Temperature of the cylinder 8	T808I	[1/10 grad C]	WORD
Temperature of the tool 1	T831I	[1/10 grad C]	WORD
Temperature of the tool 2	T832I	[1/10 grad C]	WORD

Parameters	Tag Name	Range	Format type
Mass volumes of actual value	V4062	[ccm] (End of holding pressure)	REAL
Path of change over to holding pressure	V4065	[cm]/[mm]	REAL
Hydraulic pressure while changing over	p4072	[bar]	REAL

2. **Start address** – Not used
3. **End address** – This is the divisor for the parameter value. Values of some parameters are displayed with some divisor. e.g. All are shown as in 1/10 °C. See above table. If you want data in PlantConnect to match with this, specify the divisor in end address. So for parameter T808I, end address should be 10

Example
Tag Name = t4018
End address = 0

Tag Name = T808I
End address = 10

3.5. Fanuc

For any device with Fanuc controller – series 15, 16, 0i

Protocol Document referred - Commands_to_be_used_0i.pdf (part of the document B-64304EN/01)

For communicating with Fanuc controllers, following commands have to be included in part program of the controller –
POPEN – Open communication with an external device on RS232
DPRNT – Ouput variable values on RS232 port. There can be multiple DPRNT commands in one part program.
PCLOS – Close communication
There is one more command, BPRNT, but that is not supported by PlantConnect Fanuc driver

Please see relevant controller manual for details.
Protocol Parameters -

1. All RS232 protocol parameters
2. **StartChar** – A single unique character that will be sent by controller on RS232 port before sending any parameter values in a cycle. e.g. #. So the first DPRNT command in part program should be DPRNT [#]
3. **EndChar** - A single unique character that will be sent by controller on RS232 port after sending all parameter values in that cycle. e.g. $. So the last DPRNT command in part program should DPRNT [$]

Example
PortNumber = Com1
BaudRate = 9600
DataBits = 8
Parity = 0
StopBits = 1
TimeOut = 2000
StartChar = #
EndChar = $

Sample part program commands
POPEN
DPRNT [#]
DPRNT [X#500[53]]

DPRNT [Y#501[53]]
DPRNT [Z#502[53]]
DPRNT [$]
PCLOS

Variable Configuration

1. **Tag Name** – Character(s) used for identifying the variable in part program, e.g. X. A unique character(s) must be used for each variable in the part program.
2. **Start address** – Variable id. e.g. #500
3. **End address** – End address is comprised of 2 digits, e.g. 53. 1st digit (5) indicates max. number of significant digits in the integer part of variable value. 2nd digit indicates number of significant decimal places in variable value.

Example -
Tag Name = X
Start Address = 500
End Address = 53

3.6. GSI

For any device using generic serial interface for communication.

- **i.** This driver is for devices which send data in a fixed ASCII format.
- **ii.** The data may contain following:
  - ii.i. Date time stamp
  - ii.ii. One or more values
  - ii.iii. Each value separated by some delimiter
- **iii.** The data type of the values can be one of - float, integer, digital or string.
- **iv.** The data would be sent by the device, asynchronously, i.e. only on some event.

Protocol Parameters

1. All RS232 protocol parameters.
2. Following parameters would be needed as per data received for a single variable or multiple variables -

A. Protocol parameters for parsing data for single variable

3. DataFormat

This parameter indicates whether the incoming data contains values for a single variable or multiple variables.
Set the different flags as per received data format-
Multiple variables = 0
Single variable with continuous data = 1
Single variable with data at once = 2

4. Delimiter1

First delimiter character in the data string

5. Delimiter2

Second delimiter character in the data string

6. BufferSize

Maximum size of data string to be read.

B. Protocol parameters for parsing data for multiple variables

7. StartOfData

Number indicating the beginning of data string. These many number of characters would be ignored from start of the data string. Parsing would start the character after this number. This is typically used to ignore few bytes from data in case of data string containing multiple variables.

8. StartingString

String that is found in the incoming string. This is typically used to ignore few bytes from data in case of data string containing multiple variables.

9. DataFieldWidth
Indicates the data field width in case of fixed length data that occurs after starting string
10. StartChars
Start characters of the protocol
11. EndChars
End characters of the protocol
If specified driver would validate each incoming string to ensure StartChars and EndChars
12. Delimiter1
Delimiter character. Please use this if needed.

C. Protocol parameters for parsing text file data for multiple variables

13. FileBased
This parameter indicates the data is read from text file. Flag will be set as 1, if not specified it will be 0.
14. FolderName
Folder name from where the text file will be read.

15. Token specific parameters -

- Parameter name would be Token<number>. E.g. Token1
- Value format would be VariableName, DataType,StartOfDataAfterDelimiter

VariableName needs to be same as its definition in PlantConnect (VariableMasterN.RealName)
DataType would be A/a for numbers, D/d for digital and S/s for strings.
Validate the variable type given by user in tokens to that with actual data type
StartOfDataAfterDelimiter would be a number indicating beginning of data within delimiters. This is optional, if not specified it will be 0.

Note: Number of tokens equal to the number of variables received into the data string.

Examples to add protocol parameters as per data format received:

- Single variable with continuous data:
  Received String: [009620][004895][005200][006500]...
  Protocol Parameters:
  I. Configure all RS232 protocol parameters.
  II. DataFormat = 1
  III. Delimiter1 = [
  IV. Delimiter2 =]
  V. DataFieldWidth = 6

VI. Token1 = Weight~a

- Single variable with data at once:
  Received String: ACIDITY 0.120 %
  Protocol Parameters:
  I. Configure all RS232 protocol parameters.
  II. DataFormat = 2
  III. Delimiter1 = ACIDITY
  IV. Delimiter2 = %
  V. Token1 = Acidity~a
- Multiple variables:
  Received String: 1912200514060501,12.35,10.35,...
  Protocol Parameters:
  I. Configure all RS232 protocol parameters.
  II. DataFormat = 0
  III. StartOfData = 17(it ignores the starting data '1912200514060501,')
  IV. Delimiter1 = x044(use ASCII values for delimiter comma)
  1. Token1 = Fat~a
  2. Token2 = Snf~a

3.7. Titrator

For any device used for titration process. E.g. Potentiometric Titrator, Dairy Titrator and pH etc

Protocol Parameters

All RS232 protocol parameters.

StartingString

String that is found in the incoming string. This is typically used to ignore few bytes from data.

Delimiter1

First delimiter character in the data string.

Delimiter2

Second delimiter character in the data string.

Token1

Name of parameter 1 with data type. E.g. Acidity~a

Examples to add protocol parameters as per the data string received:
Received String:
$S Result 1C.R1 V1
ACIDITY 0.196 %
Protocol Parameters:

I. Configure all RS232 protocol parameters.
II. StartingString = 1C.R1 V1
III. Delimiter1 = ACIDITY
IV. Delimiter2 = %
V. Token1 = Acidity~a

3.8. BayernHessen

For gas analyzers like Horiba

Protocol Document referred – Geysitec_Protocol.pdf, productPDF_30714.pdf

Protocol Parameters

3. All RS232 protocol parameters.
4. **AnalyzerAddress – It is the instrument address.**

Example
PortNumber = Com1
BaudRate = 19200
DataBits = 8
Parity = 0
StopBits = 1
TimeOut = 2000
AnalyzerAddress=1

Variable Configuration

1. Tag Name – Fully qualified parameter name.
2. Start address – Index of the measured value in the response string. If the response string has multiple concentration values, then start address 1 will tell the driver that the first concentration value found in response string is to be associated with this variable.

Example
Tag Name = NO2
Start Address = 1
Tag Name = NO
Start Address = 2

Alarm Configuration
Following are the alarms that are handled by the device
Alarm Id 1: Calibration error
Alarm Id 2: Battery error
Alarm Id 3: Flow rate error
Alarm Id 4: Pressure error

Alarm Id 5: Deozonator error
Alarm Id 6: Temperature error in span gas generator
Alarm Id 7: Light quantity error

3.9. ClinkDriver

To Read gas parameters like CO2, SO2, Nox,NO2 etc

Protocol Parameters

1. All RS232 protocol parameters. (Its value is given in example.)
2. DeviceID – This id is equivalent to model number of analyzer.
3. DeviceType – This paramter is used to identify devic type. Based on its type, device status will be set. Its value can be among three – Analyzer, Calibrator, DustAnalyzer.

Example
PortNumber = COM1
BaudRate = 9600
DataBits = 8
Parity = 0
StopBits = 1
TimeOut = 2000
DeviceID=42(model no. reads Nox parameters)
DeviceType=Analyzer

3.10. Dyanalab

Protocol Parameters

1) All RS232 protocol parameters. (Its value is given in example.)

Example

PortNumber = COM1
BaudRate = 9600
DataBits = 7
Parity = 2
StopBits = 2
TimeOut = 2000

Note: need to send ~ command to receive response on serial port.

Response string contains values for all variables in one string with comma seperator. Each token is the value of one variable. Thus variable mapping is done by using start address of variable. ex. Addr 1 is mapped with 1st token value and so on.

3.11. FH62Driver

Protocol Parameters

2) All RS232 protocol parameters. (Its value is given in example.)

Example

PortNumber = COM1
BaudRate = 9600
DataBits = 7
Parity = 2
StopBits = 2
TimeOut = 2000

In this driver, we need to set command name as a variable tag name. It will give reponse to respective variable as value.

3.12. HostLinkDriver

Protocol Parameters

4. All RS232 protocol parameters. (Its value is given in example.)
5. DeviceID – This id is equivalent to PLC's device Id.

Example
PortNumber = COM1
BaudRate = 9600
DataBits = 7
Parity = 2
StopBits = 2
TimeOut = 2000

DeviceID=0 (PLC's device ID)

3.13. EnviDAQDriver

Protocol Parameters

1) All RS232 protocol parameters. (Its value is given in example.)

2) **SlaveId** – Instrument's module Id

3) **HighIn** – High input value.

4) **LowIn** - Low input value.

Example
PortNumber = COM1
BaudRate = 9600
DataBits = 7
Parity = 2
StopBits = 2
TimeOut = 2000
SlaveId = 1
HighIn = 409
LowIn = 0

3.14. UT3000

Protocol Parameters

1) All RS232 protocol parameters. (Its value is given in example.)

Example

PortNumber = COM1
BaudRate = 9600
DataBits = 8
Parity = 0
StopBits = 1
TimeOut = 2000

Note: need to send ~ command to receive response on serial port.

Response string contains values for all variables in one string with space separator.

3.15. DIO55Driver

Protocol Parameters

1)PortNumber = COM Port Number ex. COM1, COM2 etc

2)StdCycleTime = Std cycle time for current product. User should write this value when a new product is setup. This will be used to calculate Actual cycle time.

3)DigitalInputNumber = Input number whose value to be read. It is zero based value. Ex. If there are 5 inputs then its value in the range 0-4.

Variable Configuration

Sr.	Parameter	Type	Start Address	Logging	Remark
1.	Ejector Pulse	Digital	1	Don't log	Read from DIO55
2.	Actual Cycle Time	Analog	2	log	Calculated in dll Time difference in seconds between occurrences of two consecutive 1 values of Ejector Pulse parameter. Ignore if this time is < (Std. Cycle Time) / 3
3.	Cycle Count	Analog	3	Log all values when this value changes	Calculated in dll. Increment by 1 for every valid cycle.
4.	Cavities	Constant (writable)		Don't log	Constant for a given mould. User should write this value when a new mould is setup
5.	Parts Produced	Calculated		log	= Cavities in each cycle

3.16. FX3U Driver

Protocol Parameters

1. All RS232 protocol parameters. (Its value is given in example.)
    - PortNumber = ttymxc3
    - BaudRate = 9600
    - DataBits = 7
    - Parity = 1
    - StopBits = 1
    - TimeOut = 2000

2. CalculateCheckSum :
    - set 1 For calculate chacksum
    - set 0 for not calculate

Variable Configuration

1. **Tag Name -**
    - Tag name is a compulsory field on UI, so write PLC register addresses then comma and write block number, bunch 10 consecutive register number but it should be same data type and also same register group and assign them same block number

      EXAMPLE:

      TagName : D0100,1
      D0109,1
      D0111,2
      D0120,2
      T0150,3
      T0300,4

2. **Start address –** Start address is not compulsory.
3. **End address –** Data types

Analog Variables

- 30 – 16 bit Unsigned Integer
- 31 - 16 bit Signed Integer
- 32 - 32 bit Unsigned Integer
- 33 - 32 bit Signed Integer
- 34 - 32 bit Float

Digital and String Variables

- DIGITAL and string is not working (function not added in c++)

4. EnVistaInputDriver

Protocol Parameters

1. Provider = Provider Name
2. Server = Server name where actual sql is installed.
3. Database = Database name to which we want to connect.
4. UserName = Authentication is required to communicate with server - User name
5. Password = Authentication is required to communicate with server - Password.

Example

Provider = SQLOLEDB
Server = DEEPASHRIJ\\sqlexpress
Database = EnvidasFW
UserName = sa
Password = aipl12*

5. Drivers using HTTP protocol for Communication

1. Driver Name : MTConnect

2. Protocol Parameter:

1. **URI** - Web address of the device to get the data.
2. **Port Number** - Machine's port number used for http request
3. **Request Type** - Device's which component value to read. It is optional.
   If not mentioned, response will return all components details
4. **DeviceName** – It's optional. If provided, it will return component details of entered device

3. Variable Configuration at server side:

1.In response to probe request we get each configured variables respective DataItem.
DataItems are the variables that we have to configure.
2. Set tag name for variable as **"<XPath of Parameter>,<XPath of Component>"**

4. To test the driver:

4.1. first fire probe request on any browser manually.

URI - mtconnect.mazakcorp.com
Port Number – 5612

Example of probe request
http://mtconnect.mazakcorp.com:5612/probe

Suppose for probe request we have below response:

```
<Components>
<Axes id="a" name="base">
<DataItems>
<DataItem category="CONDITION" id="servo" name="servo_cond" type="ACTUATOR"/>
</DataItems>
<Components>
        <Linear id="x" name="X">...</Linear>
        <Linear id="w" name="W">...</Linear>
        <Linear id="y" name="Y">...</Linear>
        <Linear id="z" name="Z">...</Linear>
        <Linear id="v" name="V">...</Linear>
        <Rotary id="br" name="B">
        <DataItems>
          <DataItem category="SAMPLE" id="bl" name="Bload" nativeUnits="PERCENT" type="LOAD"
units="PERCENT"/>
            <DataItem category="SAMPLE" id="bf" name="Bfrt" nativeUnits="DEGREE/MINUTE"
type="ANGULAR_VELOCITY" units="DEGREE/SECOND"/>
```

```
            <DataItem category="CONDITION" id="bt" name="Btravel" type="ANGLE"/>
          <DataItem category="SAMPLE" id="bpos" name="Bdeg" nativeUnits="DEGREE" subType="ACTUAL"
type="ANGLE" units="DEGREE"/>
            <DataItem category="EVENT" id="brf" name="brfunc" type="ROTARY_MODE">
                        <Constraints>
                        <Value>CONTOUR</Value>
                        <Value>INDEX</Value>
                        </Constraints>
                        </DataItem>
        </DataItems>
        </Rotary>
</Components>
```

Now if we have to read 2 parameters Angular velocity & Load of Component "Rotary" with id = "br", then tag name of variables would be
<XPath of Parameter>,<XPath of Component> where

XPath of Component = Axes/Components/Rotary[@id#"br"]
XPath of Parameter(AngularVelocity) = //Samples/AngularVelocity (i.e. //<Category/<type>)
XPath of Parameter(Load) = //Samples/Load

4.2. After configuration of variables using probe request, now to verify values of configured parameters, execute below request in browser

http://mtconnect.mazakcorp.com:5612/current or
http://mtconnect.mazakcorp.com:5612/<DeviceName>/current

above request provides current value of all parameters

To read current values of particular parameter then Component's XPath is used in current request as below:

Current request with above Component's XPath would be
http://<Machine URL>:<port number>/current?path=//Axes/Components/Rotary[@id#"br"]

we get response for current request as below:

```
<ComponentStream component="Rotary" name="B" componentId="br">
        <Samples>
            <AngularVelocity dataItemId="bf" timestamp="2016-07-18T12:40:34.650868Z" name="Bfrt"
sequence="59488881">0</AngularVelocity>
```

<Load dataItemId="bl" timestamp="2016-07-18T12:40:34.650868Z" name="Bload" sequence="59488882">**0**</Load>
<Angle dataItemId="bpos" timestamp="2016-07-18T12:40:34.650868Z" name="Bdeg" sequence="59488880" subType="ACTUAL">**-90**</Angle>
</Samples>
<Events>
<RotaryMode dataItemId="brf" timestamp="2015-09-18T02:19:14.190174Z" name="brfunc" sequence="7">**UNAVAILABLE**</RotaryMode>
</Events>
<Condition>
<Normal dataItemId="bt" timestamp="2016-07-14T17:21:55.568933Z" name="Btravel" sequence="58720573" type="ANGLE"/>
</Condition>
</ComponentStream>