# Functional Description

# Mot02 - Two Speed or Reversing Motor

## Version 5.2-0

## NOTICE

The information in this document is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this document.

In no event shall ABB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without written permission from ABB, and the contents thereof must not be imparted to a third party nor used for any unauthorized purpose.

The software or hardware described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.

## TRADEMARKS

Release: February 2012

Document number: 3BTG811792-3028

# Table of Contents

# Section 1 - Introduction

## General

Mot02 is a functional unit for motor control in Control$^{IT}$, to be operated from 800xA System. It is used to control two directions or reversing motor. A Mot02 normally performs a complete function independently.

Mot02 has the following functions and properties:

- Different control modes, set by operator or by control logic
- Control circuits supervision
- Interlocks function
- Start & stop control
- Motor current supervision
- Alarm & Event detection and handling

## Function Block

| Name | Data Type | Attributes | Direction | FD Port | Initial value | Description |
|---|---|---|---|---|---|---|
| Name | string | coldretain | in | yes | 'Mot02' | Object name |
| Description | string | coldretain | in | yes | 'Descr' | Object description |
| Enable | bool | coldretain | in | yes | true | Enable object |
| ME | bool | retain | in | yes | true | Enable control circuit alarm |
| M1 | bool | retain | in | yes | true | Control voltage |
| M2 | bool | retain | in | yes | true | Overload |
| M3 | bool | retain | in | yes | true | Motor breaker |
| M4 | bool | retain | in | yes | true | Emergency stop |
| M5 | bool | retain | in | yes | true | Short Circuit |
| IC1 | bool | retain | in | yes | true | Safety Interlock 1 |
| IC2 | bool | retain | in | yes | true | Safety Interlock 2 |
| ICs | ICConn | by_ref | in | yes |  | Safety Interlocks |
| IB1 | bool | retain | in | yes | true | Process Interlock 1 |
| IB2 | bool | retain | in | yes | true | Process Interlock 2 |
| IB3 | bool | retain | in | yes | true | Process Interlock 3 |
| IB4 | bool | retain | in | yes | true | Process Interlock 4 |
| FwdIBs | IBConn | by_ref | in | yes |  | Forward Process Interlocks |
| RevIBs | IBConn | by_ref | in | yes |  | Reverse Process Interlocks |
| IA1 | bool | retain | in | yes | true | Sequence Interlock 1 |
| IA2 | bool | retain | in | yes | true | Sequence Interlock 2 |
| FwdIAs | IAConn | by_ref | in | yes |  | Forward Sequence Interlocks |
| RevIAs | IAConn | by_ref | in | yes |  | Reverse Sequence Interlocks |
| STF | bool | retain | in | yes |  | Start function for direction change |
| T1 | time | coldretain | in | yes | 2s | Start order pulse Time |
| T2 | time | coldretain | in | yes | 4s | Supervision Time |
| T3 | time | coldretain | in | yes | 5s | Run interlock delay Time |
| T4 | time | coldretain | in | yes | 2s | Direction change delay Time |
| TWarn | time | coldretain | in | yes | 0s | Start warning time |
| E1Fwd | bool | retain | in | yes |  | Start forward order in E1 mode |
| E1Rev | bool | retain | in | yes |  | Start reverse order in E1 mode |
| E2Fwd | bool | retain | in | yes |  | Start forward order in E2 mode |
| E2Rev | bool | retain | in | yes |  | Start reverse order in E2 mode |
| Stop | bool | retain | in | yes | true | Stop order in external mode |
| Ack1 | bool | retain | in | yes |  | Main contactor acknowledge forward |
| Ack2 | bool | retain | in | yes |  | Main contactor acknowledge reverse |
| MC | real | retain | in | yes |  | Motor current |
| JogEnbl | bool | retain | in | yes |  | Order mode to Jog |
| JogFwd | bool | retain | in | yes |  | Forward Start order in Jog mode |
| JogRev | bool | retain | in | yes |  | Reverse Start order in Jog mode |
| JogFunc | bool | retain | in | yes |  | Jog start hold function |
| LEnbl | bool | retain | in | yes |  | Order mode to Local |
| L1 | bool | retain | in | yes |  | Forward Start order in Local mode |
| L2 | bool | retain | in | yes |  | Reverse Start order in Local mode |
| LStop | bool | retain | in | yes | true | Local Stop order |
| SeqMan | bool | retain | in | yes |  | Order mode to Man |

| Name | Data Type | Attributes | Direction | FD Port | Initial value | Description |
|---|---|---|---|---|---|---|
| SeqE1 | bool | retain | in | yes | | Order mode to E1 |
| SeqE2 | bool | retain | in | yes | | Order mode to E2 |
| RunInt1 | bool | retain | in | yes | true | Run Interlock 1 |
| RunInt2 | bool | retain | in | yes | true | Run Interlock 2 |
| RunInt2F | bool | retain | in | yes | | Function for Run Interlock 2 |
| AlcBlk | bool | retain | in | yes | | Block alarm |
| AlarmAck | bool | retain | in | yes | | Acknowledge alarm |
| MCAlarm | AlarmParCurr | by_ref | in | yes | | MC alarm configuration |
| InPar | Mot02_InPar | by_ref | in | yes | | In Parameter |
| EventName | string | coldretain | in | yes | '\|\|Mot02_' | Event name |
| NoInt | bool | retain | out | yes | | No Interlocks |
| NoFwdInt | bool | retain | out | yes | | No Interlocks in Forward direction |
| NoRevInt | bool | retain | out | yes | | No Interlocks in Reverse direction |
| NoICInt | bool | retain | out | yes | | No Safety Interlocks (IC) |
| NoFwdIBInt | bool | retain | out | yes | | No Process Interlocks (IB) in Forward direction |
| NoRevIBInt | bool | retain | out | yes | | No Process Interlocks (IB) in Reverse direction |
| NoFwdIAInt | bool | retain | out | yes | | No Sequence Interlocks (IA) in Forward direction |
| NoRevIAInt | bool | retain | out | yes | | No Sequence Interlocks (IA) in Reverse direction |
| Trip | bool | retain | out | yes | | Trip |
| Blk | bool | retain | out | yes | | Standby |
| RFS | bool | retain | out | yes | | Ready for start |
| SO1 | bool | retain | out | yes | | Start order forward |
| SO1P | bool | retain | out | yes | | Start order forward pulse (during T1 time) |
| SO2 | bool | retain | out | yes | | Start order reverse |
| SO2P | bool | retain | out | yes | | Start order reverse pulse (during T1 time) |
| SAck | bool | retain | out | yes | | Start order acknowledgement |
| StartWarn | bool | retain | out | yes | | Start warning |
| Run | bool | retain | out | yes | | Running |
| Sim | bool | retain | out | yes | | Simulation mode |
| JogInd | bool | retain | out | yes | | Jog mode |
| LocalInd | bool | retain | out | yes | | Local mode |
| ManInd | bool | retain | out | yes | | Man mode |
| E1Ind | bool | retain | out | yes | | E1 mode |
| E2Ind | bool | retain | out | yes | | E2 mode |
| Curr | real | retain | out | yes | | Actual current (%) |
| NoOfStart | dint | coldretain | out | yes | | Number of start |
| RunTime | real | coldretain | out | yes | | Total running time (in minutes) |
| LastRunTime | real | coldretain | out | yes | | Last/current running time (in minutes) |
| LastReset | date_and_time | coldretain | out | yes | | Last reset time |
| OutPar | Mot02_OutPar | by_ref | out | yes | | Out Parameter |
| Opr | Mot02_Opr | by_ref | out | yes | | Operator order |

# Data Types

## Mot02_InPar

**Mot02_InPar** data type contains all configuration parameter available in Mot02. Configuration can be entered by writing value to this structured data type and connecting it to input **InPar** of Mot02 function block.

| Name | Data Type | Attributes | Initial value | ISP value | Description |
|---|---|---|---|---|---|
| Class | dint | coldretain | 500 | | AE class |
| Severity | dint | coldretain | 1000 | | AE severity |
| InitMode | dint | coldretain | 5 | | Init mode (5 = Man ; 7 = E1 ; 8 = E2) |
| JogBlk | bool | coldretain | false | | Block operator order Jog mode |
| LocalBlk | bool | coldretain | false | | Block operator order Local mode |
| ManBlk | bool | coldretain | false | | Block operator order Man mode |
| E1Blk | bool | coldretain | false | | Block operator order E1 mode |
| E2Blk | bool | coldretain | false | | Block operator order E2 mode |
| StartFwdBlk | bool | coldretain | false | | Block operator order Start Forward command |
| StartRevBlk | bool | coldretain | false | | Block operator order Start Reverse command |
| StopBlk | bool | coldretain | false | | Block operator order Stop command |
| AlcBlkEvBlk | bool | coldretain | true | | Block event for AlcBlk |
| LEnblEvBlk | bool | coldretain | true | | Block event for LEnbl |
| JogEnblEvBlk | bool | coldretain | true | | Block event for JogEnbl |
| SeqManEvBlk | bool | coldretain | true | | Block event for SeqMan |
| SeqE1EvBlk | bool | coldretain | true | | Block event for SeqE1 |
| SeqE2EvBlk | bool | coldretain | true | | Block event for SeqE2 |
| ExtStartFwdEvBlk | bool | coldretain | true | | Block event for ExtStartFwd |
| ExtStartRevEvBlk | bool | coldretain | true | | Block event for ExtStartRev |
| ExtStopEvBlk | bool | coldretain | true | | Block event for ExtStop |
| IABlockNotExtMode | bool | coldretain | true | | IA blocked when not in E1 or E2 mode |
| IA1 | IAInParType1 | coldretain | | | Configuration for IA1 |
| IA2 | IAInParType1 | coldretain | | | Configuration for IA2 |
| FwdIAs1 | IAInParType1 | coldretain | | | Configuration for FwdIAs.IA1 |
| FwdIAs2 | IAInParType1 | coldretain | | | Configuration for FwdIAs.IA2 |
| RevIAs1 | IAInParType1 | coldretain | | | Configuration for RevIAs.IA1 |
| RevIAs2 | IAInParType1 | coldretain | | | Configuration for RevIAs.IA2 |
| IB1 | IBInParType1 | coldretain | | | Configuration for IB1 |
| IB2 | IBInParType1 | coldretain | | | Configuration for IB2 |
| IB3 | IBInParType1 | coldretain | | | Configuration for IB3 |
| IB4 | IBInParType1 | coldretain | | | Configuration for IB4 |
| FwdIBs1 | IBInParType1 | coldretain | | | Configuration for FwdIBs.IB1 |
| FwdIBs2 | IBInParType1 | coldretain | | | Configuration for FwdIBs.IB2 |
| FwdIBs3 | IBInParType1 | coldretain | | | Configuration for FwdIBs.IB3 |
| FwdIBs4 | IBInParType1 | coldretain | | | Configuration for FwdIBs.IB4 |
| FwdIBs5 | IBInParType1 | coldretain | | | Configuration for FwdIBs.IB5 |
| FwdIBs6 | IBInParType1 | coldretain | | | Configuration for FwdIBs.IB6 |
| FwdIBs7 | IBInParType1 | coldretain | | | Configuration for FwdIBs.IB7 |
| FwdIBs8 | IBInParType1 | coldretain | | | Configuration for FwdIBs.IB8 |
| RevIBs1 | IBInParType1 | coldretain | | | Configuration for RevIBs.IB1 |
| RevIBs2 | IBInParType1 | coldretain | | | Configuration for RevIBs.IB2 |
| RevIBs3 | IBInParType1 | coldretain | | | Configuration for RevIBs.IB3 |
| RevIBs4 | IBInParType1 | coldretain | | | Configuration for RevIBs.IB4 |
| RevIBs5 | IBInParType1 | coldretain | | | Configuration for RevIBs.IB5 |
| RevIBs6 | IBInParType1 | coldretain | | | Configuration for RevIBs.IB6 |

| Name | Data Type | Attributes | Initial value | ISP value | Description |
|---|---|---|---|---|---|
| RevIBs7 | IBInParType1 | coldretain | | | Configuration for RevIBs.IB7 |
| RevIBs8 | IBInParType1 | coldretain | | | Configuration for RevIBs.IB8 |
| IC1 | ICInParType1 | coldretain | | | Configuration for IC1 |
| IC2 | ICInParType1 | coldretain | | | Configuration for IC2 |
| ICs1 | ICInParType1 | coldretain | | | Configuration for ICs.IC1 |
| ICs2 | ICInParType1 | coldretain | | | Configuration for ICs.IC2 |
| CurrRange | RangeCurr | coldretain | | | Current range |
| ShowCurrent | dint | coldretain | 0 | | Current presentation (0 = no current shown ; 1 = current shown in percent ; 2 = current shown in unit) |
| AEConfigM1 | dint | coldretain | 1 | | AE configuration for M1 |
| AEConfigM2 | dint | coldretain | 1 | | AE configuration for M2 |
| AEConfigM3 | dint | coldretain | 1 | | AE configuration for M3 |
| AEConfigM4 | dint | coldretain | 1 | | AE configuration for M4 |
| AEConfigM5 | dint | coldretain | 1 | | AE configuration for M5 |
| AEConfigMCErr | dint | coldretain | 1 | | AE configuration for Main Contactor Error |
| AEConfigRInt1 | dint | coldretain | 0 | | AE configuration for Run Interlock 1 |
| AEConfigRInt2 | dint | coldretain | 0 | | AE configuration for Run Interlock 2 |
| EnNoOfStartCounter | bool | coldretain | false | | Enable number of start counter |
| EnRunTimeCounter | bool | coldretain | false | | Enable running time counter |
| FwdDirection | dint | coldretain | 0 | | Forward direction of arrow presentation (0 = Up, 1 = Down, 2 = Left, 3 = Right) |

## Mot02_OutPar

**Mot02_OutPar** data type contains all information on the object which not available on the output parameter of the function block.

| Name | Data Type | Attributes | Initial value | ISP value | Description |
|---|---|---|---|---|---|
| AlarmBlk | bool | retain | | | Alarm blocked |
| IntlkBlk | bool | retain | | | Interlock blocked |
| IntlkBlkActive | bool | retain | | | Interlock blocked active |
| EnOverrideAll | bool | retain | | | Override All button enabled |
| Mode | dint | retain | | | Active mode |
| NormalMode | bool | retain | | | Normal mode (Active mode = Init mode) |
| IA1Ind | bool | retain | | | IA1 interlocked |
| IA2Ind | bool | retain | | | IA2 interlocked |
| IB1Ind | bool | retain | | | IB1 interlocked |
| IB2Ind | bool | retain | | | IB2 interlocked |
| IB3Ind | bool | retain | | | IB3 interlocked |
| IB4Ind | bool | retain | | | IB4 interlocked |
| IC1Ind | bool | retain | | | IC1 interlocked |
| IC2Ind | bool | retain | | | IC2 interlocked |
| FwdIAs1Ind | bool | retain | | | FwdIAs.IA1 interlocked |
| FwdIAs2Ind | bool | retain | | | FwdIAs.IA2 interlocked |
| FwdIBs1Ind | bool | retain | | | FwdIBs.IB1 interlocked |
| FwdIBs2Ind | bool | retain | | | FwdIBs.IB2 interlocked |
| FwdIBs3Ind | bool | retain | | | FwdIBs.IB3 interlocked |
| FwdIBs4Ind | bool | retain | | | FwdIBs.IB4 interlocked |
| FwdIBs5Ind | bool | retain | | | FwdIBs.IB5 interlocked |
| FwdIBs6Ind | bool | retain | | | FwdIBs.IB6 interlocked |
| FwdIBs7Ind | bool | retain | | | FwdIBs.IB7 interlocked |
| FwdIBs8Ind | bool | retain | | | FwdIBs.IB8 interlocked |
| ICs1Ind | bool | retain | | | ICs.IC1 interlocked |

| Name | Data Type | Attributes | Initial value | ISP value | Description |
|---|---|---|---|---|---|
| ICs2Ind | bool | retain | | | ICs.IC2 interlocked |
| RevIAs1Ind | bool | retain | | | RevIAs.IA1 interlocked |
| RevIAs2Ind | bool | retain | | | RevIAs.IA2 interlocked |
| RevIBs1Ind | bool | retain | | | RevIBs.IB1 interlocked |
| RevIBs2Ind | bool | retain | | | RevIBs.IB2 interlocked |
| RevIBs3Ind | bool | retain | | | RevIBs.IB3 interlocked |
| RevIBs4Ind | bool | retain | | | RevIBs.IB4 interlocked |
| RevIBs5Ind | bool | retain | | | RevIBs.IB5 interlocked |
| RevIBs6Ind | bool | retain | | | RevIBs.IB6 interlocked |
| RevIBs7Ind | bool | retain | | | RevIBs.IB7 interlocked |
| RevIBs8Ind | bool | retain | | | RevIBs.IB8 interlocked |
| HCurr | AlarmInd | retain | | | Alarm Indication for HCurr |
| M1 | AlarmInd | retain | | | Alarm Indication for M1 |
| M2 | AlarmInd | retain | | | Alarm Indication for M2 |
| M3 | AlarmInd | retain | | | Alarm Indication for M3 |
| M4 | AlarmInd | retain | | | Alarm Indication for M4 |
| M5 | AlarmInd | retain | | | Alarm Indication for M5 |
| MCErr | AlarmInd | retain | | | Alarm Indication for MCErr |
| RInt1 | AlarmInd | retain | | | Alarm Indication for RInt1 |
| RInt2 | AlarmInd | retain | | | Alarm Indication for RInt2 |
| Dir | bool | retain | | | Running direction (0 = forward, 1 = reverse) |
| DirChange | bool | retain | | | Change of running direction |
| LastStop | dint | retain | | | Reason for last stop |
| StartTime | time | retain | | | Actual starting time |
| StopTime | time | retain | | | Actual stopping time |
| Ack | bool | retain | | | Running feedback indication |
| MC | real | retain | | | Motor current indication |

## Mot02_Opr

**Mot02_Opr** data type contains all operator order.

| Name | Data Type | Attributes | Initial value | ISP value | Description |
|---|---|---|---|---|---|
| BlockAlarm | bool | retain | | | Operator block alarms |
| Jog | bool | retain | | | Operator order Jog mode |
| Local | bool | retain | | | Operator order Local mode |
| Man | bool | retain | | | Operator order Manual mode |
| E1 | bool | retain | | | Operator order E1 mode |
| E2 | bool | retain | | | Operator order E2 mode |
| StartFwd | bool | retain | | | Operator order Start Forward command |
| StartRev | bool | retain | | | Operator order Start Reverse command |
| Stop | bool | retain | | | Operator order Stop command |
| OverrideAll | bool | retain | | | Operator override all interlocks |
| IB1Override | bool | retain | | | Operator override IB1 interlock |
| IB2Override | bool | retain | | | Operator override IB2 interlock |
| IB3Override | bool | retain | | | Operator override IB3 interlock |
| IB4Override | bool | retain | | | Operator override IB4 interlock |
| FwdIBs1Override | bool | retain | | | Operator override FwdIBs.IB1 interlock |
| FwdIBs2Override | bool | retain | | | Operator override FwdIBs.IB2 interlock |
| FwdIBs3Override | bool | retain | | | Operator override FwdIBs.IB3 interlock |
| FwdIBs4Override | bool | retain | | | Operator override FwdIBs.IB4 interlock |
| FwdIBs5Override | bool | retain | | | Operator override FwdIBs.IB5 interlock |
| FwdIBs6Override | bool | retain | | | Operator override FwdIBs.IB6 interlock |
| FwdIBs7Override | bool | retain | | | Operator override FwdIBs.IB7 interlock |

| Name | Data Type | Attributes | Initial value | ISP value | Description |
|---|---|---|---|---|---|
| FwdIBs8Override | bool | retain | | | Operator override FwdIBs.IB8 interlock |
| RevIBs1Override | bool | retain | | | Operator override RevIBs.IB1 interlock |
| RevIBs2Override | bool | retain | | | Operator override RevIBs.IB2 interlock |
| RevIBs3Override | bool | retain | | | Operator override RevIBs.IB3 interlock |
| RevIBs4Override | bool | retain | | | Operator override RevIBs.IB4 interlock |
| RevIBs5Override | bool | retain | | | Operator override RevIBs.IB5 interlock |
| RevIBs6Override | bool | retain | | | Operator override RevIBs.IB6 interlock |
| RevIBs7Override | bool | retain | | | Operator override RevIBs.IB7 interlock |
| RevIBs8Override | bool | retain | | | Operator override RevIBs.IB8 interlock |
| IA1Override | bool | retain | | | Operator override IA1 interlock |
| IA2Override | bool | retain | | | Operator override IA2 interlock |
| FwdIAs1Override | bool | retain | | | Operator override FwdIAs.IA1 interlock |
| FwdIAs2Override | bool | retain | | | Operator override FwdIAs.IA2 interlock |
| RevIAs1Override | bool | retain | | | Operator override RevIAs.IA1 interlock |
| RevIAs2Override | bool | retain | | | Operator override RevIAs.IA2 interlock |
| ResetCounter | bool | retain | | | Order reset counter |
| EnableSim | bool | retain | | | Enable simulated running feedback |
| Stop | bool | retain | | | Operator order Stop command |
| OverrideAll | bool | retain | | | Operator override all interlocks |

## Permission

All variables in **Mot02_InPar** data type are set with *Configure* permission in 800xA system. All variables in **Mot02_Opr** data type are set with *Operate* permission.

The variable permission can be configured according to the plant requirement. This can be done by adding **Property Attribute Override** aspect for the object in Control Structure or for the object type in Object Type Structure.

## Control Modes

There are five different control modes are available in priority order: Jog, Local, Man, E1 and E2. Active mode is indicated on the faceplate and object display. These modes can also be blocked individually from operator access. When motor is in one particular mode, control from other location is blocked.

### Jog Mode

Jog mode is suitable for testing motor in the field. The motor is controlled from start / stop station adjacent to the motor. Jog mode can be activated from the *Jog* button on the faceplate or by setting the input **JogEnbl** of the function block.

When motor is in Jog mode, control from other location is blocked. The interlocks supervision is limited. Only the safety interlocks **IC** and faults in **M1** - **M5** will be monitored. When the motor is leaving Jog mode, a stop order is generated automatically. It will then resume the last control mode from operator station.

The motor is started and stopped through input **JogFwd**, **JogRev** and **JogFunc**. **JogFwd** is for receiving start forward order and **JogFwd** is for receiving start reverse order while **JogFunc** is for receiving the stop or hold order from field. The function of these inputs is set as follow:

| **JogFwd/Rev** | **JogFunc** | |
|---|---|---|
| 0 | 0 | Stop order |
| 0 | 1 | No change |
| 1 | 0 | Start and stop order on **JogFwd** / **JogRev** |
| 1 | 1 | Start order on **JogFwd** / **JogRev** and stop order on **JogFunc** |

### Local Mode

The motor is controlled from a local desk or panel. As long as **JogEnbl** is not active, Local mode can be activated from the **Local** button on the faceplate or by setting the input **LEnbl** of the function block.

When motor is in Local mode, control from other location is blocked. When the motor is leaving Local mode, it will resume the last control mode from operator station.

The motor is started with a rising edge pulse to **L1** for forward direction and **L2** for reverse direction while **LStop** must be in *True* condition. Motor can be stopped with a falling edge pulse to **LStop**.

### Manual Mode

In this mode the operator directly start forward / start reverse and stop motor from the faceplate button **Fwd** / **Rev** and **Stop**. As long as **JogEnbl** and **LEnbl** is not active, Manual mode can be activated from the **Man** button on the faceplate or with a rising edge pulse to the input **SeqMan** of the function block.

### E1 and E2 Mode

The motor is controlled from external signals, i.e.: a process signal. It is normally used for automatic start / stop agitators, belt conveyors, etc. As long as **JogEnbl** and **LEnbl** is not active, E1 or E2 mode can be activated from the **E1** or **E2** button on the faceplate or with a rising edge pulse to the input **SeqE1** or **SeqE2** of the function block.

The motor is started with a rising edge pulse to **E1Fwd** / **E1Rev** for E1 mode or **E2Fwd** / **E2Rev** for E2 mode while **Stop** must be in *True* condition. Motor can be stopped with a falling edge pulse to **Stop** for both modes.

Operator can also stop the motor by pressing the faceplate button **Stop**. Control mode will then change to Manual to prevent motor from being started again automatically.

The table below describes the commands at the different control modes of Mot02.

| Order | Jog | Local | Manual | E1 | E2 |
|---|---|---|---|---|---|
| Input *JogFwd* / *JogRev* | Start / Stop | - | - | - | - |
| Input *L1* / *L2* | - | Start | - | - | - |
| Input *LStop* | - | Stop | - | - | - |
| Operator start Fwd/Rev | - | - | Start | - | - |
| Operator stop | - | - | Stop | Stop | Stop |
| Input *E1Fwd* / *E1Rev* | - | - | - | Start | - |
| Input *E2Fwd* / *E2Rev* | - | - | - | - | Start |
| Input *Stop* | - | - | - | Stop | Stop |

## Interlocks

The motor control can be interlocked by signals from the process or control logic. Interlocks or faults are active when the value is *False*. The interlocks are divided into four groups with the following designations and functions. All interlocks are blocked when motor in Jog mode except for Safety Interlocks.

### Safety Interlocks

Safety interlocks are used to interlock the motor for safety purposes. There are 4 safety interlocks available: **IC1**, **IC2**, **ICs.IC1** and **ICs.IC2**. These interlocks cannot be blocked and apply to both forward and reverse direction.

### Process Interlocks

Process interlocks are used for normal process interlocks. There are 20 process interlocks available: **IB1** - **IB4**, **FwdIBs.IB1** - **FwdIBs.IB8** and **RevIBs.IB1** - **RevIBs.IB8**.

Interlocks **IB1** - **IB4** are applied for both forward and reverse direction. Interlocks **FwdIBs.IB1** - **FwdIBs.IB8** are applied for forward direction only and Interlocks **RevIBs.IB1** - **RevIBs.IB8** are applied for reverse direction only.

All can be configured to accept blocking by the operator. They can also be configured as start interlocks. A start interlock will prevent the motor from starting, but does not stop a running motor.

### Sequence Interlocks

Sequence interlocks are used if the motor is to be interlocked against other objects such as a pump motor or a group start or other conditions in an automatic sequence. There are 6 sequence interlocks available: **IA1**, **IA2**, **FwdAs.IA1**, **FwdIAs.IA2**, **RevIAs.IA1** and **RevIAs.IA2**.

Interlocks **IA1** and **IA2** are applied for both forward and reverse direction. Interlocks **FwdIAs.IA1** and **FwdIAs.IA2** are applied for forward direction only and Interlocks **RevIAs.IA1** and **RevIAs.IA2** are applied for reverse direction only.

These interlocks are configurable as blocked or not when motor is not in E1 or E2 mode.

If **IA blocked when not Ext** is equal to 1

Motor interlocked if **IA1**, **FwdIAs.IA1** or **RevIAs.IA1** is 0 and the motor is in E1 mode

Motor interlocked if **IA2**, **FwdIAs.IA2** or **RevIAs.IA2** is 0 and the motor is in E2 mode

If **IA blocked when not Ext** is equal to 0

Motor interlocked if **IA1**, **IA2**, **FwdAs.IA1**, **FwdIAs.IA2**, **RevIAs.IA1** or **RevIAs.IA2** is 0 and the motor is in Local, Man, E1 or E2 mode.

### Run Interlocks

Run interlocks are used to obtain alarm or fault while motor is running. There are 2 run interlocks available: **RunInt1** and **RunInt2**. Run interlocks are blocked during the time *T3* after starting. Run interlocks have Alarm & Event configuration to define the alarm treatment.

**RunInt1** has a motor tripping function whereas the **RunInt2** function can be configured with input **RunInt2F**. When input **RunInt2F** is 1, the motor is tripped and when **RunInt2F** is 0, only an alarm is given.

## Ready for Start

Ready for Start means all condition is cleared for motor to start. There is no Safety and Process interlocks, Sequence interlocks are not included. No faults / unacknowledged faults alarms exists. Motor is not in Jog mode.

Ready for Start is indicated by the output parameter **RFS** in the function block.

## Start and Stop

Start and stop commands may come from faceplate button or input parameter, depending on the selected control mode.

When a start command is issued, it will check if start warning should be given. Start warning **StartWarn** is activated if start warning time **TWarn** is greater than zero. After **TWarn** time elapses, the start forward order **SO1** or start reverse order **SO2** is sent to the motor. The start forward order pulse **SO1P** or start reverse order pulse **SO2P** is a pulse with a duration determined by the input **T1**.

The motor’s main contactor acknowledges the start order by activating the running forward feedback **Ack1** or running reverse feedback **Ack2**. The main contactor must acknowledge within **T2** time.

If main contactor acknowledges the start order **SO1** / **SO2** within the time **T2**, Mot02 will keep **SO1** / **SO2** high. Otherwise, the start attempt is deemed abortive and an alarm about the contactor failure is issued. The contactor failure alarm indication disappears when the operator acknowledges the alarm. A new attempt to start the motor then can be made.

Mot02 also issues an alarm about main contactor failure when the contactor acknowledges start orders falsely, i.e. when it sets the inputs **Ack1** / **Ack2** high without any start order being issued or after stop order is given. Alarm will be generated after **T2** time.

Input parameter **STF** defines the possibility to change the direction of motor while it is running. If **STF** is set to 0, then stop order must be issued before changing to other direction. If **STF** is set to 1, then change of direction can be performed without first stopping the motor. The order for change of direction is delayed by time defined on input **T4**.

## Control Circuit Faults

Fault supervision is performed in the priority order **M1**, **M2**, **M5**, **M3** and **M4**. This means that if input **M1** is in fault condition, then inputs **M2** - **M5** are not regarded. When a number of faults occur in series, only the first fault will be reported in the alarm. Value at inputs **M1** - **M5** is 1 when there are no faults. Faults are indicated on the output **Trip**.

**M1** - **M5** have Alarm & Event configuration to define the alarm treatment. The input **ME** can be used to block the alarm from being generated. No alarm will be issued if motor is not running.

## Motor Current Supervision

The input **MC** is connected to the process signal for motor current measurement. The **MC** is normally measured in unit such as *Amp*.

The output parameter **Curr** presents the measured current in percentage. It is obtained by comparing the **MC** value against **Rated Current**. **Rated Current** is a value that corresponds to 100 % load.

Current will be supervised only after motor running for more than **Start Delay** time. High current is indicated when current is greater than the High Current Limit.

## Simulation

The function block support internal simulation. Simulation can be activated by enabling the simulation button in Interaction Window or by setting the **Opr.EnableSim** to *True* using the bulk tool. The simulation will be reset when cold download is performed.

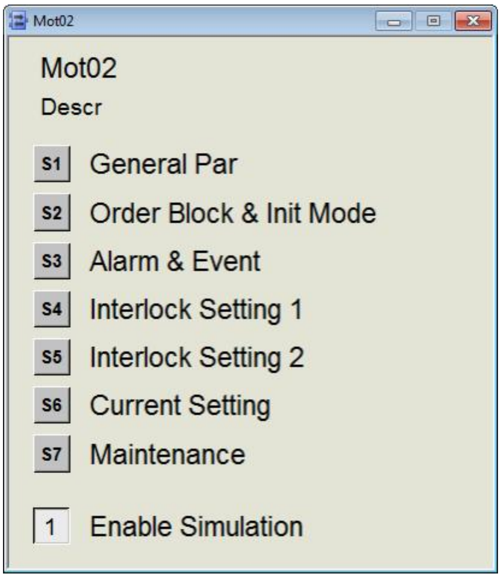
Object in simulation mode will be indicated with a frame on the graphic elements. The default color is flashing yellow as defined by *ppSimulation* in **Pulp & Paper Library Colors** aspect. It also indicates **Sim** next to the mode indication in faceplate.

Running feedback will be activated 2 seconds after start order **SO1** or **SO2** is *True*. The current will be simulated at 50%.

## Interaction Window

Interaction Window is used to configure the object's parameter in the Control Builder. It is only accessible when Control Builder is in *Online* mode by right clicking the function block.

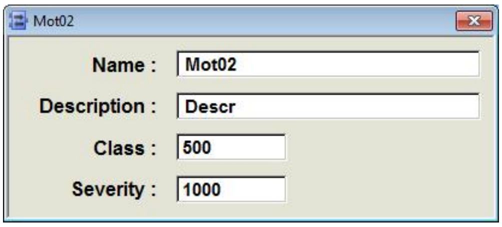
The main Interaction Window shows the name and description of the function block. The buttons are link to open the sub-window.



### 1. General Par

**Name** and **Description** can be entered from interaction window or from the function block parameters directly. After performing *Name Uploader*, these values will be used as Object Name and Description and shown in faceplate.

**Class** defines the process section or area in which alarms are grouped. By utilizing **Class**, the alarms can be filtered. Valid values are 1 - 9999. **Severity** defines the alarm priority for general alarms. Valid values are 1 - 1000 where 1000 is the highest priority.



Mot02

Name : Mot02

Description : Descr

Class : 500

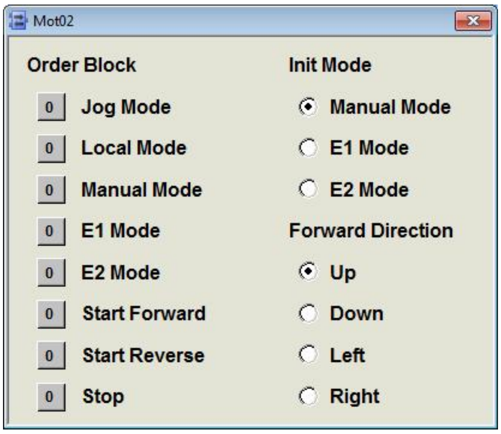
Severity : 1000

### 2. Order Block & Init Mode

**Order Block** is used to prevent operator from giving an order of certain activity. The faceplate button will be dimmed if order block is activated (set to 1).

**Init Mode** defines the initial mode of the object when cold download is performed. By default, the init mode for Mot02 is Manual mode.

**Forward Direction** is used to present the arrow indication while running in forward direction. Running in reverse direction will be indicated with the opposite of the selection. If forward direction selected is *Up*, then reverse direction will be *Down*.


Mot02

Order Block

0 Jog Mode
0 Local Mode
0 Manual Mode
0 E1 Mode
0 E2 Mode
0 Start Forward
0 Start Reverse
0 Stop

Init Mode

● Manual Mode
○ E1 Mode
○ E2 Mode

Forward Direction

● Up
○ Down
○ Left
○ Right

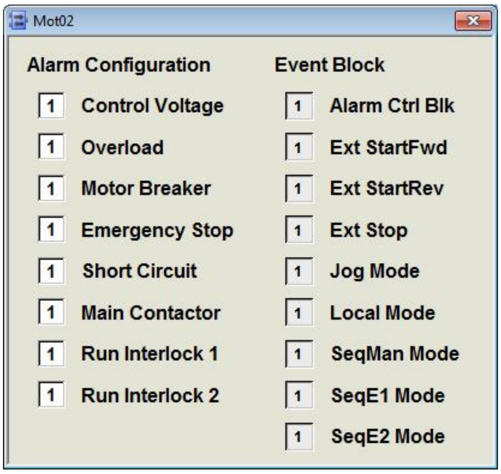
## 3. Alarm & Event

**Alarm Configuration** is used to configure how to present status changes of certain conditions. Alarm Configuration can be set in 3 different ways:

- 0 = No alarm or event will be generated
- 1 = Alarm and event will be generated
- 2 = Only event will be generated

**Event Block** is used to block the event message generated due to changes in value from logic connection. To enable event for selected message, press the button to set the value to 0.

The text message for each alarm / event is stored in the **AE Translator** aspect. List of events for all operator activity is managed in 800xA system with Audit Trail functionality.


Mot02

Alarm Configuration

1 Control Voltage
1 Overload
1 Motor Breaker
1 Emergency Stop
1 Short Circuit
1 Main Contactor
1 Run Interlock 1
1 Run Interlock 2

Event Block

1 Alarm Ctrl Blk
1 Ext StartFwd
1 Ext StartRev
1 Ext Stop
1 Jog Mode
1 Local Mode
1 SeqMan Mode
1 SeqE1 Mode
1 SeqE2 Mode

## 4. Interlock Setting 1 & 2

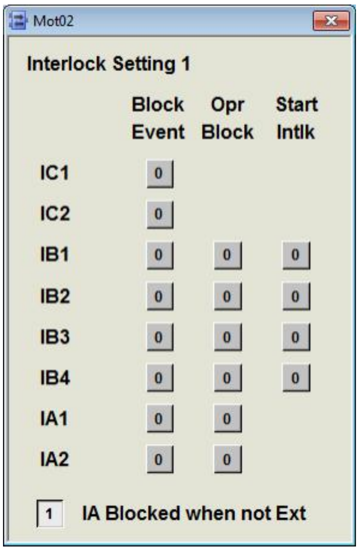
Interlock Setting is split into two. Interlock Setting 1 is for standard interlock **IC1** - **IC2**, **IB1** - **IB4** and **IA1** - **IA2**. Interlock Setting 2 is for extended interlock **ICs.IC1** - **ICs.IC2**, **FwdIBs.IB1** - **FwdIBs.IB8**, **RevIBs.IB1** - **RevIBs.IB8**, **FwdIA2.IA1** - **FwdIAs.IA2** and **RevIA2.IA1** - **RevIAs.IA2**.

When **Block Event** is disabled (set to 0), changes in value of the interlock will generate an event message. The text message for each event is stored in the **AE Translator** aspect.

Enabling the **Opr Block** (set to 1) will give operator the possibility to override the interlock(s) from faceplate.

When an interlock is configured as **Start Interlock**, it will only be monitored during the starting period. Once the motor is started, the interlock will no longer be observed.

By configuring **IA Blocked when not Ext** to 1, IA interlocks will not be monitored when the object is not in E1 or E2 mode. By setting the value to 0, IA interlocks will be observed in all modes.

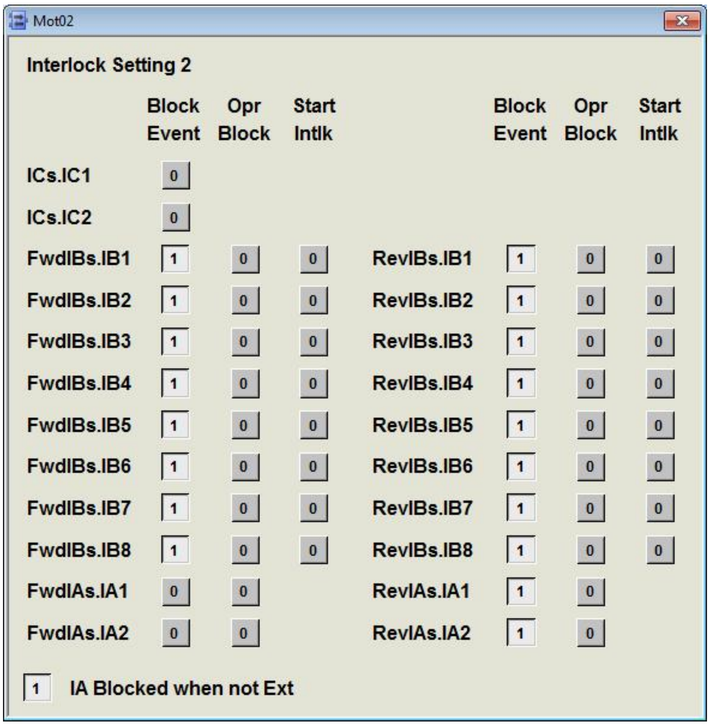


Mot02

Interlock Setting 1

| | Block Event | Opr Block | Start Intlk |
|---|---|---|---|
| IC1 | 0 | | |
| IC2 | 0 | | |
| IB1 | 0 | 0 | 0 |
| IB2 | 0 | 0 | 0 |
| IB3 | 0 | 0 | 0 |
| IB4 | 0 | 0 | 0 |
| IA1 | 0 | 0 | |
| IA2 | 0 | 0 | |

1 IA Blocked when not Ext



Mot02

Interlock Setting 2

| | Block Event | Opr Block | Start Intlk | | Block Event | Opr Block | Start Intlk |
|---|---|---|---|---|---|---|---|
| ICs.IC1 | 0 | | | | | | |
| ICs.IC2 | 0 | | | | | | |
| FwdIBs.IB1 | 1 | 0 | 0 | RevIBs.IB1 | 1 | 0 | 0 |
| FwdIBs.IB2 | 1 | 0 | 0 | RevIBs.IB2 | 1 | 0 | 0 |
| FwdIBs.IB3 | 1 | 0 | 0 | RevIBs.IB3 | 1 | 0 | 0 |
| FwdIBs.IB4 | 1 | 0 | 0 | RevIBs.IB4 | 1 | 0 | 0 |
| FwdIBs.IB5 | 1 | 0 | 0 | RevIBs.IB5 | 1 | 0 | 0 |
| FwdIBs.IB6 | 1 | 0 | 0 | RevIBs.IB6 | 1 | 0 | 0 |
| FwdIBs.IB7 | 1 | 0 | 0 | RevIBs.IB7 | 1 | 0 | 0 |
| FwdIBs.IB8 | 1 | 0 | 0 | RevIBs.IB8 | 1 | 0 | 0 |
| FwdIAs.IA1 | 0 | 0 | | RevIAs.IA1 | 1 | 0 | |
| FwdIAs.IA2 | 0 | 0 | | RevIAs.IA2 | 1 | 0 | |

1 IA Blocked when not Ext

## 5. Current Setting

**Current Presentation** determines how the current is presented in the faceplate. Three options available are no current presentation, current to be presented in percentage (%) and current to be presented in unit as defined in **Current Range**.

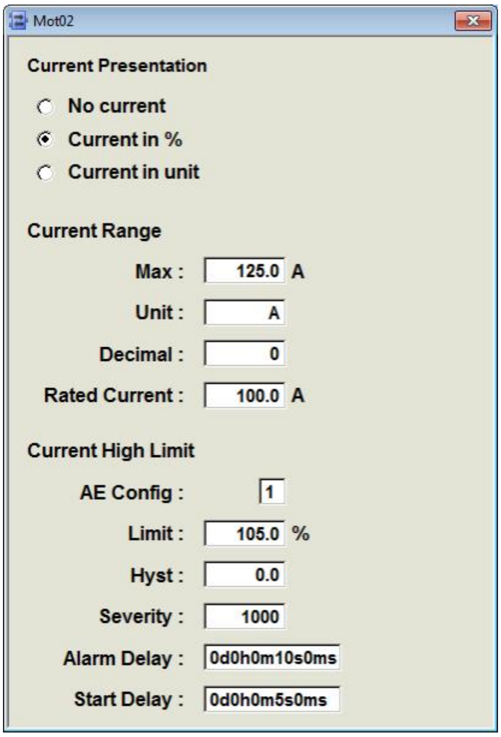
**Current Range** covers maximum range, unit, decimal and rated current. Max refers to the maximum range of the current I/O signal. **Rated Current** refers to the current value equals to 100%.

Only one current high limit can be configured. AE Config can be set in 3 different ways:

- 0 = No Alarm or Event will be generated
- 1 = Alarm and Event will be generated
- 2 = Only Event will be generated

If AE Config is set to 1, alarm will be generated if the limit stays active longer than alarm delay time. Otherwise, alarm will not be generated. Current will be supervised only after motor running for more than **Start Delay** time.

The alarm limit entered can be considered as in unit or in percentage depending on the current presentation selected. If the current presentation is to show current in %, then the alarm limit is assumed in %. If the current presentation is to show current in unit, then the alarm limit is assumed in unit as well.



Mot02

Current Presentation

○ No current
● Current in %
○ Current in unit

Current Range

Max : 125.0 A

Unit : A

Decimal : 0

Rated Current : 100.0 A

Current High Limit

AE Config : 1

Limit : 105.0 %

Hyst : 0.0

Severity : 1000

Alarm Delay : 0d0h0m10s0ms

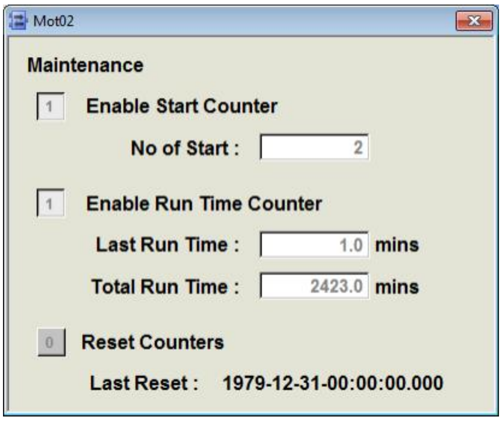
Start Delay : 0d0h0m5s0ms

## 6. Maintenance

Maintenance covers the number of start and running time.

When **Enable Start Counter** is activated, it will count how many times the motor has been successfully started. Two types of run time will be counted when activating **Enable Run Time Counter**. **Last Run Time** indicates the last run time if motor is not running or current running time if motor is still running. **Total Run Time** will count how long the motor has been running since the last reset. Both run time is incremented on per minute period.

These values can be reset to zero by pressing the **Reset Counters** button. Reset can only be done when motor is not running. **Last Reset** indicates the date and time when the last reset is performed. If Reset has not been performed before, the **Last Reset** will show *1979-12-31-00:00:00.000*.



Mot02

Maintenance

1 Enable Start Counter

No of Start : 2

1 Enable Run Time Counter

Last Run Time : 1.0 mins

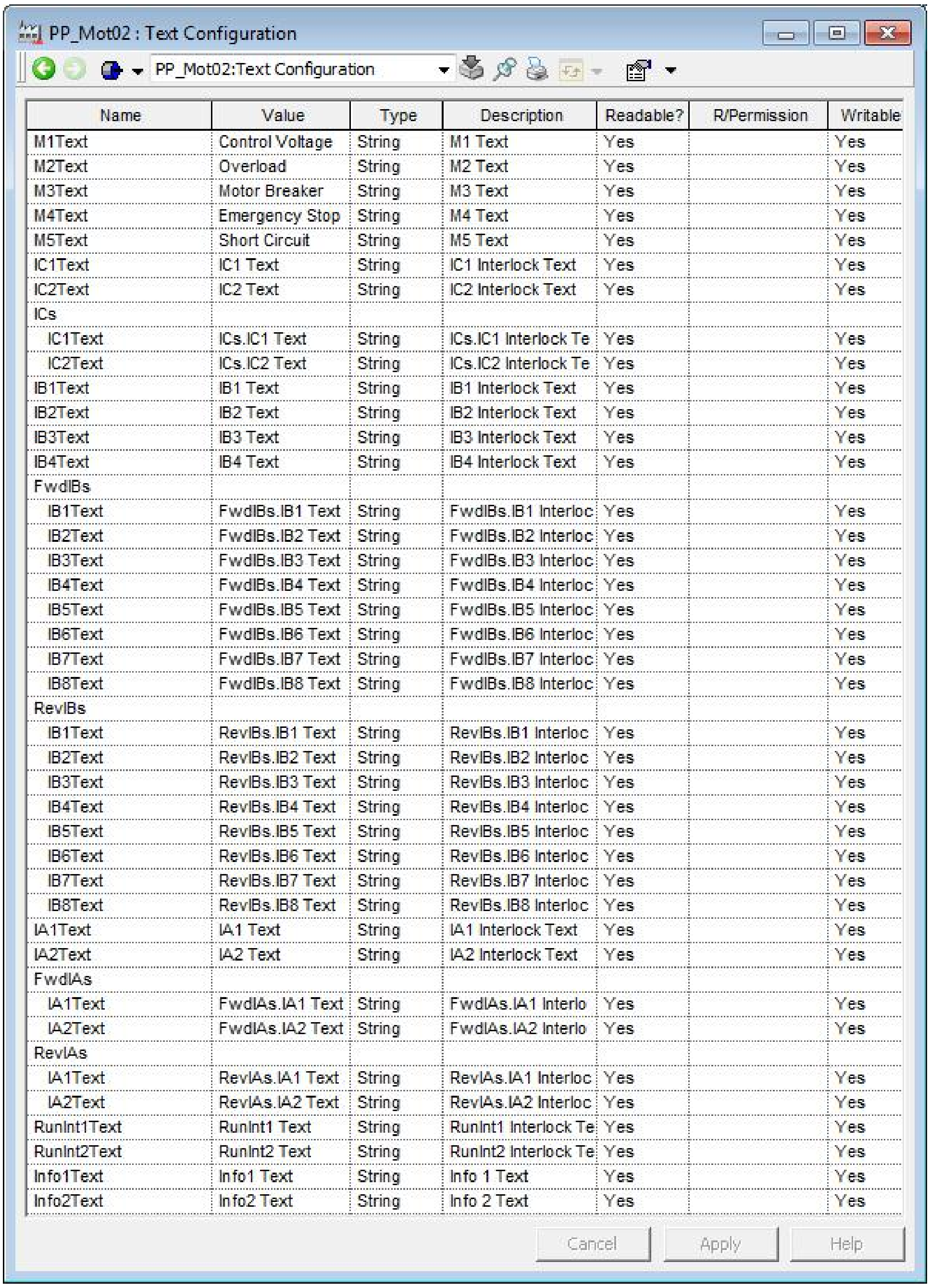
Total Run Time : 2423.0 mins

0 Reset Counters

Last Reset : 1979-12-31-00:00:00.000

# Text Configuration

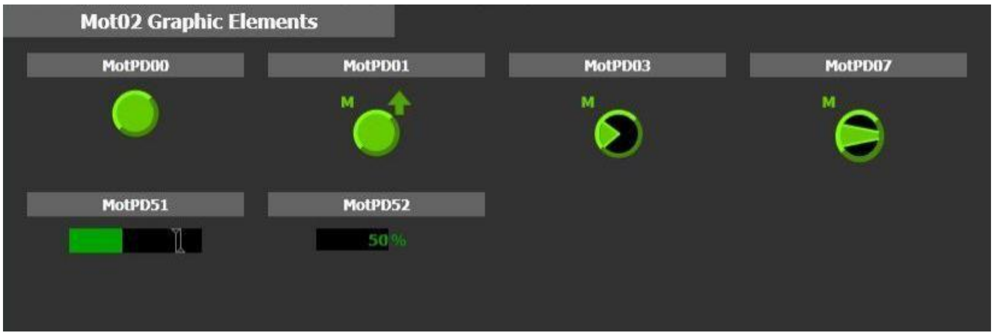
**Text Configuration** aspect is used to define the text properties of the object, such as control circuit faults, interlocks and info texts. It is accessible in the Control Structure. The length of the each text should not exceed more than 50 characters.



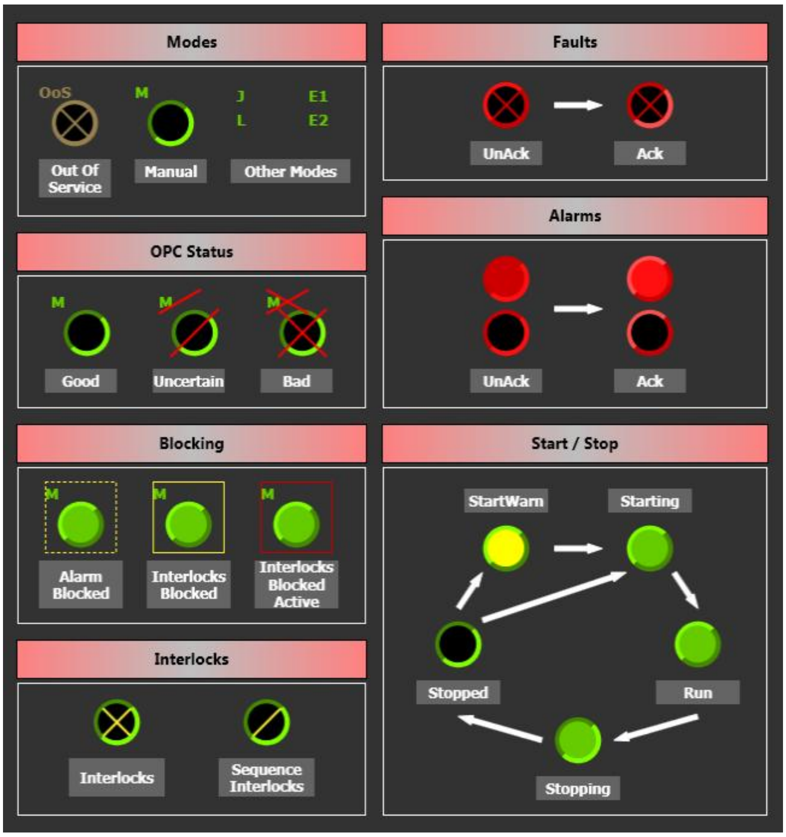
| Name | Value | Type | Description | Readable? | R/Permission | Writable |
|---|---|---|---|---|---|---|
| M1Text | Control Voltage | String | M1 Text | Yes | | Yes |
| M2Text | Overload | String | M2 Text | Yes | | Yes |
| M3Text | Motor Breaker | String | M3 Text | Yes | | Yes |
| M4Text | Emergency Stop | String | M4 Text | Yes | | Yes |
| M5Text | Short Circuit | String | M5 Text | Yes | | Yes |
| IC1Text | IC1 Text | String | IC1 Interlock Text | Yes | | Yes |
| IC2Text | IC2 Text | String | IC2 Interlock Text | Yes | | Yes |
| ICs | | | | | | |
| IC1Text | ICs.IC1 Text | String | ICs.IC1 Interlock Te | Yes | | Yes |
| IC2Text | ICs.IC2 Text | String | ICs.IC2 Interlock Te | Yes | | Yes |
| IB1Text | IB1 Text | String | IB1 Interlock Text | Yes | | Yes |
| IB2Text | IB2 Text | String | IB2 Interlock Text | Yes | | Yes |
| IB3Text | IB3 Text | String | IB3 Interlock Text | Yes | | Yes |
| IB4Text | IB4 Text | String | IB4 Interlock Text | Yes | | Yes |
| FwdIBs | | | | | | |
| IB1Text | FwdIBs.IB1 Text | String | FwdIBs.IB1 Interloc | Yes | | Yes |
| IB2Text | FwdIBs.IB2 Text | String | FwdIBs.IB2 Interloc | Yes | | Yes |
| IB3Text | FwdIBs.IB3 Text | String | FwdIBs.IB3 Interloc | Yes | | Yes |
| IB4Text | FwdIBs.IB4 Text | String | FwdIBs.IB4 Interloc | Yes | | Yes |
| IB5Text | FwdIBs.IB5 Text | String | FwdIBs.IB5 Interloc | Yes | | Yes |
| IB6Text | FwdIBs.IB6 Text | String | FwdIBs.IB6 Interloc | Yes | | Yes |
| IB7Text | FwdIBs.IB7 Text | String | FwdIBs.IB7 Interloc | Yes | | Yes |
| IB8Text | FwdIBs.IB8 Text | String | FwdIBs.IB8 Interloc | Yes | | Yes |
| RevIBs | | | | | | |
| IB1Text | RevIBs.IB1 Text | String | RevIBs.IB1 Interloc | Yes | | Yes |
| IB2Text | RevIBs.IB2 Text | String | RevIBs.IB2 Interloc | Yes | | Yes |
| IB3Text | RevIBs.IB3 Text | String | RevIBs.IB3 Interloc | Yes | | Yes |
| IB4Text | RevIBs.IB4 Text | String | RevIBs.IB4 Interloc | Yes | | Yes |
| IB5Text | RevIBs.IB5 Text | String | RevIBs.IB5 Interloc | Yes | | Yes |
| IB6Text | RevIBs.IB6 Text | String | RevIBs.IB6 Interloc | Yes | | Yes |
| IB7Text | RevIBs.IB7 Text | String | RevIBs.IB7 Interloc | Yes | | Yes |
| IB8Text | RevIBs.IB8 Text | String | RevIBs.IB8 Interloc | Yes | | Yes |
| IA1Text | IA1 Text | String | IA1 Interlock Text | Yes | | Yes |
| IA2Text | IA2 Text | String | IA2 Interlock Text | Yes | | Yes |
| FwdIAs | | | | | | |
| IA1Text | FwdIAs.IA1 Text | String | FwdIAs.IA1 Interlo | Yes | | Yes |
| IA2Text | FwdIAs.IA2 Text | String | FwdIAs.IA2 Interlo | Yes | | Yes |
| RevIAs | | | | | | |
| IA1Text | RevIAs.IA1 Text | String | RevIAs.IA1 Interloc | Yes | | Yes |
| IA2Text | RevIAs.IA2 Text | String | RevIAs.IA2 Interloc | Yes | | Yes |
| RunInt1Text | RunInt1 Text | String | RunInt1 Interlock Te | Yes | | Yes |
| RunInt2Text | RunInt2 Text | String | RunInt2 Interlock Te | Yes | | Yes |
| Info1Text | Info1 Text | String | Info 1 Text | Yes | | Yes |
| Info2Text | Info2 Text | String | Info 2 Text | Yes | | Yes |

## Process Display

Mot02 has the following graphic elements which can be inserted into graphic displays.



The following display shows the presentation in graphic elements under different states.



# Faceplate

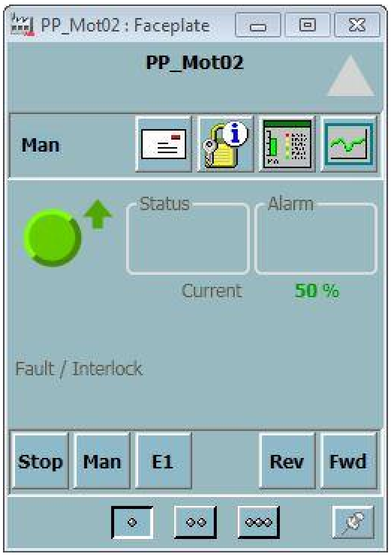
## Aspect Link

| Icon | Aspect View |
|---|---|
| 🗒 | Operator Note |
| 🔓 | Interlocks Display |
| ▦ | Object Display |
| 📈 | Object Trend Display |

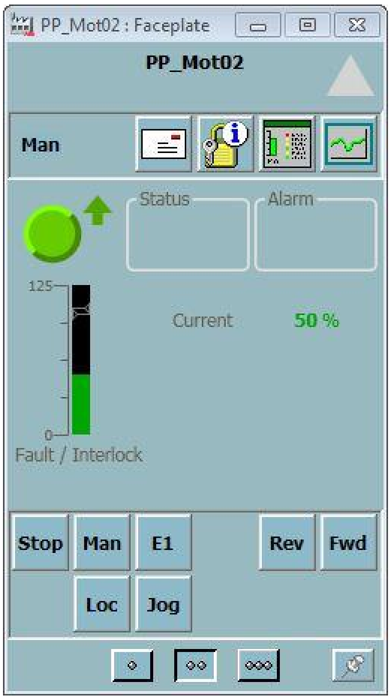
## Reduced Faceplate and Faceplate

**Reduced Faceplate** and **Faceplate** give basic information about the object. Size and contents have been optimized to cover most of the normal process operator action. Process values presented in bargraph are available in **Faceplate**.

**Reduce Faceplate** is the default view.



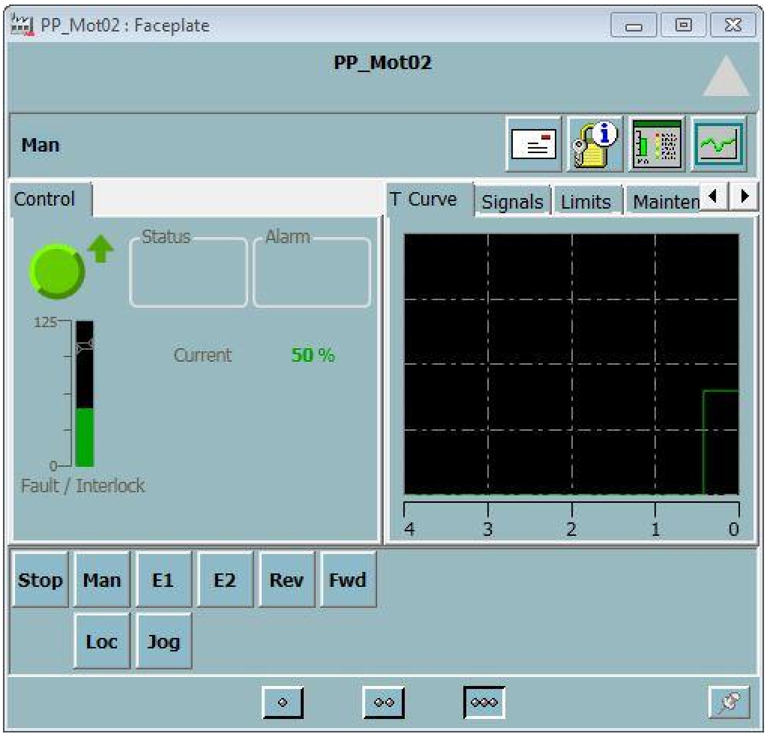
*Reduced Faceplate*



*Faceplate*

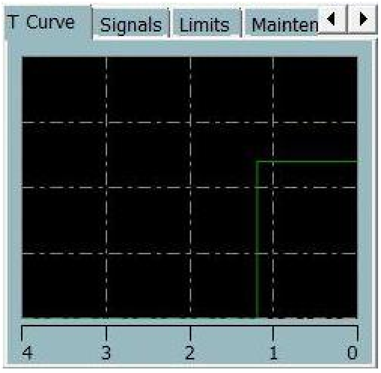
## Extended Faceplate

The extended faceplate gives more detailed information about the object. Additional information about the object is available in tabs.



## 1. T Curve

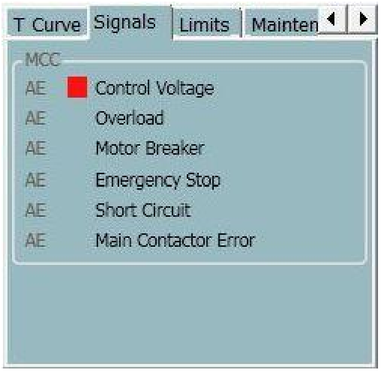
Trim Curve is used to present value in the last 4 minutes period. It contains 1 trace only: **Curr** (for Current in percentage) or **MC** (for Current in unit) depending on current presentation.



## 2. Signals

Fault signal can be configured as no alarm & event (no indication), alarm & event (indicated with **AE**) or event only (indicated with **E**). If alarm is blocked, then it will be indicated with **B**.

When a fault is active, a red square indication is shown in front of the fault signal. It will be indicated with the blinking red square indication as long as not acknowledged.



AE 🟥 Control Voltage
AE Overload
AE Motor Breaker
AE Emergency Stop
AE Short Circuit
AE Main Contactor Error

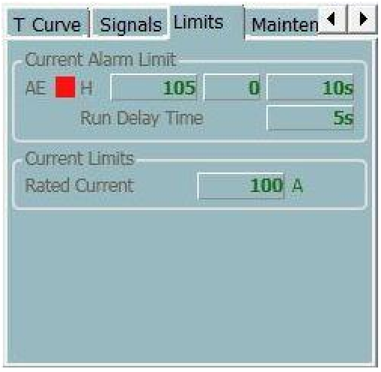
## 3. Limits

Current alarm limit, hysteresis and alarm delay can be defined in this tab. Alarm limit will be visible if configured as alarm & event (indicated with **AE**) or event only (indicated with **E**). If alarm is blocked, then it will be indicated with **B**.

Current will be supervised only after motor running for more than **Start Delay** time. When alarm limit is active, a red square indication is shown in front of the limit. Alarm will be generated if the limit stays active for more than the alarm delay time. It will be indicated with the blinking red square indication as long as not acknowledged.

User with *Tune* permission can change the alarm limit, while hysteresis, alarm and start delay can be adjusted by user with *Configure* permission. The alarm limit entered can be considered as in unit or in percentage depending on the current presentation selected. If the current presentation is to show current in %, then the alarm limit is assumed in %. If the current presentation is to show current in unit, then the alarm limit is assumed in unit as well.

**Rated Current** refers to the current value equal to 100%.



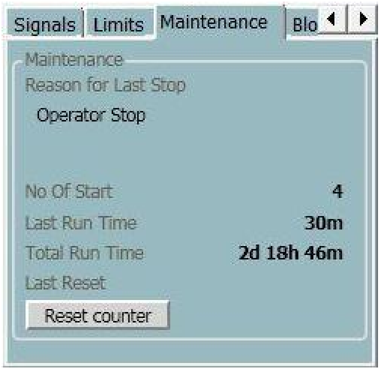
Current Alarm Limit
AE 🟥 H 105 0 10s
Run Delay Time 5s
Current Limits
Rated Current 100 A

## 4. Maintenance

**Reason for Last Stop** shows why the motor is stopped. It can be due to stop command from operator or logic, interlocks or faults.

When activated, counters value will be shown. Otherwise, nothing will be visible. **No Of Start** shows how many times the motor has been successfully started. If motor is running, it shows **Current Run Time**. If motor is stopped, it shows **Last Run Time**. **Total Run Time** shows how long the motor has been running since the last reset.

These values can be reset to zero by pressing the **Reset Counter** button. The button is enabled when motor is not running. **Last Reset** indicates the date and time when the last reset is performed. If Reset has not been performed before, the **Last Reset** will not show any value.



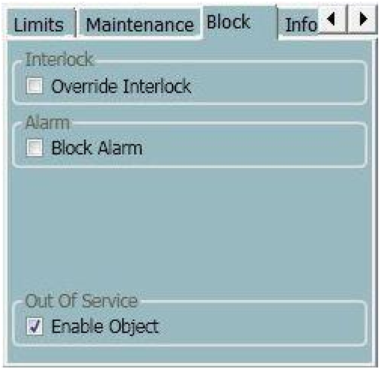
Maintenance
Reason for Last Stop
Operator Stop
No Of Start 4
Last Run Time 30m
Total Run Time 2d 18h 46m
Last Reset

### 5. Block

Checkbox **Override Interlock** will override all blockable interlocks. It is enabled when at least one interlock is configured blockable.

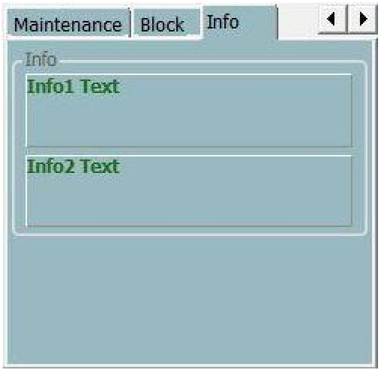
Operator can block the all alarms by ticking on the checkbox **Block Alarm**.

Un-ticking checkbox **Enable Object** will bring the object to **Out Of Service** mode.



### 6. Info

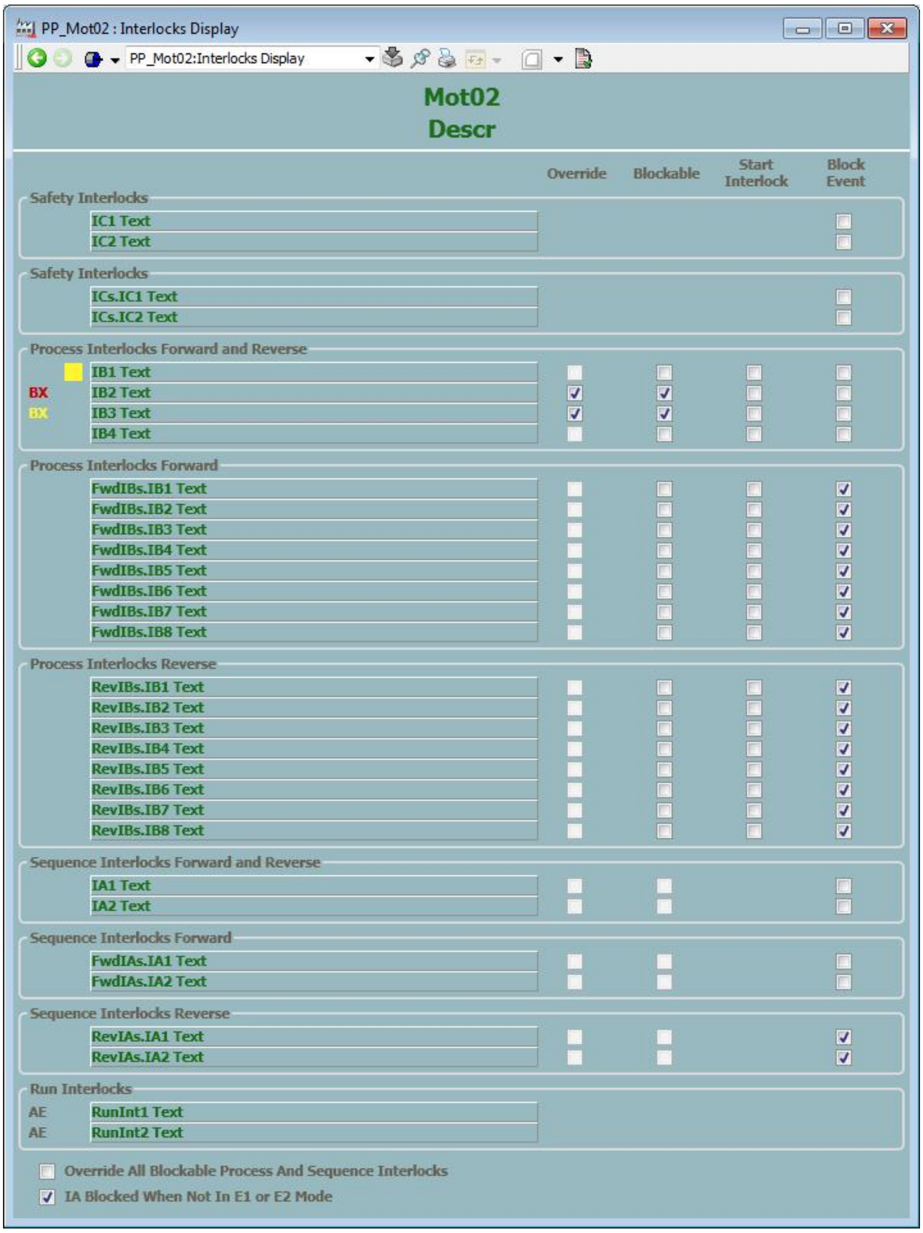
Relevant information about the object can be entered here.



## Interlock Display

**Interlock Display** shows the actual status of all interlocks. Operator can override individual interlock or all interlocks which is blockable. **Blockable**, **Start Interlock**, **Block Event** and **Interlock Texts** are set with *Configure* permission.

When interlock is active, a yellow square indication will be shown in front of the interlock. When operator blocks an interlock, a **BX** text indication will be shown. If the interlock is active, it will be in red color. Otherwise it will be in yellow color.



Mot02
Descr

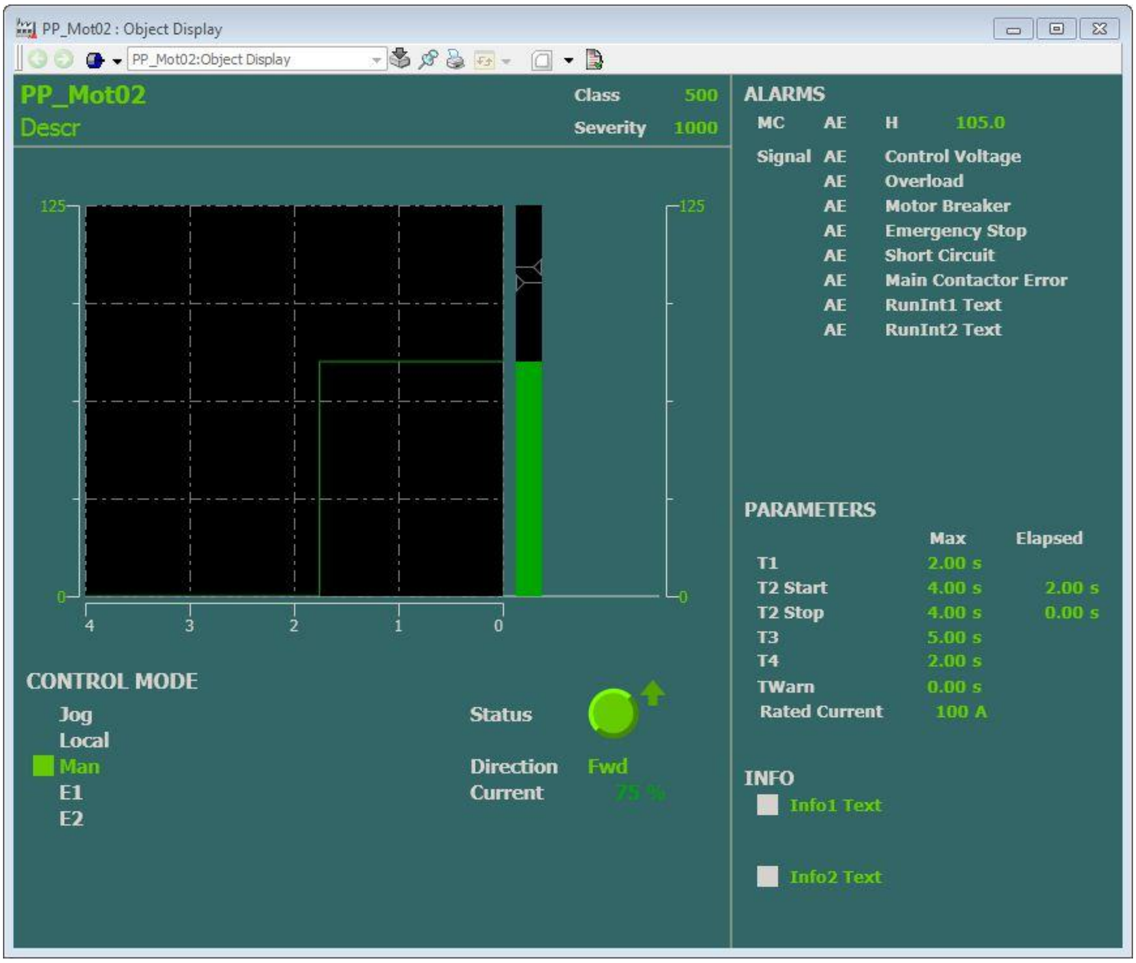
| | | Override | Blockable | Start Interlock | Block Event |
|---|---|---|---|---|---|
| Safety Interlocks | | | | | |
| | IC1 Text | | | | □ |
| | IC2 Text | | | | □ |
| Safety Interlocks | | | | | |
| | ICs.IC1 Text | | | | □ |
| | ICs.IC2 Text | | | | □ |
| Process Interlocks Forward and Reverse | | | | | |
| 🟨 | IB1 Text | □ | □ | □ | □ |
| BX | IB2 Text | ☑ | ☑ | □ | □ |
| BX | IB3 Text | ☑ | ☑ | □ | □ |
| | IB4 Text | □ | □ | □ | □ |
| Process Interlocks Forward | | | | | |
| | FwdIBs.IB1 Text | □ | □ | □ | ☑ |
| | FwdIBs.IB2 Text | □ | □ | □ | ☑ |
| | FwdIBs.IB3 Text | □ | □ | □ | ☑ |
| | FwdIBs.IB4 Text | □ | □ | □ | ☑ |
| | FwdIBs.IB5 Text | □ | □ | □ | ☑ |
| | FwdIBs.IB6 Text | □ | □ | □ | ☑ |
| | FwdIBs.IB7 Text | □ | □ | □ | ☑ |
| | FwdIBs.IB8 Text | □ | □ | □ | ☑ |
| Process Interlocks Reverse | | | | | |
| | RevIBs.IB1 Text | □ | □ | □ | ☑ |
| | RevIBs.IB2 Text | □ | □ | □ | ☑ |
| | RevIBs.IB3 Text | □ | □ | □ | ☑ |
| | RevIBs.IB4 Text | □ | □ | □ | ☑ |
| | RevIBs.IB5 Text | □ | □ | □ | ☑ |
| | RevIBs.IB6 Text | □ | □ | □ | ☑ |
| | RevIBs.IB7 Text | □ | □ | □ | ☑ |
| | RevIBs.IB8 Text | □ | □ | □ | ☑ |
| Sequence Interlocks Forward and Reverse | | | | | |
| | IA1 Text | □ | □ | | □ |
| | IA2 Text | □ | □ | | □ |
| Sequence Interlocks Forward | | | | | |
| | FwdIAs.IA1 Text | □ | □ | | □ |
| | FwdIAs.IA2 Text | □ | □ | | □ |
| Sequence Interlocks Reverse | | | | | |
| | RevIAs.IA1 Text | □ | □ | | ☑ |
| | RevIAs.IA2 Text | □ | □ | | ☑ |
| Run Interlocks | | | | | |
| AE | RunInt1 Text | | | | |
| AE | RunInt2 Text | | | | |

□ Override All Blockable Process And Sequence Interlocks

☑ IA Blocked When Not In E1 or E2 Mode

## Object Display

**Object Display** can be used to get the complete overview of the object. It has detailed information of the motor status, current value and range, signal status, parameters and available control modes.



PP_Mot02 : Object Display

PP_Mot02:Object Display

PP_Mot02 Class 500

Descr Severity 1000

CONTROL MODE

Jog

Local

🟩 Man

E1

E2

Status ●

Direction Fwd

Current 75 %

ALARMS

| | | | |
|---|---|---|---|
| MC | AE | H | 105.0 |
| Signal | AE | Control Voltage | |
| | AE | Overload | |
| | AE | Motor Breaker | |
| | AE | Emergency Stop | |
| | AE | Short Circuit | |
| | AE | Main Contactor Error | |
| | AE | RunInt1 Text | |
| | AE | RunInt2 Text | |

PARAMETERS

| | Max | Elapsed |
|---|---|---|
| T1 | 2.00 s | |
| T2 Start | 4.00 s | 2.00 s |
| T2 Stop | 4.00 s | 0.00 s |
| T3 | 5.00 s | |
| T4 | 2.00 s | |
| TWarn | 0.00 s | |
| Rated Current | 100 A | |

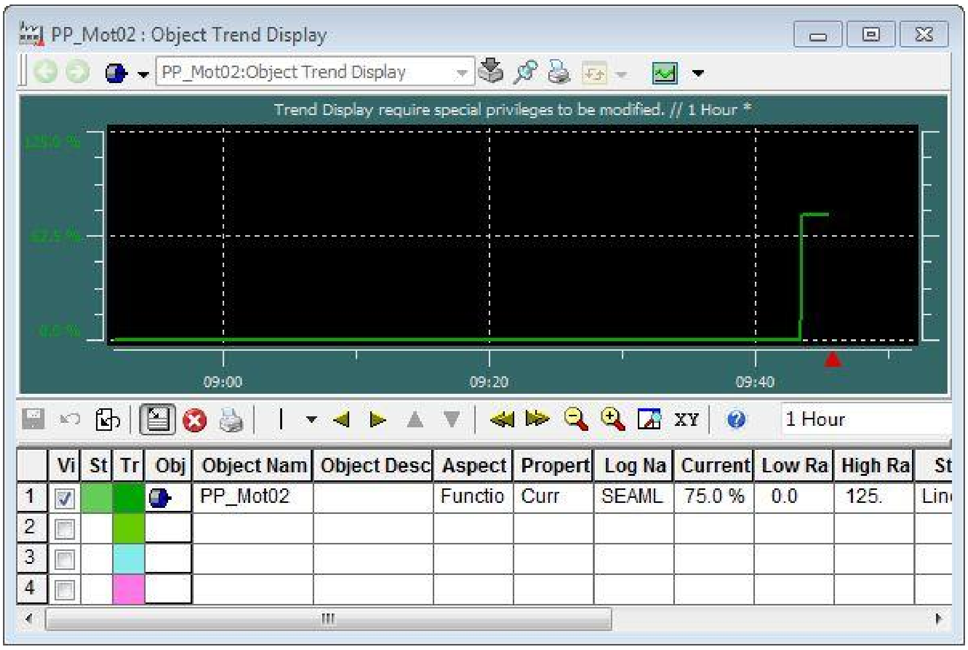
INFO

□ Info1 Text

□ Info2 Text

## Object Trend Display

Values stored can be presented graphically in the form of curves in **Object Trend Display** aspect. By default, Mot02 presents 1 trace only: **Curr**. It uses the trend template **PP Library Motor Object Trend** in Library Structure.



PP_Mot02 : Object Trend Display

PP_Mot02:Object Trend Display

Trend Display require special privileges to be modified. // 1 Hour *

| | Vi | St | Tr | Obj | Object Nam | Object Desc | Aspect | Propert | Log Na | Current | Low Ra | High Ra | St |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | ☑ | 🟩 | 🟩 | | PP_Mot02 | | Functio | Curr | SEAML | 75.0 % | 0.0 | 125. | Lin |
| 2 | □ | | 🟩 | | | | | | | | | | |
| 3 | □ | | 🟦 | | | | | | | | | | |
| 4 | □ | | 🟪 | | | | | | | | | | |

# Alarm & Event

## Alarm & Event List

All alarm & event list aspect is configured to follow the templates defined in **PP Library Alarm & Event List Configurations** in Library Structure.

## Alarm Message

The **Message Description** is hardcoded in the function block and cannot be modified. The **Condition** text is stored in the **AE Translator** aspect in Object Type Structure and support NLS functionality.

| Resource Id | Condition | Message Description |
|---|---|---|
| HC | High Current | Alarm |
| M1 | M1, Control Voltage | Fault |
| M2 | M2, Overload | Fault |
| M3 | M3, Motor Breaker | Fault |
| M4 | M4, Emergency Stop | Fault |
| M5 | M5, Short Circuit | Fault |
| MCE | Main Contactor Error | Fault |
| RInt1 | Run Interlock 1 | Fault |
| RInt2 | Run Interlock 2 | Fault |

## Event Message

The **Message Description** is stored in the **AE Translator** aspect in Object Type Structure and support NLS functionality.

| Resource Id | Message Description |
|---|---|
| AlarmAck | Alarm acknowledged by program |
| AlcBlk | Alarm blocked by program |
| ExtStartFwd | Ext Start Fwd |
| ExtStartRev | Ext Start Rev |
| ExtStop | Ext Stop |
| FwdIAs1Off | FwdIAs.IA1 Off |
| FwdIAs1On | FwdIAs.IA1 On |
| FwdIAs2Off | FwdIAs.IA2 Off |
| FwdIAs2On | FwdIAs.IA2 On |
| FwdIBs1Off | FwdIBs.IB1 Off |
| FwdIBs1On | FwdIBs.IB1 On |
| FwdIBs2Off | FwdIBs.IB2 Off |
| FwdIBs2On | FwdIBs.IB2 On |
| FwdIBs3Off | FwdIBs.IB3 Off |
| FwdIBs3On | FwdIBs.IB3 On |
| FwdIBs4Off | FwdIBs.IB4 Off |
| FwdIBs4On | FwdIBs.IB4 On |
| FwdIBs5Off | FwdIBs.IB5 Off |
| FwdIBs5On | FwdIBs.IB5 On |
| FwdIBs6Off | FwdIBs.IB6 Off |
| FwdIBs6On | FwdIBs.IB6 On |
| FwdIBs7Off | FwdIBs.IB7 Off |
| FwdIBs7On | FwdIBs.IB7 On |
| FwdIBs8Off | FwdIBs.IB8 Off |
| FwdIBs8On | FwdIBs.IB8 On |

| Resource Id | Message Description |
|---|---|
| IA1Off | IA1 Off |
| IA1On | IA1 On |
| IA2Off | IA2 Off |
| IA2On | IA2 On |
| IB1Off | IB1 Off |
| IB1On | IB1 On |
| IB2Off | IB2 Off |
| IB2On | IB2 On |
| IB3Off | IB3 Off |
| IB3On | IB3 On |
| IB4Off | IB4 Off |
| IB4On | IB4 On |
| IC1Off | IC1 Off |
| IC1On | IC1 On |
| IC2Off | IC2 Off |
| IC2On | IC2 On |
| JogEnblOff | JogEnbl Off |
| JogEnblOn | JogEnbl On |
| LEnblOff | LEnbl Off |
| LEnblOn | LEnbl On |
| RevIAs1Off | RevIAs.IA1 Off |
| RevIAs1On | RevIAs.IA1 On |
| RevIAs2Off | RevIAs.IA2 Off |
| RevIAs2On | RevIAs.IA2 On |
| RevIBs1Off | RevIBs.IB1 Off |
| RevIBs1On | RevIBs.IB1 On |
| RevIBs2Off | RevIBs.IB2 Off |
| RevIBs2On | RevIBs.IB2 On |
| RevIBs3Off | RevIBs.IB3 Off |
| RevIBs3On | RevIBs.IB3 On |
| RevIBs4Off | RevIBs.IB4 Off |
| RevIBs4On | RevIBs.IB4 On |
| RevIBs5Off | RevIBs.IB5 Off |
| RevIBs5On | RevIBs.IB5 On |
| RevIBs6Off | RevIBs.IB6 Off |
| RevIBs6On | RevIBs.IB6 On |
| RevIBs7Off | RevIBs.IB7 Off |
| RevIBs7On | RevIBs.IB7 On |
| RevIBs8Off | RevIBs.IB8 Off |
| RevIBs8On | RevIBs.IB8 On |
| SeqE1 | SeqE1 |
| SeqE2 | SeqE2 |
| SeqMan | SeqMan |