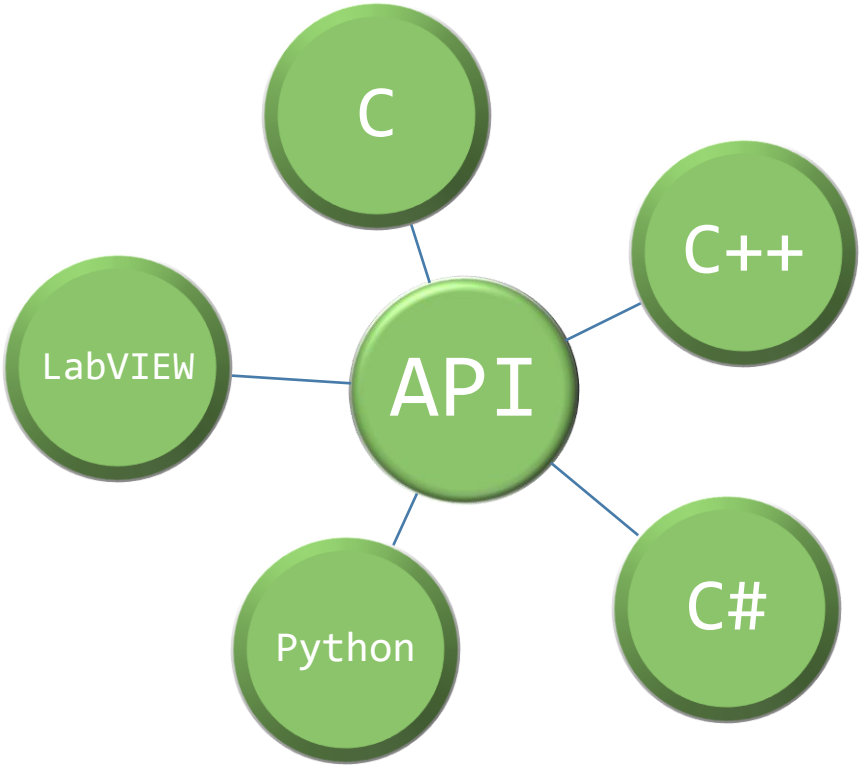
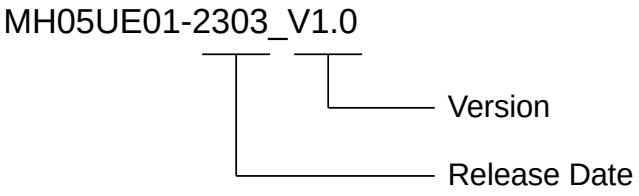
# HIMC
# API Reference Guide



# Revision History

The version of the guide is also indicated on the bottom of the front cover.




| Release Date | Version | Applicable Software Version | Revision Contents |
|---|---|---|---|
| Mar. 15th, 2023 | 1.0 | iA Studio 3.0.0 | (1) HIMC supports CoE communication: Add Read/Write SDO and PDO functions, remove Get/Set Slave Var and Run PDL relative functions.<br>(2) Add chapter/section:<br>■ Chapter 18 – Homing functions<br>(3) Add Functions:<br>■ Chapter 2<br>HIMC_IsSystemInit<br>HIMC_GetECATSt<br>HIMC_GetSlvECATSt<br>HIMC_ScanNetwork<br>■ Chapter 3<br>HIMC_SetOpMode<br>HIMC_SetBufferMode<br>■ Chapter 8<br>HIMC_SetSlvAOHex<br>HIMC_GetSlvAOHex<br>■ Chapter 11<br>HIMC_SetTouchProbeFunc<br>■ Chapter 16<br>HIMC_GetDriveErr<br>(4) Remove chapter 12, Random PT relative functions. |
| Jun. 30th, 2022 | 0.5 | iA Studio 2.0 | (1) Add chapter/section:<br>Chapter 8 – AIO Functions<br>(2) Add API:<br>■ Chapter 2<br>HIMC_SetEconMode |

| Release Date | Version | Applicable Software Version | Revision Contents |
|---|---|---|---|
| | | | HIMC_IsEconMode<br>■ Chapter 3<br>HIMC_MoveTrq<br>HIMC_MovePVT<br>HIMC_IsAcc<br>■ Chapter 6<br>HIMC_ArcAngle2D<br>HIMC_SetGrpAngMotionProfile<br>HIMC_GetGrpCoordTrans<br>HIMC_SetGrpCoordTrans<br>HIMC_GetGrpPoseCmd<br>HIMC_GetGrpPoseFb<br>HIMC_CircleRel<br>■ Chapter 7<br>HIMC_SetGPIInvert<br>HIMC_SetGPOInvert<br>HIMC_BindEMO<br>HIMC_GetAllGPIInvertSt<br>HIMC_GetAllGPOInvertSt<br>■ Chapter 10<br>HIMC_SetPT_PosArray<br>HIMC_SetPT_StateArray<br>HIMC_SetPT_StartIndex<br>HIMC_SetPT_EndIndex<br>■ Chapter 12<br>HIMC_SetCompAlgType<br>■ Chapter 14<br>HIMC_LoadHMPLTask<br>■ Chapter 15<br>HIMC_SetHmplMsgEvtCallback<br>(3) Add variable:<br>Section 3.1.1 Section 6.1.1<br>(4) Add error message:<br>Section 17.1.1, Section 17.1.2,<br>Section 17.1.3 |
| Sep. 15th, 2021 | 0.4 | iA Studio 1.4 | (1) Add chapter/section:<br>Section 7.1.1 –GPIO variables<br>(2) Add API:<br>■ Chapter 2 |

| Release Date | Version | Applicable Software Version | Revision Contents |
|---|---|---|---|
| | | | HIMC_GetFirmwareVer<br>■ Chapter 3<br>HIMC_GetVelFb<br>HIMC_GetVelErr<br>HIMC_GetCurrFb<br>HIMC_SetVelScale<br>HIMC_GetVelScale<br>HIMC_SetRollover<br>HIMC_GetRolloverTurns<br>HIMC_IsDriveErr<br>HIMC_IsPosErr<br>■ Chapter 6<br>HIMC_JogGroup<br>HIMC_JogGroupAxis<br>HIMC_SetGrpVelScale<br>HIMC_GetGrpVelScale<br>■ Chapter 11<br>HIMC_SetupComp3D<br>(3) Add variable:<br>Section 3.1.1, Section 6.1.1, Section 15.1.2<br>(4) Add error message:<br>Section 16.1.1, Section 16.1.2, Section 16.1.3, Section 19.1 |
| Jun. 30th, 2020 | 0.3 | iA Studio 1.3 | (1) Add the corresponding function name for C#, LabVIEW and Python.<br>(2) Add appendix:<br>■ Section 19.1: API error codes<br>(3) Modify API:<br>■ Section 6.2<br>HIMC_GroupReset → HIMC_ResetGroup<br>■ Chapter 17<br>PosTriggerPar: Remove parameter “polarity”<br>(4) Add API:<br>■ Section 3.2<br>HIMC_Halt<br>HIMC_Resume |

| Release Date | Version | Applicable Software Version | Revision Contents |
|---|---|---|---|
| | | | ■ Section 3.3<br>HIMC_SetAccTime<br>HIMC_SetDecTime<br>HIMC_GetKillDec<br>HIMC_SetKillDec<br>HIMC_IgnorePE<br>■ Chapter 4<br>HIMC_GetGearRatio<br>HIMC_IsInGear<br>HIMC_IsGearMaster<br>HIMC_IsGearSlave<br>■ Chapter 5<br>HIMC_GetGantryPairID<br>HIMC_IsGantryPair<br>■ Section 6.2<br>HIMC_HaltGroup<br>HIMC_ResumeGroup<br>HIMC_ArcCW2D<br>HIMC_ArcCCW2D<br>■ Section 6.3<br>HIMC_GetGrpKin<br>HIMC_GetGrpMaxVel<br>HIMC_SetGrpVel<br>HIMC_GetGrpMaxAcc<br>HIMC_SetGrpAcc<br>HIMC_SetGrpAccTime<br>HIMC_GetGrpMaxDec<br>HIMC_SetGrpDec<br>HIMC_SetGrpDecTime<br>HIMC_GetGrpSMTime<br>HIMC_SetGrpSMTime<br>HIMC_GetGrpCoordSys<br>HIMC_SetGrpCoordSys<br>HIMC_GetGrpBufferMode<br>HIMC_SetGrpBufferMode<br>HIMC_GetGrpTransMode<br>HIMC_SetGrpTransMode<br>HIMC_SetGrpTransPrm<br>HIMC_GetGrpCmdNum |

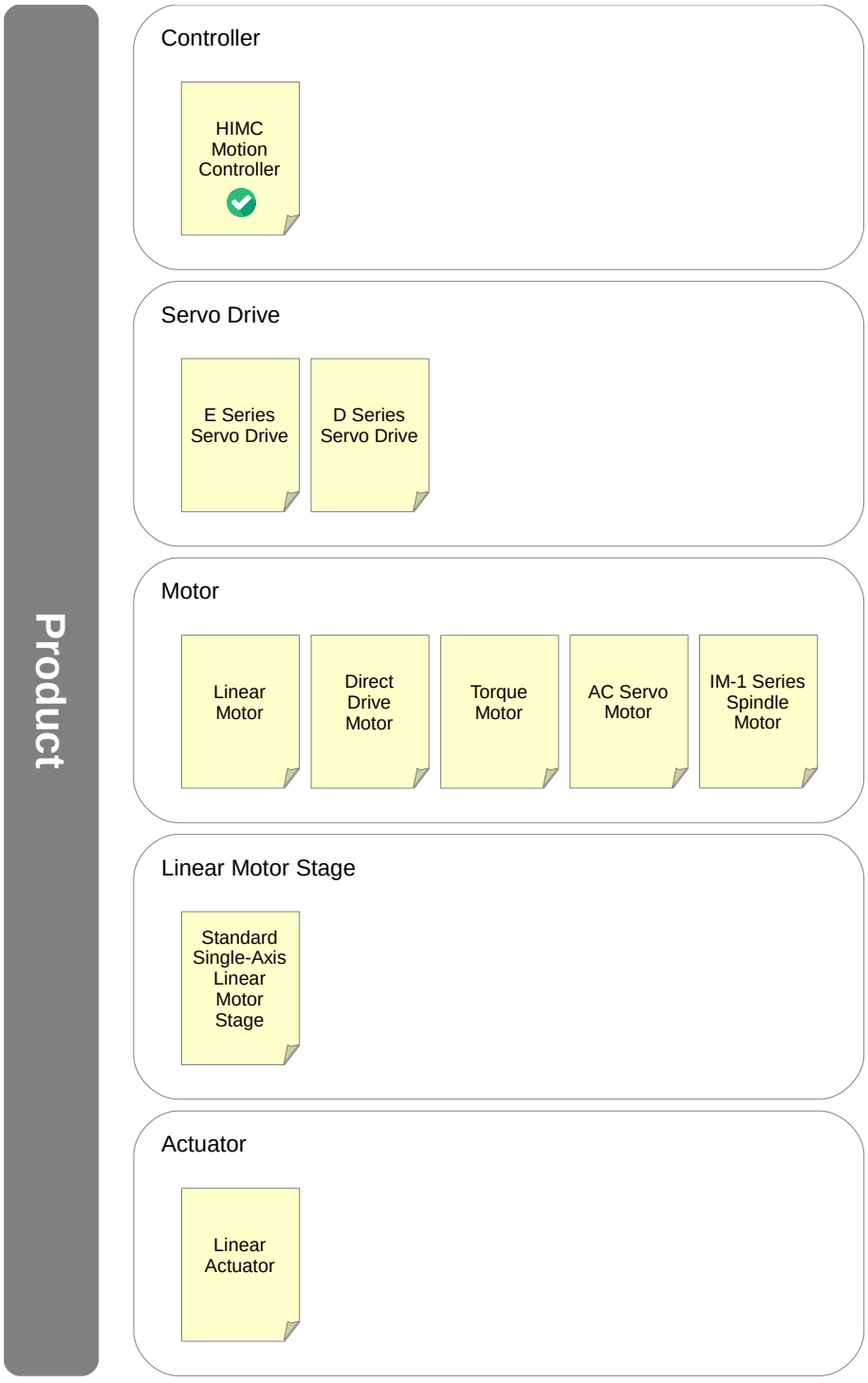
| Release Date | Version | Applicable Software Version | Revision Contents |
|---|---|---|---|
| | | | ■ Section 6.4<br>HIMC_IsGrpErrorStop<br>■ Chapter 11<br>HIMC_GetCompPos<br>■ Section 15.3<br>HIMC_RunSlvPdlFunc<br>HIMC_StopSlvPdlFunc<br>HIMC_IsSlvPdlFuncRunning |
| Apr. 2nd, 2019 | 0.2 | iA Studio 1.1.3772.0 | (1) Modify chapter's arrangement.<br>(2) Add Chapter "Synchronized Motion Functions" and "Filter Functions".<br>(3) Rename API:<br>■ Section 3.2<br>HIMC_GetSwPositiveLimitPos → HIMC_GetSWRL<br>HIMC_SetSwPositiveLimitPos → HIMC_SetSWRL<br>HIMC_GetSwNegativeLimitPos → HIMC_GetSWLL<br>HIMC_SetSwNegativeLimitPos → HIMC_SetSWLL<br>■ Chapter 13<br>HIMC_StartHMPLTask → HIMC_StartTask<br>HIMC_StopHMPLTask → HIMC_StopTask<br>(4) Modify API variables:<br>■ Section 6.1.4<br>HIMC_LinAbs<br>HIMC_LinRel<br>HIMC_CircAbs<br>■ Chapter 9<br>HIMC_SetPosTriggerConfig<br>(5) Add API:<br>■ Chapter 2<br>HIMC_ConnectToSimulator<br>HIMC_ConnectToEthernet<br>HIMC_IsSystemOper<br>HIMC_IsSystemPreOp |

| Release Date | Version | Applicable Software Version | Revision Contents |
|---|---|---|---|
| | | | HIMC_IsSystemError<br>HIMC_StopAll<br>HIMC_RescanMoE<br>■ Section 3.2<br>HIMC_GetPosOffset<br>HIMC_GetPosOut<br>HIMC_GetVelOut<br>HIMC_GetAccOut<br>HIMC_IgnoreHWL<br>HIMC_IgnoreSWL<br>HIMC_GetAxisNum<br>■ Section 6.2<br>HIMC_RemoveAxisFromGrp<br>HIMC_SetGrpKin<br>HIMC_GroupReset<br>■ Chapter 7<br>HIMC_ToggleHimcGpo<br>HIMC_ToggleSlaveGpo<br>■ Chapter 8<br>HIMC_SetTableValue<br>HIMC_GetTableValue<br>■ Chapter 11<br>HIMC_SetupComp2D<br>■ Chapter 13<br>HIMC_StartTaskFunc<br>HIMC_StopAllTask<br>HIMC_IsTaskStop<br>■ Chapter 15<br>HIMC_GetSlvVar<br>HIMC_SetSlvVar<br>HIMC_GetSlvSt<br>HIMC_SetSlvSt<br>■ Chapter 16<br>HIMC_GetAxisLastErr<br>HIMC_ClearAxisLastErr<br>HIMC_GetGrpLastErr<br>HIMC_ClearGrpLastErr<br>(6) Add Enumeration<br>■ Chapter 18 |

| Release Date | Version | Applicable Software Version | Revision Contents |
|---|---|---|---|
| | | | ShaperMode |
| Apr. 17th, 2018 | 0.1 | iA Studio 1.0.2461.0 | First edition. |

# Related Documents

The figure and table of the documents related to the product are shown below. Refer to these documents as required.



| Product | | Doc. Name | Doc. No. | Content |
|---|---|---|---|---|
| Controller | HIMC Motion Controller | HIMC Installation Guide | MH07UE01-□□□□ | Provides detailed information on installing and connecting HIMC motion controller. |
| | | HIMC iA Studio User Guide | MH01UE01-□□□□ | Provides detailed information on the human machine interface operation of HIMC motion controller. |
| | | HIMC Modbus TCP User Guide | MH02UE01-□□□□ | Provides detailed information on the way Modbus TCP communication protocol applied to HIMC motion controller. |
| | | HIMC HMPL User Guide | MH06UE01-□□□□ | Provides detailed information on HMPL library of HIMC motion controller. |
| | | HIMC API Reference Guide | MH05UE01-□□□□ | Provides detailed information on API library of HIMC motion controller. |
| | | HIOM Installation Guide | MH03UE01-□□□□ | Provides detailed information on installing and connecting HIOM (HIWIN mega-ulink IO module). |
| | | ETA3 Installation Guide | MH09UE01-□□□□ | Provides detailed information on installing and connecting ETA3 (HIMC remote module). |
| Servo Drive | E Series Servo Drive | E1 Series Servo Drive User Manual | MD09UE01-□□□□ | Provides detailed information on selecting, installing, connecting, setting, performing test run for, tuning, and monitoring E1 series servo drive. |
| | | E2 Series Servo Drive User Manual | MD28UE01-□□□□ | Provides detailed information on selecting, installing, connecting, setting, performing test run for, tuning, and monitoring E2 series servo drive. |
| | | E1 Series Servo Drive Thunder Software Operation Manual | MD12UE01-□□□□ | Provides detailed information on the human machine interface operation of E1 series servo drive. |
| | | E1 Series Servo Drive EtherCAT(CoE) Communications Command Manual | MD08UE01-□□□□ | Provides detailed information on the way EtherCAT communication protocol applied to E1 series servo drive. |
| | | E1 Series Servo Drive MECHATROLINK-III Communication Command Manual | MD24UE01-□□□□ | Provides detailed information on the way MECHATROLINK-III communication protocol applied to E1 series servo drive. |
| | | E1 Series Servo Drive PROFINET Communication Command Manual | MD02UE01-□□□□ | Provides detailed information on the way PROFINET communication protocol applied to E1 series servo drive. |
| | | E1 Series Servo Drive Gantry Control System User Manual | MD22UE01-□□□□ | Provides detailed information on the usage of E1 series servo drive gantry control system. |
| | | E1 Series Servo Drive Electronic Cam Control System User Manual | MD27UE01-□□□□ | Provides detailed information on the usage of E1 series servo drive electronic cam control system. |
| | | E1 Series Servo Drive Multi-Motion Function User Manual | MD32UE01-□□□□ | Provides detailed information on the usage of E1 series servo drive multi-motion function. |
| | | MPI Library Reference Manual | MD19UE01-□□□□ | Provides detailed information on MPI library of E1 series servo drive and D series servo drive. |
| | | MPI Examples | MD18UE01-□□□□ | Provides detailed information on MPI examples of E1 series servo drive and D series servo drive. |
| | | API Library Reference Manual for Servo Drives | MD23UE01-□□□□ | Provides detailed information on API library of E1 series servo drive and D series servo drive. |
| | | PDL Examples for E1 Series Servo Drive | MD25UE01-□□□□ | Provides detailed information on PDL examples of E1 series servo drive. |

| Product | | Doc. Name | Doc. No. | Content |
|---|---|---|---|---|
| Servo Drive | E Series Servo Drive | Application Note E1 PROFINET Drive Complete Setup with Siemens TIA Portal | MD30UE01-□□□□ | Provides detailed information on the operation of PLC software TIA Portal when E1 PROFINET drive is used with Siemens S7 series PLC. |
| | | Application Note E1 MECHATROLINK-III Drive Complete Setup with YASKAWA MPE720 | MD31UE01-□□□□ | Provides detailed information on the operation of machine controller software MPE720 when E1 MECHATROLINK-III drive is used with YASKAWA MP3000 series machine controller. |
| Servo Drive | D Series Servo Drive | D1 Servo Drive User Manual | MD20UE01-□□□□ | Provides detailed information on selecting, installing, connecting, setting, performing test run for, tuning, and monitoring D1 servo drive. |
| | | D2 Series Servo Drive User Manual | MD07UE01-□□□□ | Provides detailed information on selecting, installing, connecting, setting, performing test run for, tuning, and monitoring D2T servo drive. |
| | | D2T-LM Series Servo Drive User Manual | MD11UE01-□□□□ | Provides detailed information on selecting, installing, connecting, setting, performing test run for, tuning, and monitoring D2T-LM servo drive. |
| | | MPI Library Reference Manual | MD19UE01-□□□□ | Provides detailed information on MPI library of E1 series servo drive and D series servo drive. |
| | | MPI Examples | MD18UE01-□□□□ | Provides detailed information on MPI examples of E1 series servo drive and D series servo drive. |
| | | API Library Reference Manual for Servo Drives | MD23UE01-□□□□ | Provides detailed information on API library of E1 series servo drive and D series servo drive. |
| | | PDL Examples for D-series Drives User Manual | MD13UE01-□□□□ | Provides detailed information on PDL examples of D series servo drive. |
| Motor | Linear Motor | Linear Motor User Manual | MP99UE01-□□□□ | Provides detailed information on selecting, installing, and connecting linear motor. |
| | Direct Drive Motor | DMN Series Direct Drive Motor User Manual | MR01UE01-□□□□ | Provides detailed information on selecting, installing, and connecting DMN series direct drive motor. |
| | | DMT Series Direct Drive Motor User Manual | MR03UE01-□□□□ | Provides detailed information on selecting, installing, and connecting DMT series direct drive motor. |
| | | DMY Series Direct Drive Motor User Manual | MR04UE01-□□□□ | Provides detailed information on selecting, installing, and connecting DMY series direct drive motor. |
| | | DMS Series Direct Drive Motor User Manual | MR05UE01-□□□□ | Provides detailed information on selecting, installing, and connecting DMS series direct drive motor. |
| | | DMR Series Direct Drive Motor User Manual | MR06UE01-□□□□ | Provides detailed information on selecting, installing, and connecting DMR series direct drive motor. |
| | Torque Motor | Torque Motor User Manual | MW99UE01-□□□□ | Provides detailed information on selecting, installing, and connecting torque motor. |
| | AC Servo Motor | AC Servo Motor User Manual | MC03UE01-□□□□ | Provides detailed information on selecting, installing, and connecting AC servo motor. |
| | IM-1 Series Spindle Motor | IM-1 Series Spindle Motor User Manual | MS01UE01-□□□□ | Provides detailed information on selecting and installing IM-1 series spindle motor. |
| Linear Motor | Standard Single-Axis Linear Motor | Standard Single-Axis Linear Motor Stage User Manual | MM06UE01-□□□□ | Provides detailed information on selecting, installing, and connecting |

| Product | | Doc. Name | Doc. No. | Content |
|---|---|---|---|---|
| Stage | Stage | | | standard single-axis linear motor stage. |
| Actuator | Linear Actuator | Linear Actuator User Manual | MA99UE01-□□□□ | Provides detailed information on selecting, installing, and connecting linear actuator. |

# Table of Contents


1. Introduction ..... 1-1
   1.1 How API works ..... 1-2
   1.2 API environment set-up ..... 1-2
      1.2.1 C / C++ ..... 1-2
      1.2.2 C# ..... 1-2
      1.2.3 LabView ..... 1-2
      1.2.4 Python ..... 1-2
   1.3 Version description ..... 1-3
   1.4 Legal disclaimer ..... 1-3

2. HIMC System functions ..... 2-1
   2.1 HIMC_ConnectCtrl ..... 2-2
   2.2 HIMC_ConnectToSimulator ..... 2-3
   2.3 HIMC_ConnectToEthernet ..... 2-4
   2.4 HIMC_DisconnectCtrl ..... 2-5
   2.5 HIMC_RebootController ..... 2-6
   2.6 HIMC_IsSystemInit ..... 2-7
   2.7 HIMC_IsSystemOper ..... 2-8
   2.8 HIMC_IsSystemError ..... 2-9
   2.9 HIMC_GetECATSt ..... 2-10
   2.10 HIMC_GetSlvECATSt ..... 2-11
   2.11 HIMC_DisableAll ..... 2-12
   2.12 HIMC_StopAll ..... 2-13
   2.13 HIMC_EStop ..... 2-14
   2.14 HIMC_GetSlaveNum ..... 2-15
   2.15 HIMC_GetFirmwareVer ..... 2-16
   2.16 HIMC_ScanNetwork ..... 2-17
   2.17 HIMC_SetEconMode ..... 2-18
   2.18 HIMC_ScanNetwork ..... 2-19

3. Axis functions ..... 3-1
   3.1 Overview ..... 3-3
      3.1.1 Axis variables ..... 3-6
   3.2 Axis motion control ..... 3-9
      3.2.1 HIMC_Enable ..... 3-9
      3.2.2 HIMC_Disable ..... 3-10
      3.2.3 HIMC_Reset ..... 3-11
      3.2.4 HIMC_MoveAbs ..... 3-12
      3.2.5 HIMC_MoveRel ..... 3-13

3.2.6 HIMC_MoveVel ..... 3-14
3.2.7 HIMC_MoveTrq ..... 3-15
3.2.8 HIMC_Stop ..... 3-16
3.2.9 HIMC_Halt ..... 3-17
3.2.10 HIMC_Resume ..... 3-18
3.3 Axis setting ..... 3-19
3.3.1 HIMC_GetMaxVel ..... 3-19
3.3.2 HIMC_SetVel ..... 3-20
3.3.3 HIMC_GetMaxAcc ..... 3-21
3.3.4 HIMC_SetAcc ..... 3-22
3.3.5 HIMC_SetAccTime ..... 3-23
3.3.6 HIMC_GetMaxDec ..... 3-24
3.3.7 HIMC_SetDec ..... 3-25
3.3.8 HIMC_SetDecTime ..... 3-26
3.3.9 HIMC_GetKillDec ..... 3-27
3.3.10 HIMC_SetKillDec ..... 3-28
3.3.11 HIMC_GetSWRL ..... 3-29
3.3.12 HIMC_SetSWRL ..... 3-30
3.3.13 HIMC_GetSWLL ..... 3-31
3.3.14 HIMC_SetSWLL ..... 3-32
3.3.15 HIMC_GetSMTime ..... 3-33
3.3.16 HIMC_SetSMTime ..... 3-34
3.3.17 HIMC_GetMoveTime ..... 3-35
3.3.18 HIMC_GetSettlingTime ..... 3-36
3.3.19 HIMC_SetPos ..... 3-37
3.3.20 HIMC_GetPosFb ..... 3-38
3.3.21 HIMC_GetPosOffset ..... 3-39
3.3.22 HIMC_GetPosErr ..... 3-40
3.3.23 HIMC_GetVelFb ..... 3-41
3.3.24 HIMC_GetVelErr ..... 3-42
3.3.25 HIMC_GetCurrFb ..... 3-43
3.3.26 HIMC_GetRefPos ..... 3-44
3.3.27 HIMC_GetRefVel ..... 3-45
3.3.28 HIMC_GetRefAcc ..... 3-46
3.3.29 HIMC_GetPosOut ..... 3-47
3.3.30 HIMC_GetVelOut ..... 3-48
3.3.31 HIMC_GetAccOut ..... 3-49
3.3.32 HIMC_IgnoreHWL ..... 3-50
3.3.33 HIMC_IgnoreSWL ..... 3-51
3.3.34 HIMC_IgnorePE ..... 3-52
3.3.35 HIMC_GetAxisNum ..... 3-53
3.3.36 HIMC_SetVelScale ..... 3-54

3.3.37 HIMC_GetVelScale ..... 3-55
3.3.38 HIMC_SetRollover ..... 3-56
3.3.39 HIMC_GetRolloverTurns ..... 3-57
3.3.40 HIMC_SetOpMode ..... 3-58
3.3.41 HIMC_SetBufferMode ..... 3-59
3.3.42 HIMC_GetMultiAxesFeedbackPos ..... 3-60
3.4 Axis status ..... 3-61
3.4.1 HIMC_IsEnabled ..... 3-61
3.4.2 HIMC_IsMoving ..... 3-62
3.4.3 HIMC_IsInPos ..... 3-63
3.4.4 HIMC_IsErrorStop ..... 3-64
3.4.5 HIMC_IsGantry ..... 3-65
3.4.6 HIMC_IsGrouped ..... 3-66
3.4.7 HIMC_IsSync ..... 3-67
3.4.8 HIMC_IsHWLL ..... 3-68
3.4.9 HIMC_IsHWRL ..... 3-69
3.4.10 HIMC_IsSWLL ..... 3-70
3.4.11 HIMC_IsSWRL ..... 3-71
3.4.12 HIMC_IsDriveErr ..... 3-72
3.4.13 HIMC_IsPosErr ..... 3-73
3.4.14 HIMC_IsCompActive ..... 3-74
3.4.15 HIMC_IsAcc ..... 3-75

4. Synchronized Motion functions ..... 4-1
4.1 Overview ..... 4-2
4.1.1 Synchronized motion variables ..... 4-3
4.2 HIMC_EnableGear ..... 4-4
4.3 HIMC_DisableGear ..... 4-5
4.4 HIMC_GearIn ..... 4-6
4.5 HIMC_GearOut ..... 4-7
4.6 HIMC_GetGearRatio ..... 4-8
4.7 HIMC_IsInGear ..... 4-9
4.8 HIMC_IsGearMaster ..... 4-10
4.9 HIMC_IsGearSlave ..... 4-11

5. Gantry functions ..... 5-1
5.1 Overview ..... 5-2
5.2 HIMC_EnableGantryPair ..... 5-3
5.3 HIMC_DisableGantryPair ..... 5-4
5.4 HIMC_GetGantryPairID ..... 5-5
5.5 HIMC_IsGantryPair ..... 5-6

6. Group functions ..... 6-1
6.1 Overview ..... 6-3
6.1.1 Group variables ..... 6-6
6.1.2 Coordinate systems ..... 6-9
6.1.3 Kinematics ..... 6-13
6.1.4 Buffer modes ..... 6-13
6.1.5 Transition modes ..... 6-15
6.2 Group motion control ..... 6-17
6.2.1 HIMC_EnableGroup ..... 6-17
6.2.2 HIMC_DisableGroup ..... 6-18
6.2.3 HIMC_ResetGroup ..... 6-19
6.2.4 HIMC_StopGroup ..... 6-20
6.2.5 HIMC_HaltGroup ..... 6-21
6.2.6 HIMC_ResumeGroup ..... 6-22
6.2.7 HIMC_JogGroup ..... 6-23
6.2.8 HIMC_JogGroupAxis ..... 6-24
6.2.9 HIMC_LineAbs2D ..... 6-25
6.2.10 HIMC_LineAbs3D ..... 6-26
6.2.11 HIMC_LineRel2D ..... 6-27
6.2.12 HIMC_LineRel3D ..... 6-28
6.2.13 HIMC_Arc2D ..... 6-29
6.2.14 HIMC_ArcCW2D ..... 6-31
6.2.15 HIMC_ArcCCW2D ..... 6-32
6.2.16 HIMC_ArcAngle2D ..... 6-33
6.2.17 HIMC_Circle2D ..... 6-35
6.3 Group setting ..... 6-37
6.3.1 HIMC_AddAxesToGrp ..... 6-37
6.3.2 HIMC_RemoveAxisFromGrp ..... 6-38
6.3.3 HIMC_SetupGroup ..... 6-39
6.3.4 HIMC_UngrpAllAxes ..... 6-40
6.3.5 HIMC_GetGroupID ..... 6-41
6.3.6 HIMC_SetGrpMotionProfile ..... 6-42
6.3.7 HIMC_SetGrpAngMotionProfile ..... 6-44
6.3.8 HIMC_GetGrpKin ..... 6-46
6.3.9 HIMC_SetGrpKin ..... 6-47
6.3.10 HIMC_GetGrpMaxVel ..... 6-48
6.3.11 HIMC_SetGrpVel ..... 6-49
6.3.12 HIMC_GetGrpMaxAcc ..... 6-50
6.3.13 HIMC_SetGrpAcc ..... 6-51
6.3.14 HIMC_SetGrpAccTime ..... 6-52
6.3.15 HIMC_GetGrpMaxDec ..... 6-53
6.3.16 HIMC_SetGrpDec ..... 6-54

6.3.17 HIMC_SetGrpDecTime ..... 6-55
6.3.18 HIMC_GetGrpSMTime ..... 6-56
6.3.19 HIMC_SetGrpSMTime ..... 6-57
6.3.20 HIMC_GetGrpCoordSys ..... 6-58
6.3.21 HIMC_SetGrpCoordSys ..... 6-59
6.3.22 HIMC_GetGrpBufferMode ..... 6-60
6.3.23 HIMC_SetGrpBufferMode ..... 6-61
6.3.24 HIMC_GetGrpTransMode ..... 6-62
6.3.25 HIMC_SetGrpTransMode ..... 6-63
6.3.26 HIMC_SetGrpTransPrm ..... 6-64
6.3.27 HIMC_GetGrpCmdNum ..... 6-66
6.3.28 HIMC_SetGrpVelScale ..... 6-67
6.3.29 HIMC_GetGrpVelScale ..... 6-68
6.3.30 HIMC_GetGrpCoordTrans ..... 6-69
6.3.31 HIMC_SetGrpCoordTrans ..... 6-70
6.3.32 HIMC_GetGrpPoseCmd ..... 6-71
6.3.33 HIMC_GetGrpPoseFb ..... 6-72
6.4 Group status ..... 6-73
6.4.1 HIMC_IsGrpEnabled ..... 6-73
6.4.2 HIMC_IsGrpMoving ..... 6-74
6.4.3 HIMC_IsGrpInPos ..... 6-75
6.4.4 HIMC_IsGrpErrorStop ..... 6-76
6.5 Advanced group motion control ..... 6-77
6.5.1 HIMC_LineAbs ..... 6-77
6.5.2 HIMC_LineRel ..... 6-79
6.5.3 HIMC_CircleAbs ..... 6-81
6.5.4 HIMC_CircleRel ..... 6-83

7. GPIO functions ..... 7-1
7.1 Overview ..... 7-2
7.1.1 GPIO variables ..... 7-2
7.2 Controller IO setting ..... 7-3
7.2.1 HIMC_SetGPO ..... 7-3
7.2.2 HIMC_ToggleGPO ..... 7-4
7.2.3 HIMC_SetAllGPO ..... 7-5
7.2.4 HIMC_SetGPIInvert ..... 7-6
7.2.5 HIMC_SetGPOInvert ..... 7-7
7.2.6 HIMC_BindEMO ..... 7-8
7.3 Slave IO setting ..... 7-9
7.3.1 HIMC_SetSlaveGpo ..... 7-9
7.3.2 HIMC_ToggleSlvGPO ..... 7-10
7.3.3 HIMC_SetSlvAllGPO ..... 7-11

7.4 Controller IO status ..... 7-12
7.4.1 HIMC_GetGPI ..... 7-12
7.4.2 HIMC_GetGPO ..... 7-13
7.4.3 HIMC_GetAllGPI ..... 7-14
7.4.4 HIMC_GetAllGPO ..... 7-15
7.5 Slave IO status ..... 7-16
7.5.1 HIMC_GetSlvGPI ..... 7-16
7.5.2 HIMC_GetSlvGPO ..... 7-17

8. AIO functions ..... 8-1
8.1 Overview ..... 8-2
8.2 Slave AIO setting ..... 8-3
8.2.1 HIMC_SetSlvAIType ..... 8-3
8.2.2 HIMC_SetSlvAOType ..... 8-4
8.2.3 HIMC_SetSlvAOHex ..... 8-5
8.2.4 HIMC_SetSlvAO ..... 8-6
8.3 Slave AIO status ..... 8-7
8.3.1 HIMC_GetSlvAIType ..... 8-7
8.3.2 HIMC_GetSlvAOType ..... 8-8
8.3.3 HIMC_GetSlvAIHex ..... 8-9
8.3.4 HIMC_GetSlvAI ..... 8-10
8.3.5 HIMC_GetSlvAOHex ..... 8-11
8.3.6 HIMC_GetSlvAO ..... 8-12
8.4 Slave AO bound to HIMC internal buffer variable ..... 8-13
8.4.1 HIMC_ SetSlvAOMonitor ..... 8-13
8.4.2 HIMC_SetSlvAOParam ..... 8-15
8.4.3 HIMC_GetSlvAOScale ..... 8-17
8.4.4 HIMC_GetSlvAOOffset ..... 8-18
8.4.5 HIMC_IsSlvAOBound ..... 8-19

9. User Table functions ..... 9-1
9.1 Overview ..... 9-2
9.2 HIMC_SetUserTable ..... 9-3
9.3 HIMC_GetUserTable ..... 9-4
9.4 HIMC_SetTableValue ..... 9-5
9.5 HIMC_GetTableValue ..... 9-6
9.6 HIMC_SaveUserTable ..... 9-7
9.7 HIMC_LoadUserTable ..... 9-8

10. Position Trigger functions ..... 10-1
10.1 Overview ..... 10-2
10.1.1 PT variables ..... 10-2

10.1.2 Flow of using PT function ..... 10-4
10.2 HIMC_EnablePT ..... 10-5
10.3 HIMC_DisablePT ..... 10-6
10.4 HIMC_IsPTEnabled ..... 10-7
10.5 HIMC_SetPosTriggerConfig ..... 10-8

11. Touch Probe functions ..... 11-1
11.1 Overview ..... 11-2
11.2 HIMC_EnableTouchProbe ..... 11-3
11.3 HIMC_DisableTouchProbe ..... 11-4
11.4 HIMC_IsTouchProbeEnabled ..... 11-5
11.5 HIMC_IsTouchProbeTriggered ..... 11-6
11.6 HIMC_GetTouchProbePos ..... 11-7
11.7 HIMC_SetTouchProbeFunc ..... 11-8

12. Dynamic Error Compensation functions ..... 12-1
12.1 Overview ..... 12-2
12.2 HIMC_EnableComp ..... 12-4
12.3 HIMC_DisableComp ..... 12-5
12.4 HIMC_SetupComp ..... 12-6
12.5 HIMC_SetupComp2D ..... 12-8
12.6 HIMC_SetupComp3D ..... 12-10
12.7 HIMC_GetCompPos ..... 12-12
12.8 HIMC_SetCompAlgType ..... 12-13

13. Filter functions ..... 13-1
13.1 Overview ..... 13-2
13.2 HIMC_EnableAxisVsf ..... 13-3
13.3 HIMC_DisableAxisVsf ..... 13-4
13.4 HIMC_SetAxisVsf ..... 13-5
13.5 HIMC_EnableAxisInShape ..... 13-7
13.6 HIMC_DisableAxisInShape ..... 13-8
13.7 HIMC_SetAxisInShape ..... 13-9
13.8 HIMC_EnableGrpInShape ..... 13-11
13.9 HIMC_DisableGrpInShape ..... 13-12
13.10 HIMC_SetGrpInShape ..... 13-13

14. HMPL Task functions ..... 14-1
14.1 Overview ..... 14-2
14.2 HIMC_StartTask ..... 14-3
14.3 HIMC_StartTaskFunc ..... 14-4
14.4 HIMC_StopTask ..... 14-5

14.5 HIMC_StopAllTask ..... 14-6
14.6 HIMC_IsTaskStop ..... 14-7
14.7 HIMC_LoadHMPLTask ..... 14-8

15. Callback functions ..... 15-1
15.1 HIMC_SetHmplEvtCallback ..... 15-2
15.2 HIMC_SetErrorCallback ..... 15-3
15.3 HIMC_SetHmplMsgEvtCallback ..... 15-4

16. Variable and Function Operation functions ..... 16-1
16.1 Overview ..... 16-2
16.1.1 Controller variables list ..... 16-3
16.2 Servo drive variable operation ..... 16-7
16.2.1 HIMC_ReadSDO ..... 16-7
16.2.2 HIMC_WriteSDO ..... 16-9
16.2.3 HIMC_ReadPDO ..... 16-10
16.2.4 HIMC_WritePDO ..... 16-11
16.2.5 HIMC_ForceWritePDO ..... 16-12
16.2.6 HIMC_ReleasePDO ..... 16-13
16.3 Controller variable operation ..... 16-14
16.3.1 HIMC_GetVariableByID ..... 16-14
16.3.2 HIMC_SetVariableByID ..... 16-15
16.3.3 HIMC_GetVariableListByID ..... 16-16
16.3.4 HIMC_SetVariableListByID ..... 16-17
16.3.5 HIMC_GetGlobalVariables ..... 16-18
16.3.6 HIMC_SetGlobalVariables ..... 16-19

17. HIMC Error functions ..... 17-1
17.1 Overview ..... 17-2
17.1.1 System error messages ..... 17-3
17.1.2 Axis error massages ..... 17-6
17.1.3 Group error messages ..... 17-9
17.2 HIMC_GetLastError ..... 17-11
17.3 HIMC_GetAxisLastErr ..... 17-12
17.4 HIMC_ClearAxisLastErr ..... 17-13
17.5 HIMC_GetGrpLastErr ..... 17-14
17.6 HIMC_ClearGrpLastErr ..... 17-15
17.7 HIMC_GetDriveErr ..... 17-16
17.8 HIMC_GetErrorInformation ..... 17-17

18. Homing functions ..... 18-1
18.1 Overview ..... 18-2

18.2 HIMC_MoveHome ..... 18-9
18.3 HIMC_SetHomeMethod ..... 18-10
18.4 HIMC_SetHomeSwitchVel ..... 18-12
18.5 HIMC_SetHomeZeroVel ..... 18-13
18.6 HIMC_SetHomeAcc ..... 18-14
18.7 HIMC_SetHomeOffset ..... 18-15
18.8 HIMC_SetHomeTimeout ..... 18-16
18.9 HIMC_IsHomed ..... 18-17
18.10 HIMC_IsHoming ..... 18-18

19. Data structures ..... 19-1
19.1 ComInfo ..... 19-2
19.2 CoordPosition ..... 19-3
19.3 MotionProfile ..... 19-4
19.4 CenterPosition ..... 19-5
19.5 NormalVector ..... 19-6
19.6 TransPrm ..... 19-7
19.7 PosTriggerPar ..... 19-8

20. Enumerations ..... 20-1
20.1 ComType ..... 20-2
20.2 CoordSystem ..... 20-3
20.3 MotionBufferMode ..... 20-5
20.4 MotionTransitionMode ..... 20-6
20.5 ShaperMode ..... 20-7

21. Appendix ..... 21-1
21.1 API error codes ..... 21-2

(This page is intentionally left blank.)

# 1. Introduction


1. Introduction ..... 1-1
   1.1 How API works ..... 1-2
   1.2 API environment set-up ..... 1-2
      1.2.1 C / C++ ..... 1-2
      1.2.2 C# ..... 1-2
      1.2.3 LabView ..... 1-2
      1.2.4 Python ..... 1-2
   1.3 Version description ..... 1-3
   1.4 Legal disclaimer ..... 1-3

## 1.1 How API works

The functions introduced in this reference guide are mainly used in C language. For the corresponding name of other API environment, please refer to the description in each section.

**Note:**

**The unit of motion variables adopted by iA Studio 1.3 (included) and above:**

**linear motion (mm), rotary motion (deg), time (ms)**

**The unit of motion variables adopted by iA Studio 1.2 (included) and below:**

**linear motion (m), rotary motion (rad), time (s)**

## 1.2 API environment set-up

### 1.2.1 C / C++

1. Go to <installation directory> \documents\Software Guides\API_examples\cpp\vs_project.
2. Execute `copy_required_file.bat`.
3. Open project file `api_example.sln`.

### 1.2.2 C#

1. Go to <installation directory> \documents\Software Guides\API_examples\c_sharp.
2. Execute `copy_required_file.bat`.
3. Open project file `api_example.sln`.

### 1.2.3 LabView

1. Go to <installation directory> \documents\Software Guides\API_examples\labview.
2. Execute `copy_required_file.bat`.
3. Open project file `example.lvproj`.

### 1.2.4 Python

1. Go to <installation directory> \documents\Software Guides\API_examples\python.
2. Execute `copy_required_file.bat`.
3. Directly execute example `python example.py`.

## 1.3 Version description

When HIMC controller is applied with software version iA Studio 3.0 (included) and above, it supports HIMC controller (product model: MC-XX-XX-01-XX) with CoE communication function, but it is not compatible with HIMC controller (product model: MC-XX-XX-00-XX) with MoE communication function. Users must adopt software version iA Studio 2.X (included) and below when using HIMC controller with MoE communication function.

The unit of motion variables adopted by iA Studio 1.3 (included) and above:
linear motion (mm), rotary motion (deg), time (ms)
The unit of motion variables adopted by iA Studio 1.2 (included) and below:
linear motion (m), rotary motion (rad), time (s)

## 1.4 Legal disclaimer

Users can adopt or modify any of the sample codes provided in this guide for specific uses. However, the correctness, effectiveness and safety cannot be guaranteed in different application scenarios. Users should take full responsibility for the safety and the effectiveness of the software implementations.

(This page is intentionally left blank.)

# 2. HIMC System functions


2. HIMC System functions ..... 2-1
- 2.1 HIMC_ConnectCtrl ..... 2-2
- 2.2 HIMC_ConnectToSimulator ..... 2-3
- 2.3 HIMC_ConnectToEthernet ..... 2-4
- 2.4 HIMC_DisconnectCtrl ..... 2-5
- 2.5 HIMC_RebootController ..... 2-6
- 2.6 HIMC_IsSystemInit ..... 2-7
- 2.7 HIMC_IsSystemOper ..... 2-8
- 2.8 HIMC_IsSystemError ..... 2-9
- 2.9 HIMC_GetECATSt ..... 2-10
- 2.10 HIMC_GetSlvECATSt ..... 2-11
- 2.11 HIMC_DisableAll ..... 2-12
- 2.12 HIMC_StopAll ..... 2-13
- 2.13 HIMC_EStop ..... 2-14
- 2.14 HIMC_GetSlaveNum ..... 2-15
- 2.15 HIMC_GetFirmwareVer ..... 2-16
- 2.16 HIMC_ScanNetwork ..... 2-17
- 2.17 HIMC_SetEconMode ..... 2-18
- 2.18 HIMC_ScanNetwork ..... 2-19

## 2.1 HIMC_ConnectCtrl

### Purpose

To connect to the controller.

### Syntax

```
int HIMC_ConnectCtrl(
    const ComInfo com_info,
    int *p_ctrl_id
);
```

### Parameter

com_info [in] A structure to store the information of the connection.

p_ctrl_id [out] A pointer to the buffer to receive successfully connected controller ID.
Users can use this ID on other API functions to operate this controller.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

It is recommended to use HIMC_ConnectToSimulator or HIMC_ConnectToEthernet to connect to HIMC.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ConnectCtrl |
| LabVIEW | -- |
| Python | -- |

## 2.2 HIMC_ConnectToSimulator

### Purpose

To connect to the simulator.

### Syntax

```
int HIMC_ConnectToSimulator(
    int *p_ctrl_id
);
```

### Parameter

p_ctrl_id [out] A pointer to the buffer to receive successfully connected controller ID.
Users can use this ID on other API functions to operate this controller.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ConnectToSimulator |
| LabVIEW | HIMC Connect To Simulator.vi |
| Python | Declare HimcAPI("Simulator") |

## 2.3 HIMC_ConnectToEthernet

### Purpose

To connect to Ethernet.

### Syntax

```
int HIMC_ConnectToEthernet(
    const char *ip_address,
    const char *port,
    int *p_ctrl_id
);
```

### Parameter

| | |
|---|---|
| ip_address [in] | The string of IP address. |
| port [in] | The string of port. |
| p_ctrl_id [out] | A pointer to the buffer to receive successfully connected controller ID.<br>Users can use this ID on other API functions to operate this controller. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ConnectToEthernet |
| LabVIEW | HIMC Connect To Ethernet.vi |
| Python | Declare HimcAPI("Ethernet") |

## 2.4 HIMC_DisconnectCtrl

### Purpose

To disconnect from the controller.

### Syntax

```
int HIMC_DisconnectCtrl(
    int ctrl_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_DisconnectCtrl |
| LabVIEW | HIMC Disconnect Ctrl.vi |
| Python | Disconnect when class is deleted |

## 2.5 HIMC_RebootController

### Purpose

To reboot the controller.

### Syntax

```
int HIMC_RebootController(
    int ctrl_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_RebootController |
| LabVIEW | HIMC Reboot Controller.vi |
| Python | RebootController |

## 2.6 HIMC_IsSystemInit

### Purpose

To query whether HIMC system is at “initializing” state.

### Syntax

```
int HIMC_IsSystemInit(
    int ctrl_id,
    int *p_is_init
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| p_is_init [out] | A pointer to the buffer to receive the “initializing” state of HIMC system.<br>If HIMC system is at “IsSystemInit” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsSystemInit |
| LabVIEW | HIMC Is System Init.vi |
| Python | IsSystemInit |

## 2.7 HIMC_IsSystemOper

**Purpose**

To query whether HIMC system is at “operation” state. If it is, the communication between HIMC and the slaves is established.

**Syntax**

```
int HIMC_IsSystemOper(
    int ctrl_id,
    int *p_is_op
);
```

**Parameter**

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

p_is_op [out] A pointer to the buffer to receive the “operation” state of HIMC system.
If HIMC system is at “IsSystemOper” state, the value will be 1. Otherwise, it will be 0.

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_IsSystemOper |
| LabVIEW | HIMC Is System Oper.vi |
| Python | IsSystemOper |

## 2.8 HIMC_IsSystemError

### Purpose

To query whether HIMC system is at “error” state.

### Syntax

```
int HIMC_IsSystemError(
    int ctrl_id,
    int *p_is_error
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

p_is_error [out] A pointer to the buffer to receive the “error” state of HIMC system.
If HIMC system is at “SystemError” state, the value will be 1. Otherwise, it will be 0.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsSystemError |
| LabVIEW | HIMC Is System Error.vi |
| Python | IsSystemError |

# 2.9 HIMC_GetECATSt

## Purpose

To get EtherCAT State Machine of the controller.

## Syntax

```
int HIMC_GetECATSt(
    int ctrl_id,
    int * p_state
);
```

## Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

p_state [out] A pointer to the buffer, to recieve EtherCAT State Machine of the controller.

1：Init
2：Pre-Operation
4：Safe-Operation
8：Operation

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Remark

The motion queues of the axes and the axis groups will be cleared.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetECATSt |
| LabVIEW | HIMC Get ECAT St.vi |
| Python | GetECATSt |

## 2.10 HIMC_GetSlvECATSt

### Purpose

To get EtherCAT State Machine of the slave.

### Syntax

```
int HIMC_GetSlvECATSt(
    int ctrl_id,
    int slv_id
    int *p_state
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_id [in] | Slave index. |
| p_state [out] | A pointer to the buffer, to recieve EtherCAT State Machine of the slave.<br>1：Init<br>2：Pre-Operation<br>4：Safe-Operation<br>8：Operation |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

The motion queues of the axes and the axis groups will be cleared.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ GetSlvECATSt |
| LabVIEW | HIMC Get Slv ECAT St.vi |
| Python | GetSlvECATSt |

## 2.11 HIMC_DisableAll

### Purpose

To disable all axes and all axis groups.

### Syntax

```
int HIMC_DisableAll(
    int ctrl_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

The motion queues of the axes and the axis groups will be cleared.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_DisableAll |
| LabVIEW | HIMC Disable All.vi |
| Python | DisableAll |

## 2.12 HIMC_StopAll

### Purpose

To stop all axes and all axis groups with kill deceleration and make them stay at “enable” status.

### Syntax

```
int HIMC_StopAll(
    int ctrl_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

The motion queues of the axes and the axis groups will be cleared.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_StopAll |
| LabVIEW | HIMC Stop All.vi |
| Python | StopAll |

## 2.13 HIMC_EStop

### Purpose

To stop all the execution programs in the controller (including all HMPL tasks) and disable all axes and all axis groups.

### Syntax

```
int HIMC_EStop(
    int ctrl_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_EStop |
| LabVIEW | HIMC E Stop.vi |
| Python | EStop |

## 2.14 HIMC_GetSlaveNum

### Purpose

To get the number of the slaves that are connected to the controller.

### Syntax

```
int HIMC_GetSlaveNum(
    int ctrl_id,
    int *p_slv_num
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| p_slv_num [out] | A pointer to the buffer to receive the number of the slaves that are connected to the controller. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSlaveNum |
| LabVIEW | HIMC Get Slave Num.vi |
| Python | GetSlaveNum |

## 2.15 HIMC_GetFirmwareVer

### Purpose

To get the firmware version of the controller.

### Syntax

```
int HIMC_GetFirmwareVer(
    int ctrl_id,
    char *p_ver_buf
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| p_ver_buf [out] | A pointer to the buffer to receive the returned string of the firmware version. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetFirmwareVer |
| LabVIEW | HIMC Get Firmware Ver.vi |
| Python | GetFirmwareVer |

## 2.16 HIMC_ScanNetwork

### Purpose

To rescan the connection between the controller and the slaves.

### Syntax

```
int HIMC_ScanNetwork(
    int ctrl_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ScanNetwork |
| LabVIEW | HIMC Scan Network.vi |
| Python | ScanNetwork |

# 2.17 HIMC_SetEconMode

**Purpose**

To set HIMC API performance mode.

**Syntax**

```
int HIMC_SetEconMode(
    int ctrl_id,
    bool mode
);
```

**Parameter**

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

mode [in] HIMC API performance mode.
0: High Performance Mode (default)
1: Econ Mode

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

(1) High Performance Mode has faster HIMC API average response velocity, but its CPU usage is higher than that of Econ Mode.
(2) Econ Mode can reduce CPU usage, but it will increase HIMC API average response time.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetEconMode |
| LabVIEW | HIMC Set Econ Mode.vi |
| Python | SetEconMode |

## 2.18 HIMC_ScanNetwork

### Purpose

To query whether HIMC API performance mode is Econ Mode.

### Syntax

```
int HIMC_ScanNetwork(
    int ctrl_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

p_is_econ [in] A pointer to the buffer to receive HIMC API performance mode.
If it is Econ Mode, the value will be 1. Otherwise, it will be 0.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ScanNetwork |
| LabVIEW | HIMC Scan Network.vi |
| Python | ScanNetwork |

(This page is intentionally left blank.)

# 3. Axis functions


3. Axis functions ..... 3-1
3.1 Overview ..... 3-3
3.1.1 Axis variables ..... 3-6
3.2 Axis motion control ..... 3-9
3.2.1 HIMC_Enable ..... 3-9
3.2.2 HIMC_Disable ..... 3-10
3.2.3 HIMC_Reset ..... 3-11
3.2.4 HIMC_MoveAbs ..... 3-12
3.2.5 HIMC_MoveRel ..... 3-13
3.2.6 HIMC_MoveVel ..... 3-14
3.2.7 HIMC_MoveTrq ..... 3-15
3.2.8 HIMC_Stop ..... 3-16
3.2.9 HIMC_Halt ..... 3-17
3.2.10 HIMC_Resume ..... 3-18
3.3 Axis setting ..... 3-19
3.3.1 HIMC_GetMaxVel ..... 3-19
3.3.2 HIMC_SetVel ..... 3-20
3.3.3 HIMC_GetMaxAcc ..... 3-21
3.3.4 HIMC_SetAcc ..... 3-22
3.3.5 HIMC_SetAccTime ..... 3-23
3.3.6 HIMC_GetMaxDec ..... 3-24
3.3.7 HIMC_SetDec ..... 3-25
3.3.8 HIMC_SetDecTime ..... 3-26
3.3.9 HIMC_GetKillDec ..... 3-27
3.3.10 HIMC_SetKillDec ..... 3-28
3.3.11 HIMC_GetSWRL ..... 3-29
3.3.12 HIMC_SetSWRL ..... 3-30
3.3.13 HIMC_GetSWLL ..... 3-31
3.3.14 HIMC_SetSWLL ..... 3-32
3.3.15 HIMC_GetSMTime ..... 3-33
3.3.16 HIMC_SetSMTime ..... 3-34
3.3.17 HIMC_GetMoveTime ..... 3-35
3.3.18 HIMC_GetSettlingTime ..... 3-36
3.3.19 HIMC_SetPos ..... 3-37
3.3.20 HIMC_GetPosFb ..... 3-38
3.3.21 HIMC_GetPosOffset ..... 3-39
3.3.22 HIMC_GetPosErr ..... 3-40
3.3.23 HIMC_GetVelFb ..... 3-41

3.3.24 HIMC_GetVelErr ..... 3-42
3.3.25 HIMC_GetCurrFb ..... 3-43
3.3.26 HIMC_GetRefPos ..... 3-44
3.3.27 HIMC_GetRefVel ..... 3-45
3.3.28 HIMC_GetRefAcc ..... 3-46
3.3.29 HIMC_GetPosOut ..... 3-47
3.3.30 HIMC_GetVelOut ..... 3-48
3.3.31 HIMC_GetAccOut ..... 3-49
3.3.32 HIMC_IgnoreHWL ..... 3-50
3.3.33 HIMC_IgnoreSWL ..... 3-51
3.3.34 HIMC_IgnorePE ..... 3-52
3.3.35 HIMC_GetAxisNum ..... 3-53
3.3.36 HIMC_SetVelScale ..... 3-54
3.3.37 HIMC_GetVelScale ..... 3-55
3.3.38 HIMC_SetRollover ..... 3-56
3.3.39 HIMC_GetRolloverTurns ..... 3-57
3.3.40 HIMC_SetOpMode ..... 3-58
3.3.41 HIMC_SetBufferMode ..... 3-59
3.3.42 HIMC_GetMultiAxesFeedbackPos ..... 3-60
3.4 Axis status ..... 3-61
3.4.1 HIMC_IsEnabled ..... 3-61
3.4.2 HIMC_IsMoving ..... 3-62
3.4.3 HIMC_IsInPos ..... 3-63
3.4.4 HIMC_IsErrorStop ..... 3-64
3.4.5 HIMC_IsGantry ..... 3-65
3.4.6 HIMC_IsGrouped ..... 3-66
3.4.7 HIMC_IsSync ..... 3-67
3.4.8 HIMC_IsHWLL ..... 3-68
3.4.9 HIMC_IsHWRL ..... 3-69
3.4.10 HIMC_IsSWLL ..... 3-70
3.4.11 HIMC_IsSWRL ..... 3-71
3.4.12 HIMC_IsDriveErr ..... 3-72
3.4.13 HIMC_IsPosErr ..... 3-73
3.4.14 HIMC_IsCompActive ..... 3-74
3.4.15 HIMC_IsAcc ..... 3-75

# 3.1 Overview

HIMC provides single axis motion commands, including point-to-point (P2P), JOG and synchronized motion. Users can use the related motion functions based on application and requirement. Figure 3.1.1 is the flow diagram of HIMC axis motion control. Reference position will be generated after motion command goes through the built-in profile generator (PG), and the position output for servo drive will be generated after the output reference position adds error compensation value.

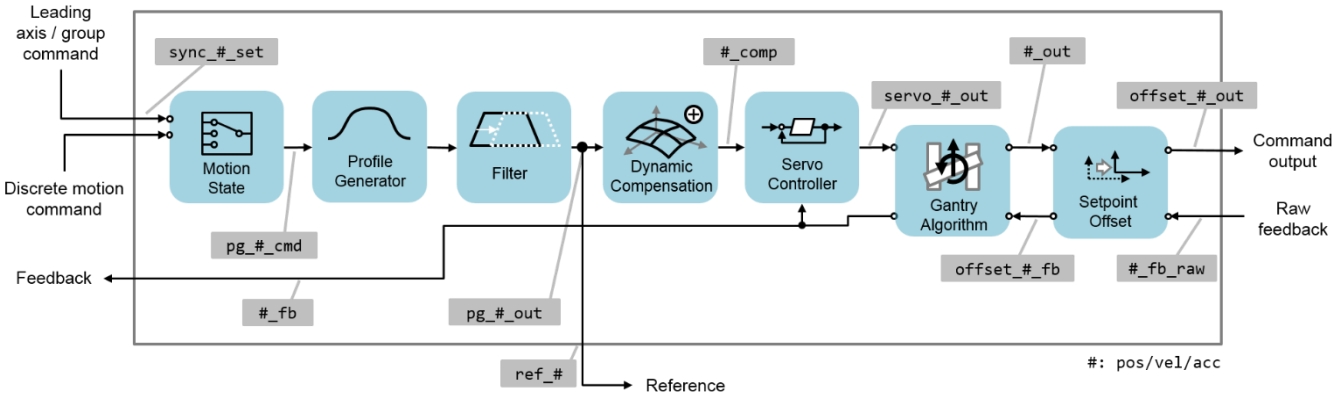



Figure 3.1.1

HIMC's profile generator has built-in S-Curve velocity planning, as Figure 3.1.2 shows. Users can set profile generator's maximum velocity, maximum acceleration, maximum deceleration, and smooth time.

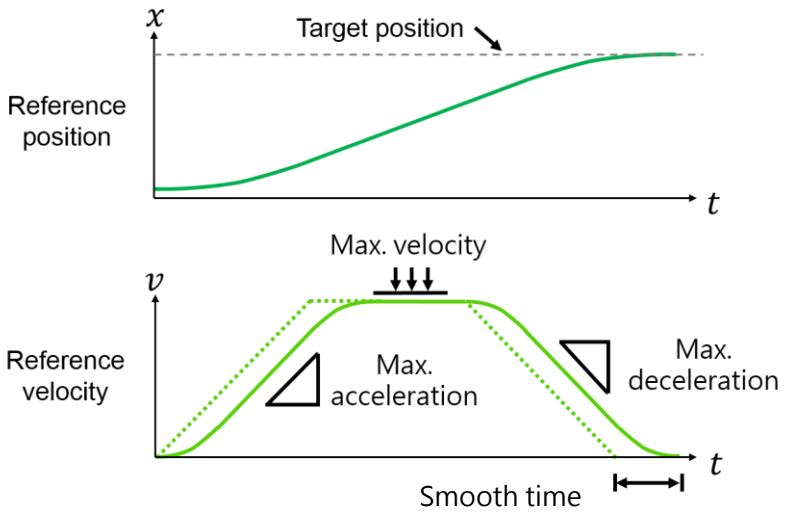



Figure 3.1.2

As Figure 3.1.3 shows, adding smooth time will delay the reference velodity, but it can effectively lower the jerk generated by high acceleration to increase the stability of system. The relationship of jerk, maximum acceleration and smooth time is shown as follows.

Jerk = Maximum acceleration / Smooth time (Ts)

As for total acceleration time, it can be obtained via the following formula.

Total acceleration time (T) = T-Curve acceleration time (Ta) + Smooth time (Ts)

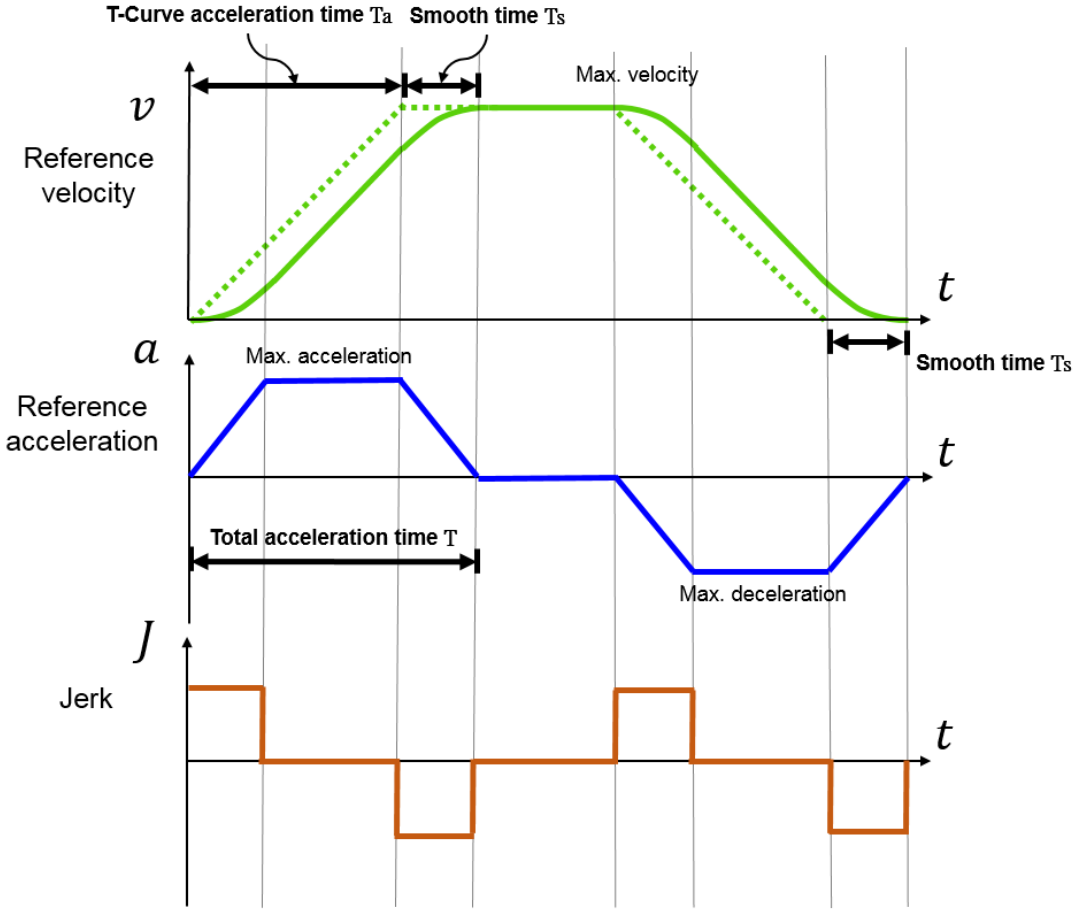



Figure 3.1.3

Besides, HIMC axis motion commands support the change of dynamic target position and velocity planning (On the Fly Modification). During axis motion, users can change axis' target position, maximum velocity, and maximum acceleration / deceleration. HIMC's profile generator will move to the new target position based on the new commands and the motion parameters.

Axis motion status can be divided into “moving” and “in-positiion”, as Figure 3.1.4 shows. Continue to send axis position planning commands in section I, and end it before entering section II. Controller will judge whether the axis is in-position based on the set target radius and debounce time.

If axis feedback position remains in reference position’s target radius after debounce time, axis position will be viewed as in-position. At this time, the status of “axis is in-position” is established inside the controller. However, if axis feedback position exceeds reference position’s target radius during debounce time, the calculation of settling time will be reset. Not until the next time axis feedback position enters target radius will the in-position condition of debounce time be checked.

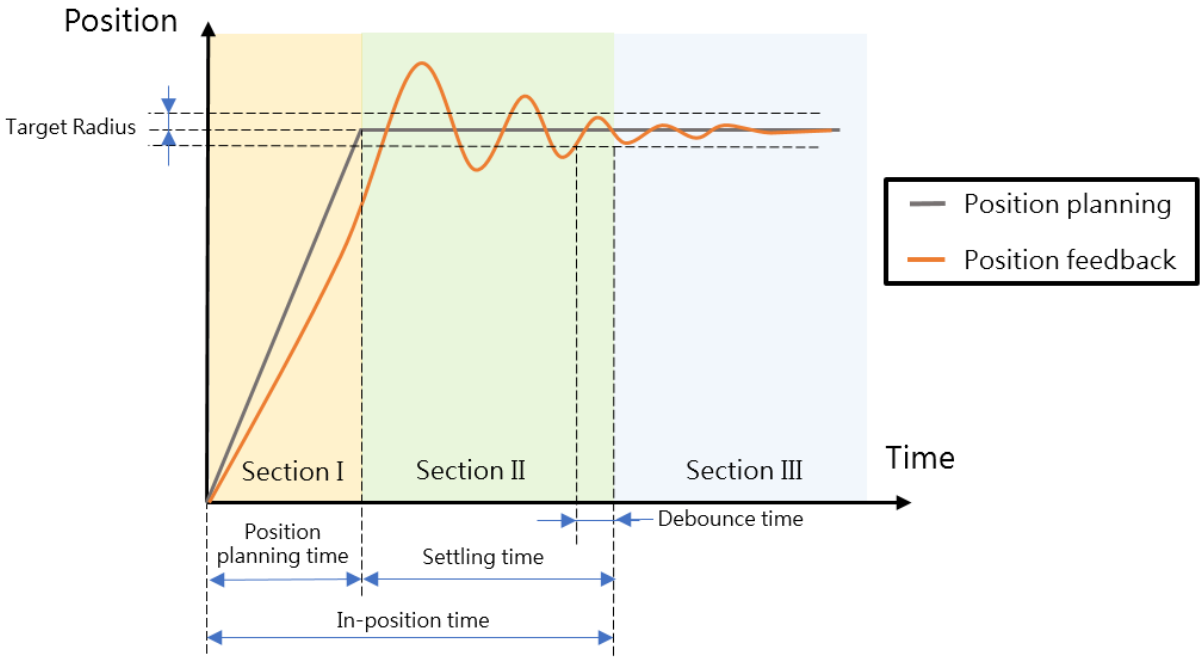



Figure 3.1.4

Refer to Figure 3.1.4, axis motion status is described as follows.

1. Section I: Axis is moving and not in-position.
2. Section II: Axis is not moving but not in-position.
3. Section III: Axis is not moving and in-position.

If the axis is at the “Synchronized” state, axis group or master axis will generate the motion profile for axis motion. Axis itself does not generate the motion profile; instead, it just follows the reference position planned by axis group or master axis.

## 3.1.1 Axis variables

Axis variables are divided into three categories, motion command variables, profile generator variables and status variables. Users can select the desired variables via Scope Manager in iA Studio (refer to section 4.8 in “iA Studio User Guide”). Detailed descriptions are shown in Table 3.1.1.1 to Table 3.1.1.5.

Table 3.1.1.1 Motion command variables for axis

| Name | Variable | Unit | Description |
|---|---|---|---|
| Sync Position Setpoint | sync_pos_set | mm or deg | Synchronized position set-point. It is the target position to be followed when the axis is in synchronized motion (such as axis group, camming, or gearing operations). |
| Position Command | pg_pos_cmd | mm or deg | Profile generator position command. It is the target position to be followed when the axis is in discrete motion (point-to-point). |
| Reference Position | ref_pos | mm or deg | Reference position. It is the position set-point generated from the profile generator according to predefined motion profile. |
| Reference Velocity | ref_vel | mm/s or deg/s | Reference velocity. It is the velocity set-point generated from the profile generator according to predefined motion profile. |
| Reference Acceleration | ref_acc | $mm/s^2$ or $deg/s^2$ | Reference acceleration. It is the acceleration set-point generated from the profile generator according to predefined motion profile. |
| Position Compensation | pos_comp_set | mm or deg | Position compensation value. It is the output of error map of dynamic error compensation function. If the function is disabled, it will be zero. |
| Compensated Position | pos_comp | mm or deg | Compensated position command value. It is the sum of reference position and position compensation value. |
| Position Offset | pos_offset | mm or deg | Position offset. The default value is zero. If users set a new axis position without moving the motor, it will be nonzero. |
| Position Output | pos_out | mm or deg | Position output. It is the axis position command without position offset. |
| Velocity Output | vel_out | mm/s or deg/s | Velocity output. It is the axis velocity command without velocity offset. |
| Acceleration Output | acc_out | $mm/s^2$ or $deg/s^2$ | Acceleration output. It is the axis acceleration command without acceleration offset. |
| Offsetted Position Output | offset_pos_out | mm or deg | Position output with offset. It is the final calculated axis position command with position offset. The value will be converted to the unit “count” and be transmitted to the corresponding slave. |
| Raw Position Feedback | pos_fb_raw | mm or deg | Raw position feedback. It is the encoder feedback read from the slave. |
| Offsetted Position Feedback | offset_pos_fb | mm or deg | Offsetted position feedback. It is the position feedback with position offset. |
| Position Feedback | pos_fb | mm or deg | Position feedback. It is in an axis coordinate system. |
| Velocity Feedback | vel_fb | mm/s or deg/s | Velocity feedback. It is in an axis coordinate system. |
| Position Error | pos_err | mm or deg | Position error. It is the difference between position output and raw position feedback. |
| Velocity Error | vel_err | mm/s or | Velocity error. It is the difference between velocity output and raw |

| Name | Variable | Unit | Description |
|---|---|---|---|
| | | deg/s | velocity feedback. |
| Move Time | movetime | ms | Move time. |
| Settling Time | settlingtime | ms | Settling time. |

Table 3.1.1.2 Profile generator variables for axis

| Name | Variable | Unit | Description |
|---|---|---|---|
| Max. Profile Velocity | max_vel | mm/s or deg/s | Maximum profile velocity. Not necessarily reached. |
| Max. Profile Acceleration | max_acc | $mm/s^2$ or $deg/s^2$ | Maximum profile acceleration. Not necessarily reached. |
| Profile Deceleration | max_dec | $mm/s^2$ or $deg/s^2$ | Maximum profile deceleration. Not necessarily reached. |
| Smooth Time | sm_factor | ms | Smooth time. Its input range is from 0 to 500. Increasing the value can reduce mechanical vibration during motion, but the total motion time will be affected. |

Table 3.1.1.3 Status variables for axis

| Name | Variable | Unit | Description |
|---|---|---|---|
| Fault Status | fault_status | N/A | Error status of axis; refer to Table 3.1.1.4 for bit definition. |
| Motion Status | motion_status | N/A | Motion status of axis; refer to Table 3.1.1.5 for bit definition. |

Table 3.1.1.4 Bit definition for axis error status

| Bit | Name | Description | Default Response |
|---|---|---|---|
| 0 | Error Stop | Axis at “error stop” state | N/A |
| 1 | Drive fault | Slave drive fault | Controller disables the axis. |
| 2 | Position error | Position error exceeds protection limit | Controller disables the axis |
| 3 | Hardware right limit | Axis hardware right limit triggered | Controller stops the motion. |
| 4 | Hardware left limit | Axis hardware left limit triggered | Controller stops the motion. |
| 5 | Software right limit | Axis software right limit triggered | Controller stops the motion. |
| 6 | Software left limit | Axis software left limit triggered | Controller stops the motion. |

Table 3.1.1.5 Bit definition for axis motion status

| Bit | Name | Description | Remark |
|---|---|---|---|
| 0 | Enabled | The axis is enabled. | N/A |
| 1 | Moving | The axis is moving. | Refer to section 3.1. |
| 2 | In Position | The axis is in-position. | Refer to section 3.1. |
| 3 | Synchronous | The axis is at the “Synchronized” state. | The axis is in an axis group or is the slave axis in synchronized motion. |
| 4 | Group | The axis is in an axis group. | Refer to section 6.1. |
| 5 | Gantry | The axis is a gantry axis. | Refer to section 5.1. |
| 6 | Input Shape | Enable Input Shape filter. | Refer to section 13.1. |
| 7 | VSF | Enable VSF filter. | Refer to section 13.1. |
| 8 | Gear | The axis is the electronic gear slave axis. | Refer to section 4.1. |
| 9 | Cam | The axis is the electronic cam slave axis. | Not supported yet. |
| 10 | Accelerating | The axis is accelerating. | Refer to section 3.1. |
| 11 | Homed | The axis completes homing. | N/A |

# 3.2 Axis motion control

## 3.2.1 HIMC_Enable

### Purpose

To enable an axis.

### Syntax

```
int HIMC_Enable(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure object 0x6040(Control word) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_Enable |
| LabVIEW | HIMC Enable.vi |
| Python | Enable |

## 3.2.2 HIMC_Disable

### Purpose

To disable an axis.

### Syntax

```
int HIMC_Disable(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

(1) The motion queue of the axis will be cleared.

(2) Users must configure object 0x6040(Control word) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_Disable |
| LabVIEW | HIMC Disable.vi |
| Python | Disable |

## 3.2.3 HIMC_Reset

### Purpose

To reset an axis when it is at the “error stop” state.

### Syntax

```
int HIMC_Reset(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

(1) Operate this function when the axis is in the error stop mode.

(2) Users must configure object 0x6040(Control word) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_Reset |
| LabVIEW | HIMC Reset.vi |
| Python | Reset |

## 3.2.4 HIMC_MoveAbs

### Purpose

To move the axis to an absolute target position.

### Syntax

```
int HIMC_MoveAbs(
    int    ctrl_id,
    int    axis_id,
    double pos
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| pos [in] | The value of an absolute target position.<br>Parameter unit: mm or deg |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure the corresponding command as PDO when using this function. For example, to configure CSP mode as 0x607A (Target position).

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_MoveAbs |
| LabVIEW | HIMC Move Abs.vi |
| Python | MoveAbs |

## 3.2.5 HIMC_MoveRel

### Purpose

To move the axis by a relative distance.

### Syntax

```
int HIMC_MoveRel(
    int    ctrl_id,
    int    axis_id,
    double rel_dist
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

rel_dist [in] The value to a relative distance.
Parameter unit: mm or deg

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure the corresponding command as PDO when using this function. For example, to configure CSP mode as 0x607A (Target position).

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_MoveRel |
| LabVIEW | HIMC Move Rel.vi |
| Python | MoveRel |

## 3.2.6 HIMC_MoveVel

### Purpose

To start a never-ending motion at a specific velocity.

### Syntax

```
int HIMC_MoveVel(
    int    ctrl_id,
    int    axis_id,
    double vel
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

vel [in] The value of a specific velocity.
It can be either positive or negative to indicate the direction of the motion.
Parameter unit: mm/s or deg/s

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure the corresponding command as PDO when using this function. For example, to configure CSP mode as 0x607A (Target position).

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_MoveVel |
| LabVIEW | HIMC Move Vel.vi |
| Python | MoveVel |

## 3.2.7 HIMC_MoveTrq

### Purpose

To start a never-ending motion at a specific torque.

### Syntax

```
int HIMC_MoveTrq(
    int    ctrl_id,
    int    axis_id,
    double trq_cmd
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

trq_cmd [in] Torque command.
Parameter unit: N-m

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

(1) This function is only applicable to “Profile Torque” mode.
(2) If the torque command is larger than the continuous torque of motor, the motor will move with the value of continuous torque.
(3) Users must configure object 0x6071 (Target torque) as PDO and set the force constant of the motor.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_MoveTrq |
| LabVIEW | HIMC Move Trq.vi |
| Python | MoveTrq |

## 3.2.8 HIMC_Stop

### Purpose

To stop the motion of an axis.

### Syntax

```
int HIMC_Stop(
    int ctrl_id,
    int axis_id
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

The motion queue of the axis will be cleared.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_Stop |
| LabVIEW | HIMC Stop.vi |
| Python | Stop |

## 3.2.9 HIMC_Halt

### Purpose

To halt the motion of an axis; its velocity will be set as 0.

### Syntax

```
int HIMC_Halt(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

In PP/PV/PT mode, users must configure object 0x6040(Control word) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_Halt |
| LabVIEW | HIMC Halt.vi |
| Python | Halt |

## 3.2.10 HIMC_Resume

### Purpose

To resume the motion of an axis from “halt” status.

### Syntax

```
int HIMC_Resume(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

In PP/PV/PT mode, users must configure object 0x6040(Control word) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_Resume |
| LabVIEW | HIMC Resume.vi |
| Python | Resume |

# 3.3 Axis setting

## 3.3.1 HIMC_GetMaxVel

### Purpose

To get the maximum profile velocity of an axis.

### Syntax

```
int HIMC_GetMaxVel(
    int    ctrl_id,
    int    axis_id,
    double *p_vel
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_vel [out] A pointer to the buffer to receive the maximum profile velocity of an axis.
Parameter unit: mm/s or deg/s

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetMaxVel |
| LabVIEW | HIMC Get Max Vel.vi |
| Python | GetMaxVel |

## 3.3.2 HIMC_SetVel

### Purpose

To set the maximum profile velocity of an axis.

### Syntax

```
int HIMC_SetVel(
    int    ctrl_id,
    int    axis_id,
    double vel
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

vel [in] The new maximum profile velocity of an axis.
Parameter unit: mm/s or deg/s
Input range: nonzero positive value

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetVel |
| LabVIEW | HIMC Set Vel.vi |
| Python | SetVel |

## 3.3.3 HIMC_GetMaxAcc

### Purpose

To get the maximum profile acceleration of an axis.

### Syntax

```
int HIMC_GetMaxAcc(
    int    ctrl_id,
    int    axis_id,
    double *p_acc
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained from HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_acc [out] A pointer to the buffer to receive the maximum profile acceleration of an axis.
Parameter unit: $mm/s^2$ or $deg/s^2$

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetMaxAcc |
| LabVIEW | HIMC Get Max Acc.vi |
| Python | GetMaxAcc |

## 3.3.4 HIMC_SetAcc

### Purpose

To set the maximum profile acceleration of an axis.

### Syntax

```
int HIMC_SetAcc(
    int    ctrl_id,
    int    axis_id,
    double acc
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

acc [in] The new maximum profile acceleration of an axis.
Parameter unit: $mm/s^2$ or $deg/s^2$
Input range: nonzero positive value

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetAcc |
| LabVIEW | HIMC Set Acc.vi |
| Python | SetAcc |

## 3.3.5 HIMC_SetAccTime

### Purpose

To set the acceleration time of an axis.

### Syntax

```
int HIMC_SetAccTime(
    int    ctrl_id,
    int    axis_id,
    double acc_time
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| acc_time [in] | The acceleration time of an axis.<br>Parameter unit: ms<br>Input range: nonzero positive value |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetAccTime |
| LabVIEW | HIMC Set Acc Time.vi |
| Python | SetAccTime |

## 3.3.6 HIMC_GetMaxDec

### Purpose

To get the maximum profile deceleration of an axis.

### Syntax

```
int HIMC_GetMaxDec(
    int    ctrl_id,
    int    axis_id,
    double *p_dec
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_dec [out] A pointer to the buffer to receive the maximum profile deceleration of an axis.
Parameter unit: $mm/s^2$ or $deg/s^2$

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetMaxDec |
| LabVIEW | HIMC Get Max Dec.vi |
| Python | GetMaxDec |

## 3.3.7 HIMC_SetDec

### Purpose

To set the maximum profile deceleration of an axis.

### Syntax

```
int HIMC_SetDec(
    int    ctrl_id,
    int    axis_id,
    double dec
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

dec [in] The new maximum profile deceleration of an axis.
Parameter unit: $mm/s^2$ or $deg/s^2$
Input range: nonzero positive value

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetDec |
| LabVIEW | HIMC Set Dec.vi |
| Python | SetDec |

## 3.3.8 HIMC_SetDecTime

### Purpose

To set the deceleration time of an axis.

### Syntax

```
int HIMC_SetDecTime(
    int    ctrl_id,
    int    axis_id,
    double dec_time
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

dec_time [in] The deceleration time of an axis.
Parameter unit: ms
Input range: nonzero positive value

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetDecTime |
| LabVIEW | HIMC Set Dec Time.vi |
| Python | SetDecTime |

## 3.3.9 HIMC_GetKillDec

### Purpose

To get the kill deceleration of an axis.

### Syntax

```
int HIMC_GetKillDec(
    int    ctrl_id,
    int    axis_id,
    double *p_kill_dec
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_kill_dec [out] A pointer to the buffer to receive the kill deceleration of an axis.
Parameter unit: $mm/s^2$ or $deg/s^2$

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetKillDec |
| LabVIEW | HIMC Get Kill Dec.vi |
| Python | GetKillDec |

## 3.3.10 HIMC_SetKillDec

### Purpose

To set the kill deceleration of an axis.

### Syntax

```
int HIMC_SetKillDec(
    int    ctrl_id,
    int    axis_id,
    double kill_dec
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| kill_dec [in] | The new kill deceleration of an axis.<br>Parameter unit: mm/s$^2$ or deg/s$^2$<br>Input range: nonzero positive value |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetKillDec |
| LabVIEW | HIMC Set Kill Dec.vi |
| Python | SetKillDec |

## 3.3.11 HIMC_GetSWRL

### Purpose

To get the software right limit position of an axis.

### Syntax

```
int HIMC_GetSWRL(
    int    ctrl_id,
    int    axis_id,
    double *p_right_limit_pos
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_right_limit_pos [out] | A pointer to the buffer to receive the software right limit position of an axis.<br>Parameter unit: mm or deg |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSWRL |
| LabVIEW | HIMC Get SWRL.vi |
| Python | GetSWRL |

## 3.3.12 HIMC_SetSWRL

### Purpose

To set the software right limit position of an axis.

### Syntax

```
int HIMC_SetSWRL(
    int    ctrl_id,
    int    axis_id,
    double right_limit_pos
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| right_limit_pos [in] | The new software right limit position of an axis.<br>Parameter unit: mm or deg |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSWRL |
| LabVIEW | HIMC Set SWRL.vi |
| Python | SetSWRL |

## 3.3.13 HIMC_GetSWLL

### Purpose

To get the software left limit position of an axis.

### Syntax

```
int HIMC_GetSWLL(
    int    ctrl_id,
    int    axis_id,
    double *p_left_limit_pos
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_left_limit_pos [out] | A pointer to the buffer to receive the software left limit position of an axis.<br>Parameter unit: mm or deg |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSWLL |
| LabVIEW | HIMC Get SWLL.vi |
| Python | GetSWLL |

## 3.3.14 HIMC_SetSWLL

### Purpose

To set the software left limit position of an axis.

### Syntax

```
int HIMC_SetSWLL(
    int    ctrl_id,
    int    axis_id,
    double left_limit_pos
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| left_limit_pos [in] | The new software left limit position of an axis.<br>Parameter unit: mm or deg |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSWLL |
| LabVIEW | HIMC Set SWLL.vi |
| Python | SetSWLL |

## 3.3.15 HIMC_GetSMTime

### Purpose

To get the profile smooth time of an axis.

### Syntax

```
int HIMC_GetSMTime(
    int    ctrl_id,
    int    axis_id,
    double *p_smooth_time
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_smooth_time [out] | A pointer to the buffer to receive the profile smooth time of an axis.<br>Parameter unit: ms |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSMTime |
| LabVIEW | HIMC Get SM Time.vi |
| Python | GetSMTime |

## 3.3.16 HIMC_SetSMTime

### Purpose

To set the profile smooth time of an axis.

### Syntax

```
int HIMC_SetSMTime(
    int    ctrl_id,
    int    axis_id,
    double smooth_time
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| smooth_time [in] | The new profile smooth time of an axis.<br>Parameter unit: ms<br>Input range: 0 ~ 500 |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

This function is not applicable when the axis is moving.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSMTime |
| LabVIEW | HIMC Set SM Time.vi |
| Python | SetSMTime |

## 3.3.17 HIMC_GetMoveTime

### Purpose

To get the move time of an axis.

### Syntax

```
int HIMC_GetMoveTime(
    int    ctrl_id,
    int    axis_id,
    double *p_move_time
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_move_time [out] | A pointer to the buffer to receive the move time of an axis.<br>Parameter unit: ms |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetMoveTime |
| LabVIEW | HIMC Get Move Time.vi |
| Python | GetMoveTime |

## 3.3.18 HIMC_GetSettlingTime

### Purpose

To get the settling time of an axis.

### Syntax

```
int HIMC_GetSettlingTime(
    int    ctrl_id,
    int    axis_id,
    double *p_settling_time
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_settling_time [out] | A pointer to the buffer to receive the settling time of an axis.<br>Parameter unit: ms |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSettlingTime |
| LabVIEW | HIMC Get Settling Time.vi |
| Python | GetSettlingTime |

## 3.3.19 HIMC_SetPos

### Purpose

To set the position of an axis and change home offset.

### Syntax

```
int HIMC_SetPos(
    int    ctrl_id,
    int    axis_id,
    double pos
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| pos [in] | The value of the axis' current position.<br>Parameter unit: mm or deg |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

This function is not applicable when the axis is at the "Synchronized" state, added to an axis group, or at the error state.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetPos |
| LabVIEW | HIMC Set Pos.vi |
| Python | SetPos |

## 3.3.20 HIMC_GetPosFb

### Purpose

To get the feedback position of an axis.

### Syntax

```
int HIMC_GetPosFb(
    int    ctrl_id,
    int    axis_id,
    double *p_pos
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_pos [out] | A pointer to the buffer to receive the feedback position of an axis.<br>Parameter unit: mm or deg |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure object 0x6064(Position actual value) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetPosFb |
| LabVIEW | HIMC Get Pos Fb.vi |
| Python | GetPosFb |

## 3.3.21 HIMC_GetPosOffset

### Purpose

To get the position offset of an axis.

### Syntax

```
int HIMC_GetPosOffset(
    int    ctrl_id,
    int    axis_id,
    double *p_pos
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_pos [out] | A pointer to the buffer to receive the position offset of an axis.<br>Parameter unit: mm or deg |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetPosOffset |
| LabVIEW | HIMC Get Pos Offset.vi |
| Python | GetPosOffset |

## 3.3.22 HIMC_GetPosErr

### Purpose

To get the position error of an axis.

### Syntax

```
int HIMC_GetPosErr(
    int    ctrl_id,
    int    axis_id,
    double *p_pos_err
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_pos_err [out] | A pointer to the buffer to receive the position error of an axis.<br>Parameter unit: mm or deg |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetPosErr |
| LabVIEW | HIMC Get Pos Err.vi |
| Python | GetPosErr |

## 3.3.23 HIMC_GetVelFb

### Purpose

To get the velocity feedback of an axis.

### Syntax

```
int HIMC_GetVelFb(
    int    ctrl_id,
    int    axis_id,
    double *p_vel
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_vel [out] A pointer to the buffer to receive the velocity feedback of an axis.
Parameter unit: mm/s or deg/s

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetVelFb |
| LabVIEW | HIMC Get Vel Fb.vi |
| Python | GetVelFb |

## 3.3.24 HIMC_GetVelErr

### Purpose

To get the velocity error of an axis.

### Syntax

```
int HIMC_GetVelErr(
    int    ctrl_id,
    int    axis_id,
    double *p_vel_err
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_vel_err [out] A pointer to the buffer to receive the velocity error of an axis.
Parameter unit: mm/s or deg/s

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetVelErr |
| LabVIEW | HIMC Get Vel Err.vi |
| Python | GetVelErr |

## 3.3.25 HIMC_GetCurrFb

### Purpose

To get the current feedback of an axis.

### Syntax

```
int HIMC_GetCurrFb(
    int    ctrl_id,
    int    axis_id,
    int *p_curr
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_curr [out] A pointer to the buffer to receive the current feedback of an axis.
Parameter unit: 0.1% rated current.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure object 0x6077(Torque actual value) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetCurrFb |
| LabVIEW | HIMC Get Curr Fb.vi |
| Python | GetCurrFb |

## 3.3.26 HIMC_GetRefPos

### Purpose

To get the reference position of an axis.

### Syntax

```
int HIMC_GetRefPos(
    int    ctrl_id,
    int    axis_id,
    double *p_pos
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_pos [out] A pointer to the buffer to receive the reference position of an axis.
Parameter unit: mm or deg

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetRefPos |
| LabVIEW | HIMC Get Ref Pos.vi |
| Python | GetRefPos |

## 3.3.27 HIMC_GetRefVel

### Purpose

To get the reference velocity of an axis.

### Syntax

```
int HIMC_GetRefVel(
    int    ctrl_id,
    int    axis_id,
    double *p_vel
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_vel [out] A pointer to the buffer to receive the reference velocity of an axis.
Parameter unit: mm/s or deg/s

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetRefVel |
| LabVIEW | HIMC Get Ref Vel.vi |
| Python | GetRefVel |

## 3.3.28 HIMC_GetRefAcc

### Purpose

To get the reference acceleration of an axis.

### Syntax

```
int HIMC_GetRefAcc(
    int    ctrl_id,
    int    axis_id,
    double *p_acc
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_acc [out] A pointer to the buffer to receive the reference acceleration of an axis.
Parameter unit: $mm/s^2$ or $deg/s^2$

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetRefAcc |
| LabVIEW | HIMC Get Ref Acc.vi |
| Python | GetRefAcc |

## 3.3.29 HIMC_GetPosOut

### Purpose

To get the position command output of an axis sent by the controller to the slave drive.

### Syntax

```
int HIMC_GetPosOut(
    int    ctrl_id,
    int    axis_id,
    double *p_pos
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_pos [out] A pointer to the buffer to receive the position command output of an axis.
Parameter unit: mm or deg

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetPosOut |
| LabVIEW | HIMC Get Pos Out.vi |
| Python | GetPosOut |

## 3.3.30 HIMC_GetVelOut

### Purpose

To get the velocity command output of an axis sent by the controller to the slave drive.

### Syntax

```
int HIMC_GetVelOut(
    int    ctrl_id,
    int    axis_id,
    double *p_vel
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_vel [out] A pointer to the buffer to receive the velocity command output of an axis.
Parameter unit: mm/s or deg/s

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetVelOut |
| LabVIEW | HIMC Get Vel Out.vi |
| Python | GetVelOut |

## 3.3.31 HIMC_GetAccOut

### Purpose

To get the acceleration command output of an axis sent by the controller to the slave drive.

### Syntax

```
int HIMC_GetAccOut(
    int    ctrl_id,
    int    axis_id,
    double *p_acc
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_acc [out] A pointer to the buffer to receive the acceleration command output of an axis.
Parameter unit: $mm/s^2$ or $deg/s^2$

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetAccOut |
| LabVIEW | HIMC Get Acc Out.vi |
| Python | GetAccOut |

## 3.3.32 HIMC_IgnoreHWL

### Purpose

To ignore the warning messages of hardware limit protection.

### Syntax

```
int HIMC_IgnoreHWL(
    int ctrl_id,
    int axis_id,
    int cmd
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

cmd [in] Set it as “1” to ignore the messages.
Set it as “0” to restore the messages (default).

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IgnoreHWL |
| LabVIEW | HIMC Ignore HWL.vi |
| Python | IgnoreHWL |

## 3.3.33 HIMC_IgnoreSWL

### Purpose

To ignore the warning messages of software limit protection.

### Syntax

```
int HIMC_IgnoreSWL(
    int ctrl_id,
    int axis_id,
    int cmd
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

cmd [in] Set it as “1” to ignore the messages.
Set it as “0” to restore the messages (default).

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IgnoreSWL |
| LabVIEW | HIMC Ignore SWL.vi |
| Python | IgnoreSWL |

## 3.3.34 HIMC_IgnorePE

### Purpose

To ignore the warning messages of position error limit.

### Syntax

```
int HIMC_IgnorePE(
    int ctrl_id,
    int axis_id,
    int cmd
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

cmd [in] Set it as “1” to ignore the messages.
Set it as “0” to restore the messages (default).

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IgnorePE |
| LabVIEW | HIMC Ignore PE.vi |
| Python | IgnorePE |

## 3.3.35 HIMC_GetAxisNum

### Purpose

To get the number of the axes connected to the controller.

### Syntax

```
int HIMC_GetAxisNum(
    int ctrl_id,
    int *num
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| num [out] | A pointer to the buffer to receive the number of the axes connected to the controller. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetAxisNum |
| LabVIEW | HIMC Get Axis Num.vi |
| Python | GetAxisNum |

## 3.3.36 HIMC_SetVelScale

### Purpose

To set the velocity scale of axis motion.

### Syntax

```
int HIMC_SetVelScale(
    int    ctrl_id,
    int    axis_id,
    double vel_scale
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

vel_scale [in] The new velocity scale of axis motion.
Input range: 0 ~ 100

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetVelScale |
| LabVIEW | HIMC Set Vel Scale.vi |
| Python | SetVelScale |

## 3.3.37 HIMC_GetVelScale

### Purpose

To get the velocity scale of axis motion.

### Syntax

```
int HIMC_GetVelScale(
    int    ctrl_id,
    int    axis_id,
    double *p_vel_scale
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_vel_scale [out] | A pointer to the buffer to receive the velocity scale of axis motion.<br>Its range is from 0 to 100. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetVelScale |
| LabVIEW | HIMC Get Vel Scale.vi |
| Python | GetVelScale |

## 3.3.38 HIMC_SetRollover

### Purpose

To set the position rollover value of an axis.

### Syntax

```
int HIMC_SetRollover(
    int    ctrl_id,
    int    axis_id,
    double rollover_val
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| rollover_val [in] | The position rollover value of an axis.<br>Parameter unit: mm or deg<br>Input range: nonzero positive value |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

(1) If parameter “rollover_val” is set as 0, the function is closed.

(2) This function is applicable only when the axis is disabled.

(3) This function is not applicable when the axis is added to an axis group.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetRollover |
| LabVIEW | HIMC Set Rollover.vi |
| Python | SetRollover |

## 3.3.39 HIMC_GetRolloverTurns

### Purpose

To get the number of turns when an axis is on rollover mode.

### Syntax

```
int HIMC_GetRolloverTurns(
    int ctrl_id,
    int axis_id,
    int *p_turns
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_turns [out] A pointer to the buffer to receive the number of turns when an axis is on rollover mode.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetRolloverTurns |
| LabVIEW | HIMC Get Rollover Turns.vi |
| Python | GetRolloverTurns |

## 3.3.40 HIMC_SetOpMode

### Purpose

To set the operational mode of an axis.

### Syntax

```
int HIMC_SetOpMode(
    int ctrl_id,
    int axis_id,
    int op_mode
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| Op_mode [in] | New operational mode of an axis.<br>Input range: 1 (PP), 3 (PV), 4 (PT), 8(CSP), 9(CSV), 10(CST) |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure object 0x6060(Mode of operation) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| eader | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetOpMode |
| LabVIEW | HIMC Set Op Mode.vi |
| Python | SetOpMode |

## 3.3.41 HIMC_SetBufferMode

**Purpose**

To set the buffer mode of an axis.

**Syntax**

```
int HIMC_SetBufferMode(
    int ctrl_id,
    int axis_id,
    int buf_mode
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| buf_mode [in] | New buffer mode of an axis.<br>Input range: 0 (immediate stop mode), 1 (buffer mode), 2 (continuous motion mode) |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetBufferMode |
| LabVIEW | HIMC Set Buffer Mode.vi |
| Python | SetBufferMode |

## 3.3.42 HIMC_GetMultiAxesFeedbackPos

### Purpose

To get the feedback position of the list of axes.

### Syntax

```
int HIMC_GetMultiAxesFeedbackPos(
    int ctrl_id,
    int *p_axes_id_array,
    int num_of_axes,
    int *p_pos_array
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| p_axes_id_array [in] | A pointer to the buffer to store the list of axes ID. |
| num_of_axes [in] | Number of axes. |
| p_pos_array [out] | A pointer to the buffer to receive the feedback position of the list of axes.<br>Parameter unit: mm or deg |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetMultiAxesFeedbackPos |
| LabVIEW | HIMC Get Multi Axes Feedback Pos.vi |
| Python | GetMultiAxesFeedbackPos |

# 3.4 Axis status

## 3.4.1 HIMC_IsEnabled

### Purpose

To query the “enable” status of an axis.

### Syntax

```
int HIMC_IsEnabled(
    int ctrl_id,
    int axis_id,
    int *p_enabled
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_enabled [out] A pointer to the buffer to receive the enable status of an axis.
If the axis is at the “Enabled” state, the value will be 1. Otherwise, it will be 0.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure object 0x6041(Stratus word) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsEnabled |
| LabVIEW | HIMC Is Enabled.vi |
| Python | IsEnabled |

## 3.4.2 HIMC_IsMoving

### Purpose

To query the “moving” status of an axis. If the axis is moving, PG (profile generator) continues outputting new positions.

### Syntax

```
int HIMC_IsMoving(
    int ctrl_id,
    int axis_id,
    int *p_is_moving
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_moving [out] | A pointer to the buffer to receive the moving status of an axis.<br>If the axis is at the “Moving” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsMoving |
| LabVIEW | HIMC Is Moving.vi |
| Python | IsMoving |

## 3.4.3 HIMC_IsInPos

### Purpose

To query the “in-position” status of an axis. If the axis is in-position, the position error is kept within an error window (target radius) for a specific duration (debounce time).

### Syntax

```
int HIMC_IsInPos(
    int ctrl_id,
    int axis_id,
    int *p_in_position
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_in_position [out] | A pointer to the buffer to receive the in-position status of an axis.<br>If the axis is at the “InPos” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsInPos |
| LabVIEW | HIMC Is In Pos.vi |
| Python | IsInPos |

## 3.4.4 HIMC_IsErrorStop

### Purpose

To query whether the axis is at the “error stop” state.

### Syntax

```
int HIMC_IsErrorStop(
    int ctrl_id,
    int axis_id,
    int *p_is_errorstop
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_errorstop [out] | A pointer to the buffer to receive the error stop status of an axis.<br>If the axis is at the “ErrorStop” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsErrorStop |
| LabVIEW | HIMC Is Error Stop.vi |
| Python | IsErrorStop |

## 3.4.5 HIMC_IsGantry

### Purpose

To query whether the axis is at the “gantry” state.

### Syntax

```
int HIMC_IsGantry(
    int ctrl_id,
    int axis_id,
    int *p_is_gantry
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_gantry [out] | A pointer to the buffer to receive the gantry status of an axis.<br>If the axis is at the “Gantry” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsGantry |
| LabVIEW | HIMC Is Gantry.vi |
| Python | IsGantry |

## 3.4.6 HIMC_IsGrouped

### Purpose

To query whether the axis is grouped into an axis group.

### Syntax

```
int HIMC_IsGrouped(
    int ctrl_id,
    int axis_id,
    int *p_is_grouped
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_grouped [out] | A pointer to the buffer to receive the grouped status of an axis.<br>If the axis is at the “Grouped” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsGrouped |
| LabVIEW | HIMC Is Grouped.vi |
| Python | IsGrouped |

## 3.4.7 HIMC_IsSync

### Purpose

To query whether the axis is at the “Synchronized” state.

### Syntax

```
int HIMC_IsSync(
    int ctrl_id,
    int axis_id,
    int *p_is_sync
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_is_sync [out] A pointer to the buffer to receive the sync status of an axis.
If the axis is at the “Sync” state, the value will be 1. Otherwise, it will be 0.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsSync |
| LabVIEW | HIMC Is Sync.vi |
| Python | IsSync |

## 3.4.8 HIMC_IsHWLL

### Purpose

To query whether the axis reaches the hardware left limit.

### Syntax

```
int HIMC_IsHWLL(
    int ctrl_id,
    int axis_id,
    int *p_is_hwll
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_is_hwll [out] A pointer to the buffer to receive the HWLL status of an axis.
If the axis is at the “HWLL” state, the value will be 1. Otherwise, it will be 0.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

When using this function, users must configure object 0x60FD(Digital inputs) as PDO and specify bit 0 as left limit input.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsHWLL |
| LabVIEW | HIMC Is HWLL.vi |
| Python | IsHWLL |

## 3.4.9 HIMC_IsHWRL

### Purpose

To query whether the axis reaches the hardware right limit.

### Syntax

```
int HIMC_IsHWRL(
    int ctrl_id,
    int axis_id,
    int *p_is_hwrl
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_is_hwrl [out] A pointer to the buffer to receive the HWRL status of an axis.
If the axis is at the “HWRL” state, the value will be 1. Otherwise, it will be 0.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

When using this function, users must configure object 0x60FD(Digital inputs) as PDO and specify bit 1 as right limit input.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsHWRL |
| LabVIEW | HIMC Is HWRL.vi |
| Python | IsHWRL |

## 3.4.10 HIMC_IsSWLL

### Purpose

To query whether the axis reaches the software left limit.

### Syntax

```
int HIMC_IsSWLL(
    int ctrl_id,
    int axis_id,
    int *p_is_swll
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_is_swll [out] A pointer to the buffer to receive the SWLL status of an axis.
If the axis is at the “SWLL” state, the value will be 1. Otherwise, it will be 0.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsSWLL |
| LabVIEW | HIMC Is SWLL.vi |
| Python | IsSWLL |

## 3.4.11 HIMC_IsSWRL

### Purpose

To query whether the axis reaches the software right limit.

### Syntax

```
int HIMC_IsSWRL(
    int ctrl_id,
    int axis_id,
    int *p_is_swrl
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_is_swrl [out] A pointer to the buffer to receive the SWRL status of an axis.
If the axis is at the “SWRL” state, the value will be 1. Otherwise, it will be 0.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsSWRL |
| LabVIEW | HIMC Is SWRL.vi |
| Python | IsSWRL |

## 3.4.12 HIMC_IsDriveErr

### Purpose

To query whether the axis triggers drive alarms.

### Syntax

```
int HIMC_IsDriveErr(
    int ctrl_id,
    int axis_id,
    int *p_is_driveerr
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_driveerr [out] | A pointer to the buffer to receive the DriveErr status of an axis.<br>If the axis is at the “DriveErr” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure object 0x6041(Status word) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsDriveErr |
| LabVIEW | HIMC Is Drive Err.vi |
| Python | IsDriveErr |

## 3.4.13 HIMC_IsPosErr

### Purpose

To query whether the position error of an axis exceeds the protection limit.

### Syntax

```
int HIMC_IsPosErr(
    int ctrl_id,
    int axis_id,
    int *p_is_poserr
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_poserr [out] | A pointer to the buffer to receive the PosErr status of an axis.<br>If the axis is at the “PosErr” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

The error protection limit indicates the position error tolerance of an axis in controller.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsPosErr |
| LabVIEW | HIMC Is Pos Err.vi |
| Python | IsPosErr |

## 3.4.14 HIMC_IsCompActive

### Purpose

To query whether the compensation function is active.

### Syntax

```
int HIMC_IsCompActive(
    int ctrl_id,
    int axis_id,
    int *p_is_compactive
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_compactive [out] | A pointer to the buffer to receive the compensation active status of an axis.<br>If the axis is at the “CompActive” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsCompActive |
| LabVIEW | HIMC Is Comp Active.vi |
| Python | IsCompActive |

## 3.4.15 HIMC_IsAcc

### Purpose

To query whether the axis is accelerating.

### Syntax

```
int HIMC_IsAcc(
    int ctrl_id,
    int axis_id,
    int *p_is_acc
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_is_acc [out] A pointer to the buffer to receive the accelerating status of an axis.
If the axis is at the “Acc” state, the value will be 1. Otherwise, it will be 0.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsAcc |
| LabVIEW | HIMC Is Acc.vi |
| Python | IsAcc |

(This page is intentionally left blank.)

# 4. Synchronized Motion functions


4. Synchronized Motion functions ..... 4-1
4.1 Overview ..... 4-2
4.1.1 Synchronized motion variables ..... 4-3
4.2 HIMC_EnableGear ..... 4-4
4.3 HIMC_DisableGear ..... 4-5
4.4 HIMC_GearIn ..... 4-6
4.5 HIMC_GearOut ..... 4-7
4.6 HIMC_GetGearRatio ..... 4-8
4.7 HIMC_IsInGear ..... 4-9
4.8 HIMC_IsGearMaster ..... 4-10
4.9 HIMC_IsGearSlave ..... 4-11

# 4.1 Overview

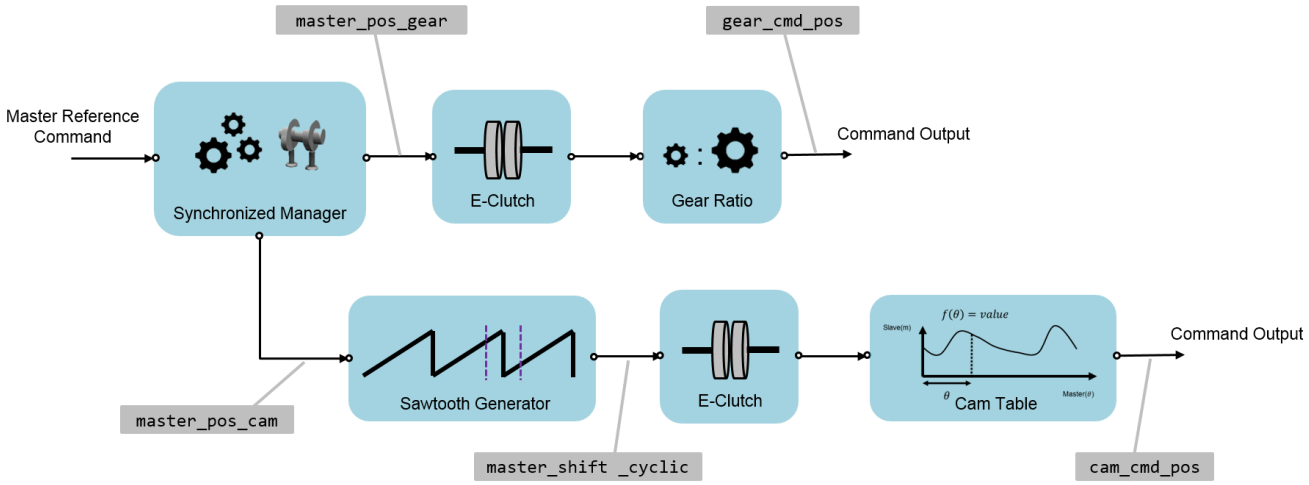



Figure 4.1.1

Users can define synchronized motion between one axis and another. Master axis, a leading axis, will generate position command first, and then slave axis will refer to master axis based on the motion configuration. If master-slave relationship is constant, the motion is electronic gearing. On the other hand, if slave axis needs to follow a pattern, the motion is electronic camming. In Figure 4.1.2, axis 0 acts as master axis, leading axis 1, 2, 3 and 4. Axis 1, 2 and 3 adopt electronic gearing, while axis 4 adopts electronic camming.

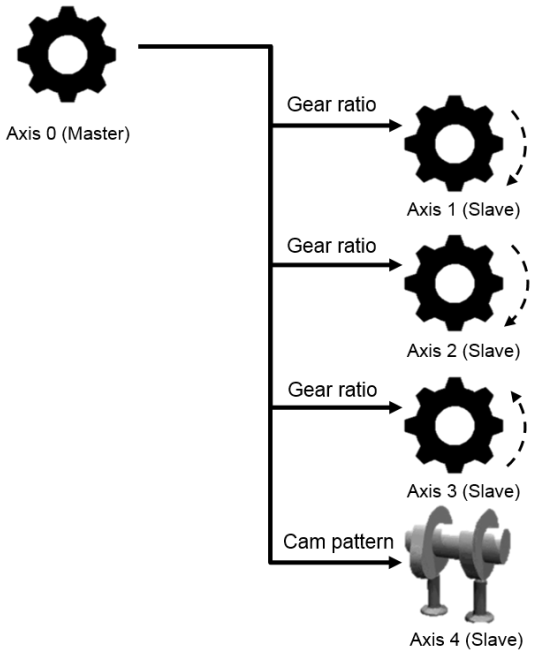



Figure 4.1.2

## 4.1.1 Synchronized motion variables

Common synchronized motion variables are given in Table 4.1.1.1. Users can select the desired variables via Scope Manager in iA Studio (refer to section 4.8 in “iA Studio User Guide”).

Table 4.1.1.1

| Name | Variable | Unit | Description |
|---|---|---|---|
| Raw Master Position | master_pos_gear | mm or deg | Position command of master axis. |
| Gear Command Position | gear_cmd_pos | mm or deg | Slave axis outputs position command. |
| Gear Ratio | gear_ratio | mm or deg | Gear ratio. |

## 4.2 HIMC_EnableGear

### Purpose

To couple two axes in a master-slave relationship.

### Syntax

```
int HIMC_EnableGear(
    int ctrl_id,
    int axis_master_id,
    int axis_slv_id
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_master_id [in] | Master axis index. |
| axis_slv_id [in] | Slave axis index. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

This function is applicable only when both axes are enabled.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_EnableGear |
| LabVIEW | HIMC Enable Gear.vi |
| Python | EnableGear |

# 4.3 HIMC_DisableGear

**Purpose**

To uncouple two axes from the master-slave relationship to two independent axes.

**Syntax**

```
int HIMC_DisableGear(
    int ctrl_id,
    int axis_slv_id
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_slv_id [in] | Slave axis index. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_DisableGear |
| LabVIEW | HIMC Disable Gear.vi |
| Python | DisableGear |

## 4.4 HIMC_GearIn

### Purpose

To change slave axis' state from disengaged to engaged.

### Syntax

```
int HIMC_GearIn(
    int ctrl_id,
    int axis_master_id,
    int axis_slv_id,
    double gear_ratio
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_master_id [in] | Master axis index. |
| axis_slv_id [in] | Slave axis index. |
| gear_ratio [in] | Value of gear ratio. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

This function is applicable only when both axes are enabled.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GearIn |
| LabVIEW | HIMC Gear In.vi |
| Python | GearIn |

# 4.5 HIMC_GearOut

## Purpose

To change slave axis' state from engaged to disengaged.

## Syntax

```
int HIMC_GearOut(
    int ctrl_id,
    int axis_slv_id
);
```

## Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_slv_id [in] Slave axis index.

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GearOut |
| LabVIEW | HIMC Gear Out.vi |
| Python | GearOut |

# 4.6 HIMC_GetGearRatio

## Purpose

To get the gear ratio of slave axis.

## Syntax

```
int HIMC_GetGearRatio(
    int ctrl_id,
    int axis_slv_id,
    double *p_ratio
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_slv_id [in] | Slave axis index. |
| p_ratio [out] | A pointer to the buffer to receive the gear ratio of slave axis. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGearRatio |
| LabVIEW | HIMC Get Gear Ratio.vi |
| Python | GetGearRatio |

# 4.7 HIMC_IsInGear

**Purpose**

To query whether the slave axis is at the “engaged” state.

**Syntax**

```
int HIMC_IsInGear(
    int ctrl_id,
    int axis_id,
    int *p_is_in_gear
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_in_gear [out] | A pointer to the buffer to receive the engaged status of slave axis.<br>If the slave axis is at the “InGear” state, the value will be 1. Otherwise, it will be 0. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_IsInGear |
| LabVIEW | HIMC Is In Gear.vi |
| Python | IsInGear |

# 4.8 HIMC_IsGearMaster

**Purpose**

To query whether the axis is master axis.

**Syntax**

```
int HIMC_IsGearMaster(
    int ctrl_id,
    int axis_id,
    int *p_is_gear_master
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_gear_master [out] | A pointer to the buffer to receive the status of whether the axis is master axis.<br>If the axis is at the “GearMaster” state, the value will be 1. Otherwise, it will be 0. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_IsGearMaster |
| LabVIEW | HIMC Is Gear Master.vi |
| Python | IsGearMaster |

# 4.9 HIMC_IsGearSlave

**Purpose**

To query whether the axis is slave axis.

**Syntax**

```
int HIMC_IsGearSlave(
    int ctrl_id,
    int axis_id,
    int *p_is_gear_slv
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_gear_slv [out] | A pointer to the buffer to receive the status of whether the axis is slave axis.<br>If the axis is at the “GearSlave” state, the value will be 1. Otherwise, it will be 0. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_IsGearSlave |
| LabVIEW | HIMC Is Gear Slave.vi |
| Python | IsGearSlave |

(This page is intentionally left blank.)

# 5. Gantry functions

5. Gantry functions ..... 5-1
5.1 Overview ..... 5-2
5.2 HIMC_EnableGantryPair ..... 5-3
5.3 HIMC_DisableGantryPair ..... 5-4
5.4 HIMC_GetGantryPairID ..... 5-5
5.5 HIMC_IsGantryPair ..... 5-6

# 5.1 Overview

The gantry configuration transforms a pair of right-hand-side (RHS) axis and left-hand-side (LHS) axis into a pair of imaginary linear axis and yaw axis, as Figure 5.1.1 shows. After establishing gantry configuration, users can give RHS axis a linear-axis-direction command to drive both RHS and LHS axes in the same direction, and give LHS axis a rotary motion command of yaw-axis-direction.

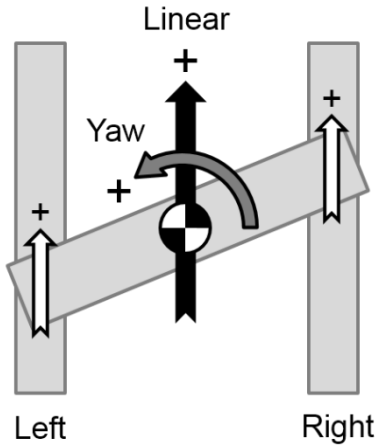



Figure 5.1.1

In gantry configuration, linear axis' and yaw axis' position feedback are defined as follows.

$$Pos_{linear} = \frac{Pos_{RHS} + Pos_{LHS}}{2}; \qquad Pos_{yaw} = \frac{Pos_{RHS} - Pos_{LHS}}{2}$$

| | | | |
|---|---|---|---|
| $Pos_{linear}$ | : Linear axis' position feedback | $Pos_{yaw}$ | : Yaw axis' position feedback |
| $Pos_{RHS}$ | : RHS axis' position feedback | $Pos_{LHS}$ | : LHS axis' position feedback |

Figure 5.1.2 is a position feedback schematic of linear axis, yaw axis, RHS axis and LHS axis.

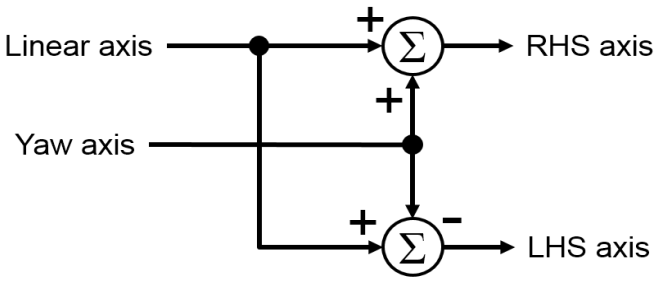



Figure 5.1.2

## 5.2 HIMC_EnableGantryPair

### Purpose

To set up a gantry pair.

### Syntax

```
int HIMC_EnableGantryPair(
    int ctrl_id,
    int lhs_axis_id,
    int rhs_axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

lhs_axis_id [in] Left-hand-side axis index.

rhs_axis_id [in] Right-hand-side axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

This function is applicable only when both axes are disabled.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_EnableGantryPair |
| LabVIEW | HIMC Enable Gantry Pair.vi |
| Python | EnableGantryPair |

# 5.3 HIMC_DisableGantryPair

## Purpose

To split a gantry pair.

## Syntax

```
int HIMC_DisableGantryPair(
    int ctrl_id,
    int axis_id
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Either axis index in a gantry pair. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Remark

This function is applicable only when both axes are disabled.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_DisableGantryPair |
| LabVIEW | HIMC Disable Gantry Pair.vi |
| Python | DisableGantryPair |

# 5.4 HIMC_GetGantryPairID

## Purpose

To get the gantry pair ID of any gantry axis.

## Syntax

```
int HIMC_GetGantryPairID(
    int ctrl_id,
    int axis_id,
    int *p_id
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Either axis index in a gantry pair. |
| p_id [out] | A pointer to the buffer to receive the gantry pair ID.<br>If the input axis is not gantry axis, it will be -1. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGantryPairID |
| LabVIEW | HIMC Get Gantry Pair ID.vi |
| Python | GetGantryPairID |

## 5.5 HIMC_IsGantryPair

### Purpose

To query whether the two axes are a gantry pair.

### Syntax

```
int HIMC_IsGantryPair(
    int ctrl_id,
    int axis_id_1,
    int axis_id_2,
    int *p_is_gantry_pair
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id_1 [in] | Axis index 1. |
| axis_id_2 [in] | Axis index 2. |
| p_is_gantry_pair [out] | A pointer to the buffer to receive the status of whether the two axes are a gantry pair. If the two axes are at the “GantryPair” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsGantryPair |
| LabVIEW | HIMC Is Gantry Pair.vi |
| Python | IsGantryPair |

# 6. Group functions


6. Group functions ..... 6-1
6.1 Overview ..... 6-3
6.1.1 Group variables ..... 6-6
6.1.2 Coordinate systems ..... 6-9
6.1.3 Kinematics ..... 6-13
6.1.4 Buffer modes ..... 6-13
6.1.5 Transition modes ..... 6-15
6.2 Group motion control ..... 6-17
6.2.1 HIMC_EnableGroup ..... 6-17
6.2.2 HIMC_DisableGroup ..... 6-18
6.2.3 HIMC_ResetGroup ..... 6-19
6.2.4 HIMC_StopGroup ..... 6-20
6.2.5 HIMC_HaltGroup ..... 6-21
6.2.6 HIMC_ResumeGroup ..... 6-22
6.2.7 HIMC_JogGroup ..... 6-23
6.2.8 HIMC_JogGroupAxis ..... 6-24
6.2.9 HIMC_LineAbs2D ..... 6-25
6.2.10 HIMC_LineAbs3D ..... 6-26
6.2.11 HIMC_LineRel2D ..... 6-27
6.2.12 HIMC_LineRel3D ..... 6-28
6.2.13 HIMC_Arc2D ..... 6-29
6.2.14 HIMC_ArcCW2D ..... 6-31
6.2.15 HIMC_ArcCCW2D ..... 6-32
6.2.16 HIMC_ArcAngle2D ..... 6-33
6.2.17 HIMC_Circle2D ..... 6-35
6.3 Group setting ..... 6-37
6.3.1 HIMC_AddAxesToGrp ..... 6-37
6.3.2 HIMC_RemoveAxisFromGrp ..... 6-38
6.3.3 HIMC_SetupGroup ..... 6-39
6.3.4 HIMC_UngrpAllAxes ..... 6-40
6.3.5 HIMC_GetGroupID ..... 6-41
6.3.6 HIMC_SetGrpMotionProfile ..... 6-42
6.3.7 HIMC_SetGrpAngMotionProfile ..... 6-44
6.3.8 HIMC_GetGrpKin ..... 6-46
6.3.9 HIMC_SetGrpKin ..... 6-47
6.3.10 HIMC_GetGrpMaxVel ..... 6-48
6.3.11 HIMC_SetGrpVel ..... 6-49
6.3.12 HIMC_GetGrpMaxAcc ..... 6-50

6.3.13 HIMC_SetGrpAcc ... 6-51
6.3.14 HIMC_SetGrpAccTime ... 6-52
6.3.15 HIMC_GetGrpMaxDec ... 6-53
6.3.16 HIMC_SetGrpDec ... 6-54
6.3.17 HIMC_SetGrpDecTime ... 6-55
6.3.18 HIMC_GetGrpSMTime ... 6-56
6.3.19 HIMC_SetGrpSMTime ... 6-57
6.3.20 HIMC_GetGrpCoordSys ... 6-58
6.3.21 HIMC_SetGrpCoordSys ... 6-59
6.3.22 HIMC_GetGrpBufferMode ... 6-60
6.3.23 HIMC_SetGrpBufferMode ... 6-61
6.3.24 HIMC_GetGrpTransMode ... 6-62
6.3.25 HIMC_SetGrpTransMode ... 6-63
6.3.26 HIMC_SetGrpTransPrm ... 6-64
6.3.27 HIMC_GetGrpCmdNum ... 6-66
6.3.28 HIMC_SetGrpVelScale ... 6-67
6.3.29 HIMC_GetGrpVelScale ... 6-68
6.3.30 HIMC_GetGrpCoordTrans ... 6-69
6.3.31 HIMC_SetGrpCoordTrans ... 6-70
6.3.32 HIMC_GetGrpPoseCmd ... 6-71
6.3.33 HIMC_GetGrpPoseFb ... 6-72
6.4 Group status ... 6-73
6.4.1 HIMC_IsGrpEnabled ... 6-73
6.4.2 HIMC_IsGrpMoving ... 6-74
6.4.3 HIMC_IsGrpInPos ... 6-75
6.4.4 HIMC_IsGrpErrorStop ... 6-76
6.5 Advanced group motion control ... 6-77
6.5.1 HIMC_LineAbs ... 6-77
6.5.2 HIMC_LineRel ... 6-79
6.5.3 HIMC_CircleAbs ... 6-81
6.5.4 HIMC_CircleRel ... 6-83

# 6.1 Overview

HIMC provides axis group motion commands of multi-axis linear and circular simultaneous interpolation function, including LineAbs2D / 3D, LineRel2D / 3D, Arc2D, Circle2D, etc. Compared with axis motion commands, axis group motion commands ensure the synchronization of each axis in the group. The start time and the stop time of each axis motion are the same, and the controller will adjust the motion velocity of each axis based on the reference velocity given by users. The basic function of HIMC controller supports up to 4 axes for axis group motion commands (product model: MC-XX-XX-XX-00). If there is a need for 5-axis (or above axes') simultaneous machining for axis group motion function, please contact HIWIN Mikrosystem or local distributors for further information.

Figure 6.1.1 is the parameter flow diagram of HIMC axis group motion command. As the position feedback of each axis goes through the calculation of forward kinematics, axis group's position feedback in machine coordinate system (Cartesian Position Feedback) will be obtained. Based on the target command given by users, controller will plan the interpolation command in space (Cartesian Position Command) due to axis group's motion profile (as Figure 6.1.2 shows), and calculate the corresponding position command of each axis with inverse kinematics.

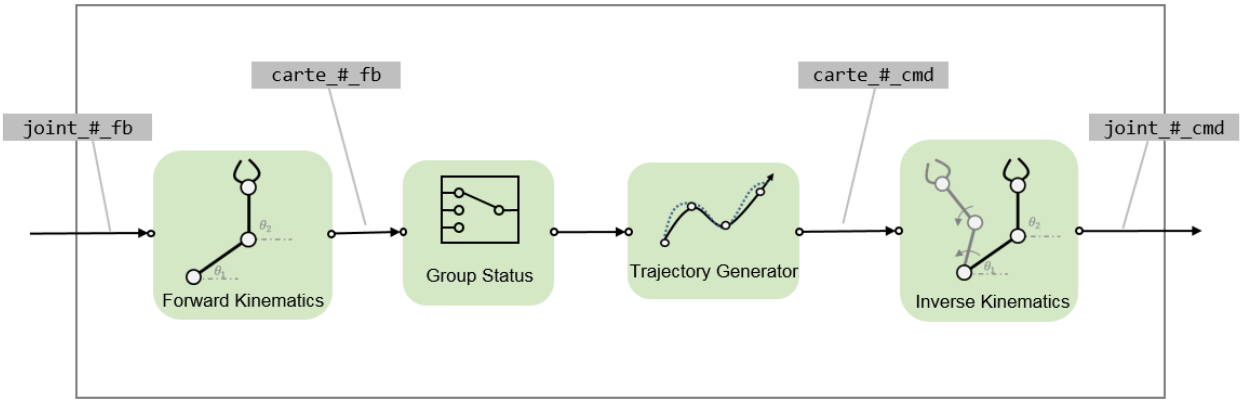



Figure 6.1.1

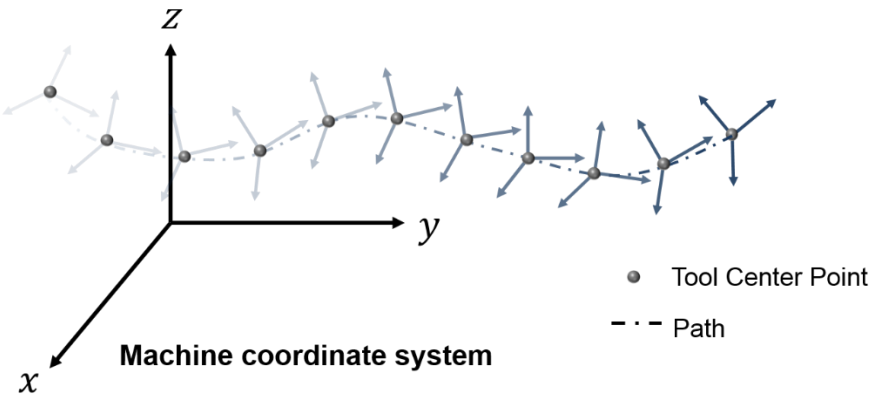



Figure 6.1.2

In axis group motion commands, HIMC will calculate the moving distance of each segment in space. Different from axis motion commands, the velocity planning is planned along axis group's moving direction in space, and the moving direction will change based on the direction of moving command.

Axis group motion commands are similar to axis motion commands (Refer to chapter 3); it also adopts S-Curve velocity planning, as Figure 6.1.3 shows. Axis group motion in space consists of two parts, translation and rotation. Translation command consists of the position commands of XYZ, while rotation command consists of the rotation commands of ABC. With axis group, users can set velocity planning parameters of translation and rotation, including profile generator's maximum velocity, maximum acceleration, maximum deceleration and smooth time.

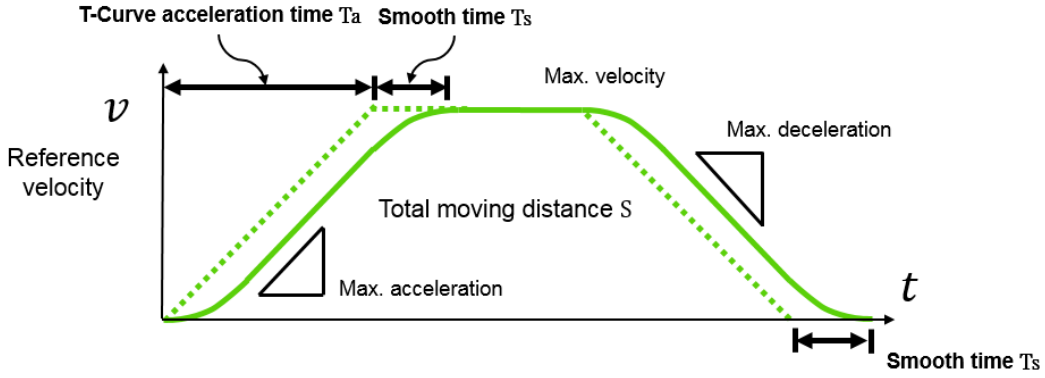



Figure 6.1.3

Each axis group motion command will be viewed as one segment, as Figure 6.1.4 shows. During the motion, based on the translation command and rotation command of each segment as well as the velocity planning parameters set by users, HIMC will calculate the move time of translation command and rotation command. The velocity planning parameters of the longer move time will be viewed as the feed rate of axis group. As for the shorter move time, it will move according to the motion command apportioned by the feed rate command.

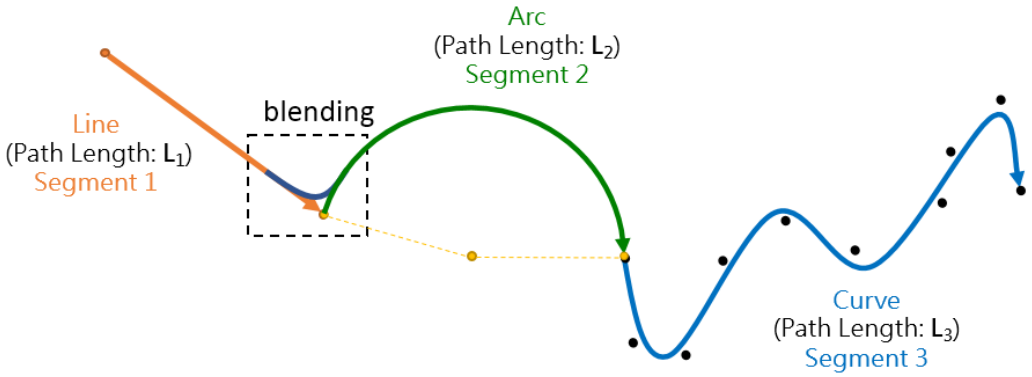



Figure 6.1.4

In HIMC, there is a built-in buffer for axis group commands. The segment of each motion command will be placed in this buffer; up to 512 motion commands can be accepted at the same time. Motion commands larger than this capacity limit will be discarded by the controller, and an error message will be displayed. Between the motion commands of segments, velocity and path will be planned based on the buffer mode and transition mode set by users. The planned velocity and path may change due to the selected mode. Take Figure 6.1.4 as an example, if buffer modes are used to set the velocity handover of each segment, the total length of this axis motion will be "$S = L_1(Line) + L_2(Arc) + L_3(Curve)$". Refer to section 6.1.4 and 6.1.5 for details.

Axis group motion status is similar to axis motion status; it can also be divided into "moving" and "in-positiion". During the motion, there are three phases like Figure 3.1.4 shown, including:

1. Axis group is moving and not in-position.
2. Axis group is not moving but not in-position.
3. Axis group is not moving and in-position.

Unlike axis motion command using target radius and debounce time to check whether the axis is in-position, axis group motion command checks whether all axes in the group are in-position. That is, if the axis group is in-position, all axes in the group are at the "in-position" state.

## 6.1.1 Group variables

Axis group variables are divided into three categories, motion command variables, profile generator variables and status variables. Users can select the desired variables via Scope Manager in iA Studio (refer to section 4.8 in “iA Studio User Guide”). Detailed descriptions are shown in Table 6.1.1.1 to Table 6.1.1.5.

Table 6.1.1.1 Motion command variables for axis group

| Name | Variable | Unit | Description |
|---|---|---|---|
| Cartesian Position Command | carte_pose_cmd | mm or deg | Space position command for axis group in machine coordinated system (MCS). It is an array containing the value of [X Y Z A B C]. |
| Cartesian Velocity Command | carte_vel_cmd | mm/s or deg/s | Space velocity command for axis group in machine coordinated system (MCS). It is an array containing the value of [X Y Z A B C]. |
| Cartesian Position Feedback | carte_pose_fb | mm or deg | Space position feedback for axis group in machine coordinated system (MCS). It is an array containing the value of [X Y Z A B C]. |
| Axis Position Command | joint_pos_cmd | mm or deg | Axis position command for axis group in axis coordinate system (ACS). It is an array. |
| Axis Velocity Command | joint_vel_cmd | mm/s or deg/s | Axis velocity command for axis group in axis coordinate system (ACS). It is an array. |
| Axis Acceleration Command | joint_acc_cmd | $mm/s^2$ or $deg/s^2$ | Axis acceleration command for axis group in axis coordinate system (ACS). It is an array. |
| Axis Position Feedback | joint_pos_fb | mm or deg | Axis position feedback for axis group in axis coordinate system (ACS). It is an array. |
| Cartesian Position Error | carte_pose_err | mm or deg | Space position error for axis group in machine coordinated system (MCS). It is an array containing the value of [X Y Z A B C]. |
| Reference Group Position | grp_pg_pos | mm or deg | Reference position for axis group. It is the position set-point generated from the profile generator according to axis group command’s motion profile. |
| Reference Group Velocity | grp_pg_vel | mm/s or deg/s | Reference velocity for axis group. It is the velocity set-point generated from the profile generator according to axis group command’s motion profile. |
| Reference Group Acceleration | grp_pg_acc | $mm/s^2$ or $deg/s^2$ | Reference acceleration for axis group. It is the acceleration set-point generated from the profile generator according to axis group command’s motion profile. |

Table 6.1.1.2 Profile generator variables for axis group

| Name | Variable | Unit | Description |
|---|---|---|---|
| Group Max. Linear Profile Velocity | grp_lin_vel | mm/s | Maximum linear profile velocity for axis group. Not necessarily reached. |
| Group Max. Linear Profile Acceleration | grp_lin_acc | $mm/s^2$ | Maximum linear profile acceleration for axis group. Not necessarily reached. |
| Group Max. Linear Profile Deceleration | grp_lin_dec | $mm/s^2$ | Maximum linear profile deceleration for axis group. Not necessarily reached. |
| Group Linear Smooth Time | grp_lin_sf | ms | Linear profile smooth time for axis group. Its input range is from 0 to 500. Increasing the value can reduce mechanical vibration during motion, but the total motion time will be affected. |
| Group Max. Angular Profile Velocity | grp_ang_vel | deg/s | Maximum angular profile velocity for axis group. Not necessarily reached. |
| Group Max. Angular Profile Acceleration | grp_ang_acc | $deg/s^2$ | Maximum angular profile acceleration for axis group. Not necessarily reached. |
| Group Max. Angular Profile Deceleration | grp_ang_dec | $deg/s^2$ | Maximum angular profile deceleration for axis group. Not necessarily reached. |
| Group Angular Smooth Time | grp_ang_sf | ms | Angular profile smooth time for axis group. Its input range is from 0 to 500. Increasing the value can reduce mechanical vibration during motion, but the total motion time will be affected. |

Table 6.1.1.3 Status variables for axis group

| Name | Variable | Unit | Description |
|---|---|---|---|
| Group Fault Status | grp_fault_status | N/A | Error status of axis group; refer to Table 6.1.1.4 for bit definition. |
| Group Motion Status | grp_motion_status | N/A | Motion status of axis group; refer to Table 6.1.1.5 for bit definition. |

Table 6.1.1.4 Bit definition for axis group error status

| Bit | Name | Description | Default Response |
|---|---|---|---|
| 0 | Error Stop | Axis group at “error stop” state | N/A |
| 1 | Axis Fault | Slave drive fault | Controller disables the axis; axis group is out of sync. |
| 2 | Software Limit | Axis software limit triggered | Controller stops the motion; axis group is out of sync. |

Table 6.1.1.5 Bit definition for axis group motion status

| Bit | Name | Description | Remark |
|---|---|---|---|
| 0 | Enabled | The axis group is enabled. | N/A |
| 1 | Moving | The axis group is moving. | N/A |
| 2 | In Position | The axis group is in-position. | All the axes in the axis group are in-position. |
| 3 | Input Shape | Enable axis group's Input Shape filter. | Refer to section 13.1. |

## 6.1.2 Coordinate systems

Table 6.1.2.1 shows the definition and description of HIMC's coordinate systems, including Axis Coordinate System (ACS), Machine Coordinate System (MCS), Product Coordinate System (PCS), Workpiece Coordinate System (WCS), Global Coordinate System and Coordinate Offset.

Table 6.1.2.1

| HMPL definition | Description |
|---|---|
| CS_ACS | Axis Coordinate System.<br>It is related to the motion of individual motors. |
| CS_MCS | Machine Coordinate System.<br>(Sometimes called "World Coordinate System" or "Base Coordinate System")<br>It is a coordinate system with a fixed origin on the machine, and it is linked to ACS via kinematics transformation (refer to section 6.1.3). It has 6 dimensions in total to indicate the position and orientation in space (3 translational, 3 rotational). |
| CS_PCS | Product Coordinate System (or "Program Coordinate System" in CNC program).<br>It is attached to the product or the workpiece, and it can set coordinate transformation parameters. |
| CS_WCS#<br>(#=1~15) | Workpiece Coordinate System.<br>It is used to set workpiece zero point, and it provides up to 15 independent workpiece coordinate systems. The default is no offset. It depends on Product Coordinate System, so it can set coordinate transformation parameters. |
| CS_GLOBAL | Global Coordinate System.<br>It is used to set global zero point, and it can establish the global spatial relationship of each axis group.<br>Not supported yet. |
| CS_OFFSET | Coordinate Offset.<br>It is used to set temporary zero point. The default is no offset, that is, the coordinate origin of offset is the coordinate origin of machine. It depends on Product Coordinate System, so it can set coordinate transformation parameters. |

Figure 6.1.2.1 shows an example of the relationship among ACS, MCS and PCS for a SCARA robot with two rotary axes. ACS and MCS are transformed through forward and inverse kinematics (refer to section 6.1.3). On the other hand, there is a coordinate transformation relationship between MCS and PCS. The position on the coordinate system is obtained through the translation and the rotation of the coordinate.

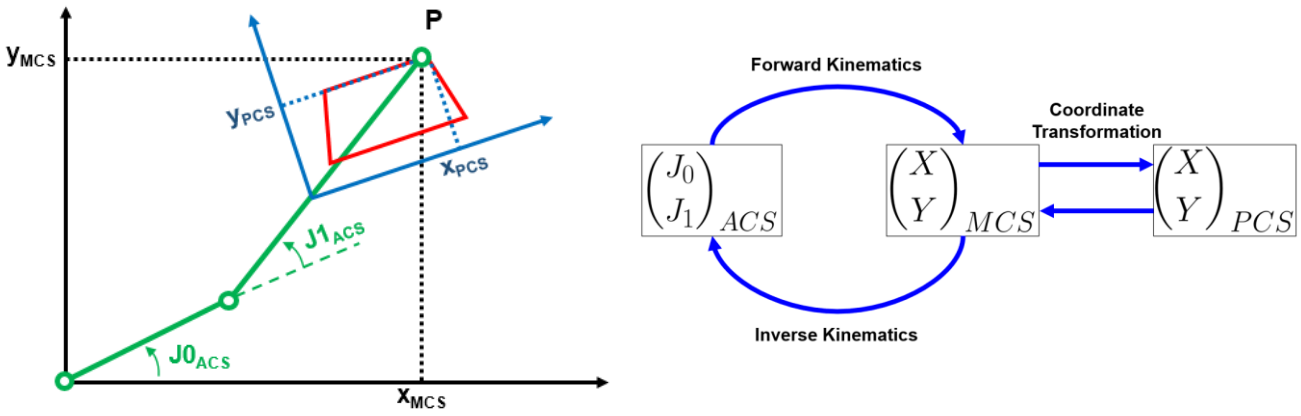



Figure 6.1.2.1

When transforming MCS into PCS, HIMC can set machine's workpiece coordinate (WCS1~15) and coordinate offset (OFFSET) based on requirement. For the setting of coordinate system, 3 translational degrees of freedom (X, Y, Z) and 3 rotational degrees of freedom (A, B, C) are used to indicate the pose in space.

HIMC adopts the "Roll-Pitch-Yaw" rotation convention in fixed angle. As Figure 6.1.2.2 shows, the degree of freedom to rotate along the X axis is **Roll**, angle **A**; the degree of freedom to rotate along the Y axis is **Pitch**, angle **B**; the degree of freedom to rotate along the Z axis is **Yaw**, angle **C**. This rotation convention is the same as using the sequence of ZYX in Tait-Bryan angles to indicate the orientation of object in space, as Figure 6.1.2.3 shows.

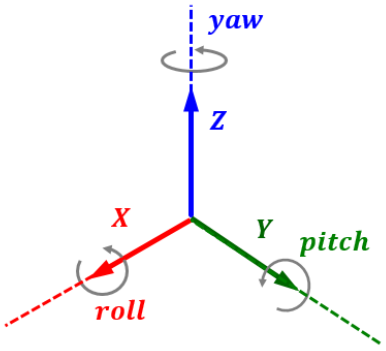



Figure 6.1.2.2

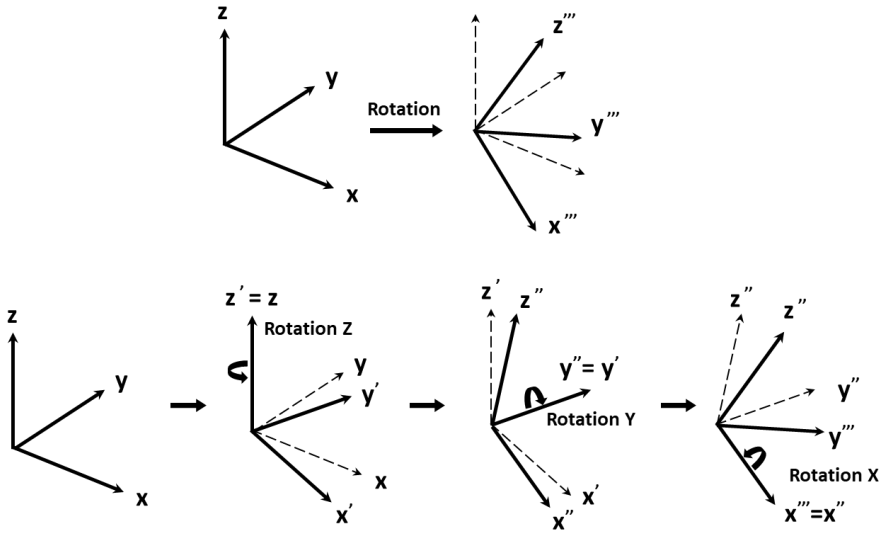



Figure 6.1.2.3

If there is no coordinate offset, the relationship between each WCS and MCS is shown in Figure 6.1.2.4.

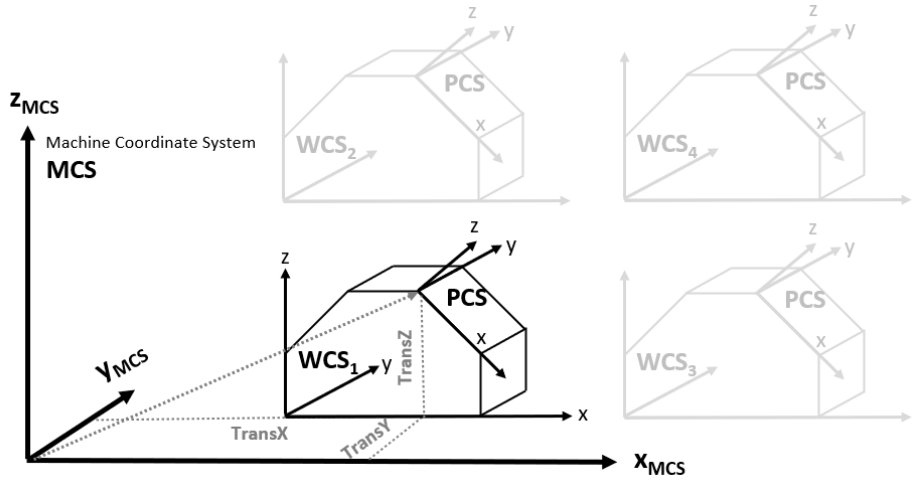



Figure 6.1.2.4

If coordinate offset is added, the relationship between WCS and MCS is shown in Figure 6.1.2.5. The transformation of coordinate offset will be added.

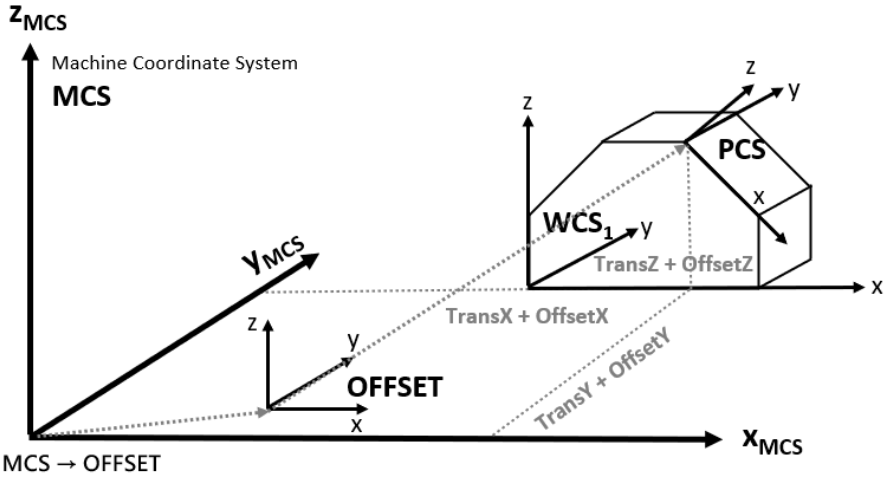



Figure 6.1.2.5

Based on the functions mentioned above, users can define the parameters for coordinate transformation in HIMC and establish the transformation relationship among the coordinate systems. Figure 6.1.2.6 shows the relationship among the coordinate systems. To help users understand easily, the figure only shows the coordinate of XY plane. In actual application, users can set 6 degrees of freedom (X, Y, Z, A, B, C) for coordinate transformation.

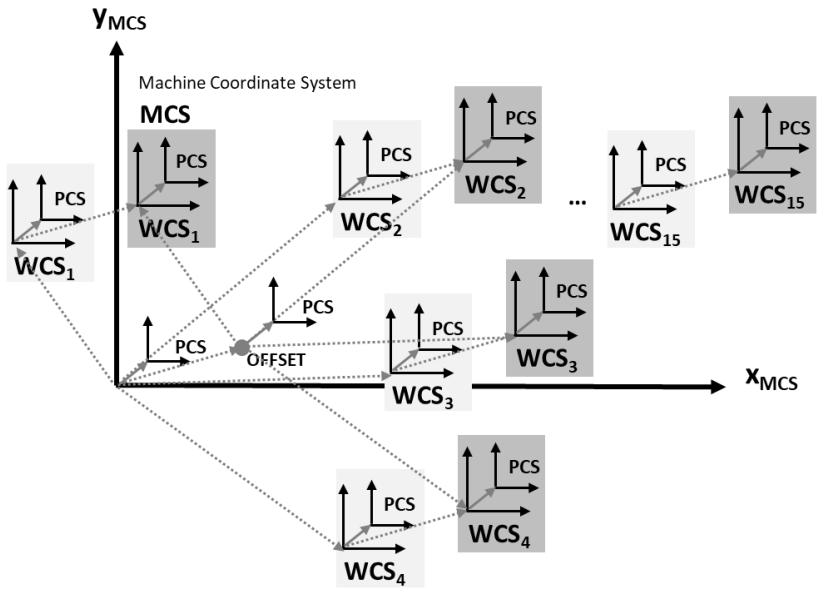



Figure 6.1.2.6

## 6.1.3 Kinematics

Kinematics mainly deals with the transformation between ACS (Axis Coordinate System) and MCS (Machine Coordinate System). Forward kinematics is the calculation from each axis' position feedback in ACS to the coordinate position of MCS. On the contrary, inverse kinematics is the calculation from coordinate position of MCS to each axis' position in ACS. Table 6.1.3.1 shows the definition of kinematics configuration provided by HIMC.

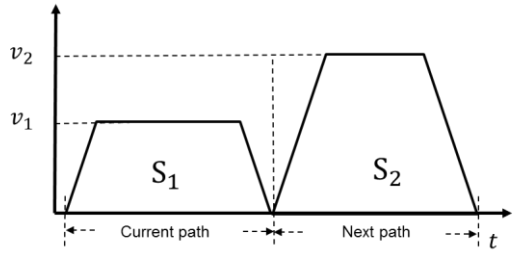
Table 6.1.3.1

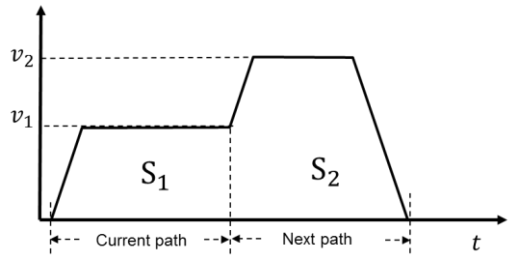
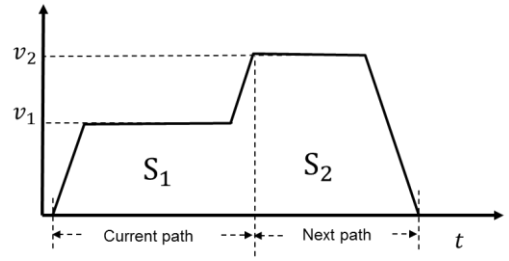
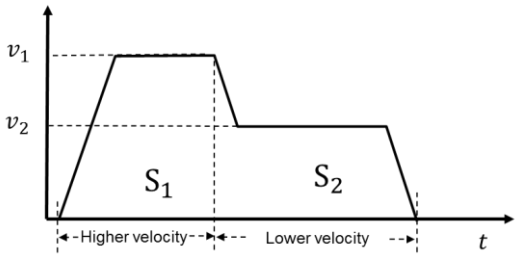
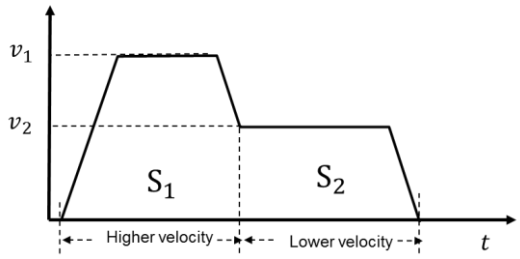
| ID | Name | Description |
|---|---|---|
| 1 | Cartesian | Map each axis in the coordinated motion group to X, Y, Z, A, B, C axis of Cartesian coordinate respectively. The maximum allowable number of axes in joint space is 6. (Default for axis group) |
| 2 | SCARA | (Not supported) |
| 3 | WAFER | (Not supported) |
| 4 | 6-Axis Articulated Robot | (Not supported) |

## 6.1.4 Buffer modes

Buffer mode determines the velocity profile at the end-points of adjacent paths. Users can use this setting to plan the profile velocity of two adjacent paths. Table 6.1.4.1 shows the definition of buffer modes provided by HIMC.

Table 6.1.4.1

| HMPL definition | Description |
|---|---|
| BM_ABORT | Abort the ongoing motion and immediately start the next one. |
| BM_BUFF | Start next path after current path is done.<br>  |

| | |
|---|---|
| BM_PREV | The velocity is blended with current path's velocity. (Blending)<br>  |
| BM_NEXT | The velocity is blended with next path's velocity. (Blending)<br>  |
| BM_HIGH | The velocity is blended with the higher velocity between the two paths. (Blending)<br>  |
| BM_LOW | The velocity is blended with the lower velocity between the two paths. (Blending)<br>  |

## 6.1.5 Transition modes

Transition mode determines the type of the transition curve between adjacent paths. With this setting, HIMC will do the calculation of corner smoothing between two linear motion commands based on the set parameter (refer to Table 6.1.5.1). To achieve a better planning path, this will affect the original motion profile.

Table 6.1.5.1

| HMPL definition | Description | Corresponding parameter | Unit |
|---|---|---|---|
| TM_NONE | None: Insert no transition curve (default mode) | N/A | N/A |
| TM_START_VEL (Not supported) | Start velocity | TPStartVelocity | mm/s or deg/s |
| TM_CONST_VEL | Constant velocity | TPVelocity | mm/s or deg/s |
| TM_CORNER_DIST | Corner distance | TPCornerDistance | mm or deg |
| TM_MAX_CORNER_DEV | Max. corner deviation | TPCornerDeviation | mm or deg |
| TM_MAX_CORNER_CURV | Max. corner curvature | TPCornerCurv | K |

When using advanced group motion control's command "HIMC_LineAbs" and "HIMC_LineRel" with transition mode TM_CONST_VEL to assign the velocity of circular motion, the distance from circular start position to corner is determined by transition mode's distance parameter set in section 6.3.26, as Figure 6.1.5.1 shows. With transition mode TM_CORNER_DIST to assign the distance from circular start position to corner, the velocity of circular motion is determined by transition mode's velocity parameter set in section 6.3.26.

If the smoothing circular radius calculated by transition mode exceeds the length of any segment, the transition mode function between the segments will be ignored.

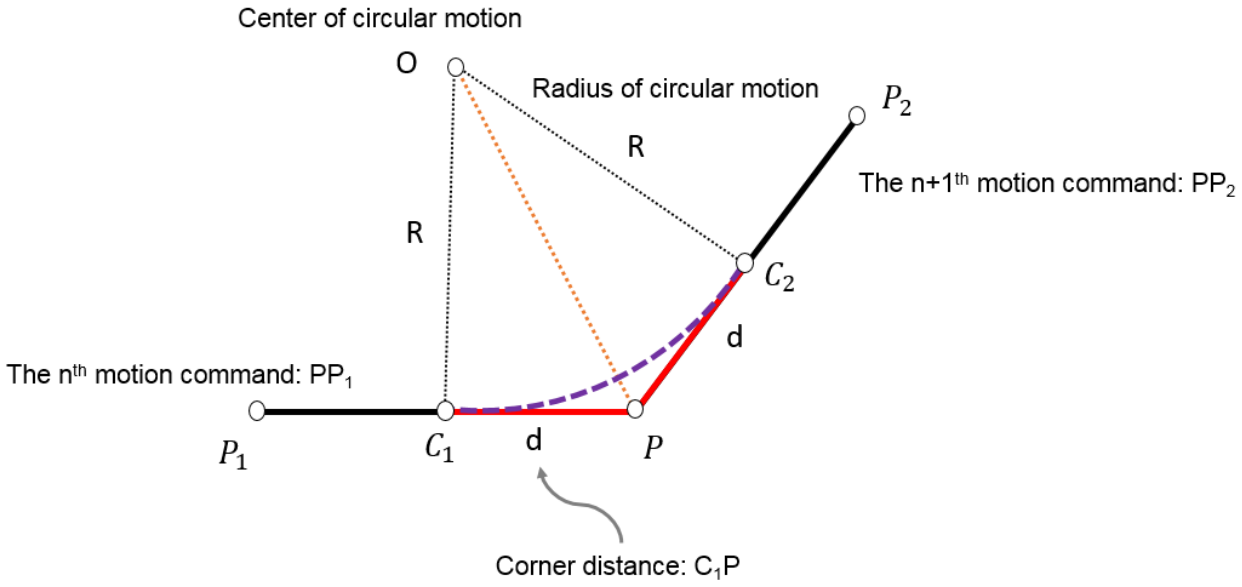

Center of circular motion
O
Radius of circular motion
R
R
$P_2$
The n+1th motion command: PP2
$C_2$
d
The nth motion command: PP1
$P_1$
$C_1$
d
$P$
Corner distance: C1P

Figure 6.1.5.1

# 6.2 Group motion control

## 6.2.1 HIMC_EnableGroup

### Purpose

To enable an axis group.

### Syntax

```
int HIMC_EnableGroup(
    int ctrl_id,
    int group_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

All the axes in the group should be enabled before executing this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_EnableGroup |
| LabVIEW | HIMC Enable Group.vi |
| Python | EnableGroup |

## 6.2.2 HIMC_DisableGroup

### Purpose

To disable an axis group.

### Syntax

```
int HIMC_DisableGroup(
    int ctrl_id,
    int group_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_DisableGroup |
| LabVIEW | HIMC Disable Group.vi |
| Python | DisableGroup |

## 6.2.3 HIMC_ResetGroup

### Purpose

To change an axis group's state from "Group Error Stop" to "Group Standby".

### Syntax

```
int HIMC_ResetGroup(
    int ctrl_id,
    int group_id
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

This function can only be used after all errors have been cleared.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ResetGroup |
| LabVIEW | HIMC Reset Group.vi |
| Python | ResetGroup |

## 6.2.4 HIMC_StopGroup

### Purpose

To stop the motion of an axis group.

### Syntax

```
int HIMC_StopGroup(
    int ctrl_id,
    int group_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

The motion queue of the axis group will be cleared.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_StopGroup |
| LabVIEW | HIMC Stop Group.vi |
| Python | StopGroup |

## 6.2.5 HIMC_HaltGroup

### Purpose

To halt the motion of an axis group; its velocity will be set as 0.

### Syntax

```
int HIMC_HaltGroup(
    int ctrl_id,
    int group_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

If the axis group is not in-position, it will keep moving.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_HaltGroup |
| LabVIEW | HIMC Halt Group.vi |
| Python | HaltGroup |

## 6.2.6 HIMC_ResumeGroup

### Purpose

To resume the motion of an axis group from “halt” status.

### Syntax

```
int HIMC_ResumeGroup(
    int ctrl_id,
    int group_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ResumeGroup |
| LabVIEW | HIMC Resume Group.vi |
| Python | ResumeGroup |

## 6.2.7 HIMC_JogGroup

**Purpose**

To make an axis group start a never-ending motion at a specific velocity in the designated direction of machine coordinate system.

**Syntax**

```
int HIMC_JogGroup(
    int    ctrl_id,
    int    group_id,
    int    carte_dir,
    double jog_vel
);
```

**Parameter**

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

carte_dir [in] The direction of motion in the machine coordinate system.
The number 0 ~ 5 orderly repersents 6-DOF {X, Y, Z, A, B, C} in the machine coordinate system.

jog_vel [in] The value of a specific velocity.
Its positive / negative value represents the same / reverse direction movement of the direction of motion.
Parameter unit: mm/s or deg/s

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_JogGroup |
| LabVIEW | HIMC Jog Group.vi |
| Python | JogGroup |

## 6.2.8 HIMC_JogGroupAxis

### Purpose

To make the specific axis in an axis group start a never-ending motion at a specific velocity in the axis coordinate system.

### Syntax

```
int HIMC_JogGroupAxis(
    int    ctrl_id,
    int    group_id,
    int    grp_axis,
    double jog_vel
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

grp_axis [in] Axis index in the axis group.
The number 0 ~ 8 orderly repersents the sequence that each axis is added to the axis group.

jog_vel [in] The value of a specific velocity.
Its positive / negative value represents the same / reverse direction movement of the direction of motion.
Parameter unit: mm/s or deg/s

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_JogGroupAxis |
| LabVIEW | HIMC Jog Group Axis.vi |
| Python | JogGroupAxis |

## 6.2.9 HIMC_LineAbs2D

### Purpose

To command an interpolated, two-dimensional linear movement on an axis group toward an absolute target position in the machine coordinate system.

### Syntax

```
int HIMC_LineAbs2D(
    int    ctrl_id,
    int    group_id,
    double end_x,
    double end_y
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

end_x [in] The value of the absolute target position in X coordinate.

end_y [in] The value of the absolute target position in Y coordinate.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_LineAbs2D |
| LabVIEW | HIMC Line Abs2D.vi |
| Python | LineAbs2D |

## 6.2.10 HIMC_LineAbs3D

**Purpose**

To command an interpolated, three-dimensional linear movement on an axis group toward an absolute target position in the machine coordinate system.

**Syntax**

```
int HIMC_LineAbs3D(
    int    ctrl_id,
    int    group_id,
    double end_x,
    double end_y,
    double end_z
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| end_x [in] | The value of the absolute target position in X coordinate. |
| end_y [in] | The value of the absolute target position in Y coordinate. |
| end_z [in] | The value of the absolute target position in Z coordinate. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_LineAbs3D |
| LabVIEW | HIMC Line Abs3D.vi |
| Python | LineAbs3D |

## 6.2.11 HIMC_LineRel2D

### Purpose

To command an interpolated, two-dimensional linear movement on an axis group toward a relative position in the machine coordinate system.

### Syntax

```
int HIMC_LineRel2D(
    int    ctrl_id,
    int    group_id,
    double distance_x,
    double distance_y
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

distance_x [in] The value of the relative distance in X coordinate.

distance_y [in] The value of the relative distance in Y coordinate.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_LineRel2D |
| LabVIEW | HIMC Line Rel2D.vi |
| Python | LineRel2D |

## 6.2.12 HIMC_LineRel3D

### Purpose

To command an interpolated, three-dimensional linear movement on an axis group toward a relative position in the machine coordinate system.

### Syntax

```
int HIMC_LineRel3D(
    int    ctrl_id,
    int    group_id,
    double distance_x,
    double distance_y,
    double distance_z
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| distance_x [in] | The value of the relative distance in X coordinate. |
| distance_y [in] | The value of the relative distance in Y coordinate. |
| distance_z [in] | The value of the relative distance in Z coordinate. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_LineRel3D |
| LabVIEW | HIMC Line Rel3D.vi |
| Python | LineRel3D |

## 6.2.13 HIMC_Arc2D

**Purpose**

To command an interpolated, two-dimensional circular movement on an axis group toward an absolute target position in the machine coordinate system.

**Syntax**

```
int HIMC_Arc2D(
    int    ctrl_id,
    int    group_id,
    double border_x,
    double border_y,
    double end_x,
    double end_y
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| border_x [in] | The value of the absolute border position in X coordinate. |
| border_y [in] | The value of the absolute border position in Y coordinate. |
| end_x [in] | The value of the absolute end position in X coordinate. |
| end_y [in] | The value of the absolute end position in Y coordinate. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

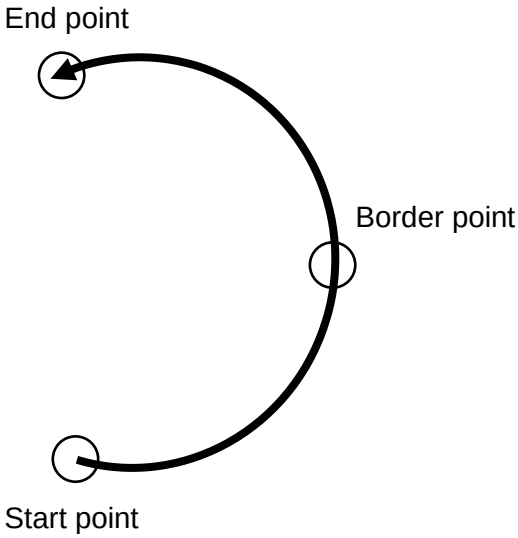
| | |
|---|---|
| C# | HIMC_Arc2D |
| LabVIEW | HIMC Arc2D.vi |
| Python | Arc2D |

**Advantage**

Users can specify the border point (the farthest point in the movement), and make sure that the machine can reach it.

**Disadvange**

It is restricted to the angle < 2π in a single command.



## 6.2.14 HIMC_ArcCW2D

### Purpose

To command an interpolated, two-dimensional circular movement on an axis group toward an absolute target position in the machine coordinate system clockwise.

### Syntax

```
int HIMC_ArcCW2D(
    int    ctrl_id,
    int    group_id,
    double center_x,
    double center_y,
    double end_x,
    double end_y
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| center_x [in] | The value of the absolute center position in X coordinate. |
| center_y [in] | The value of the absolute center position in Y coordinate. |
| end_x [in] | The value of the absolute end position in X coordinate. |
| end_y [in] | The value of the absolute end position in Y coordinate. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ArcCW2D |
| LabVIEW | HIMC ArcCW2D.vi |
| Python | ArcCW2D |

## 6.2.15 HIMC_ArcCCW2D

### Purpose

To command an interpolated, two-dimensional circular movement on an axis group toward an absolute target position in the machine coordinate system counterclockwise.

### Syntax

```
int HIMC_ArcCCW2D(
    int    ctrl_id,
    int    group_id,
    double center_x,
    double center_y,
    double end_x,
    double end_y
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| center_x [in] | The value of the absolute center position in X coordinate. |
| center_y [in] | The value of the absolute center position in Y coordinate. |
| end_x [in] | The value of the absolute end position in X coordinate. |
| end_y [in] | The value of the absolute end position in Y coordinate. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ArcCCW2D |
| LabVIEW | HIMC ArcCCW2D.vi |
| Python | ArcCCW2D |

## 6.2.16 HIMC_ArcAngle2D

**Purpose**

To command an interpolated, two-dimensional circular movement on an axis group toward an absolute target position in the machine coordinate system based on the given angle.

**Syntax**

```
int HIMC_ArcAngle2D(
    int    ctrl_id,
    int    group_id,
    double center_x,
    double center_y,
    double angle
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| center_x [in] | The value of the absolute center position in X coordinate. |
| center_y [in] | The value of the absolute center position in Y coordinate. |
| angle [in] | The angle that start point and end point relative to the absolute center position.<br>It determines the direction and the rotation angle of circular movement.<br>Parameter unit: deg |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

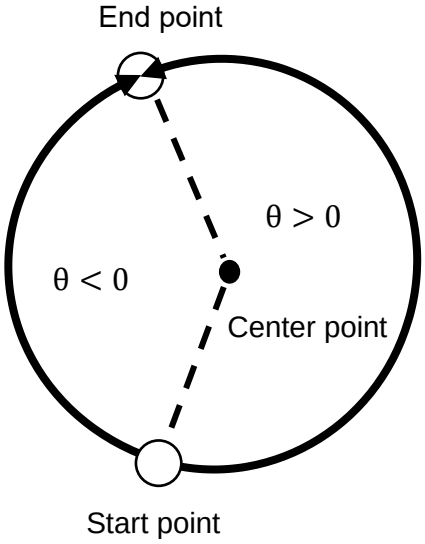
Parameter “angle” represents the direction of circular trajectory’s rotation.
If “angle” > 0, move counterclockwise; if “angle” < 0, move clockwise.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_ArcAngle2D |
| LabVIEW | HIMC Arc Angle2D.vi |
| Python | ArcAngle2D |




The value of θ determines the direction of circular movement.

$\theta > 0$: move counterclockwise

$\theta < 0$: move clockwise

## 6.2.17 HIMC_Circle2D

**Purpose**

To command an interpolated, two-dimensional circular movement on an axis group toward an absolute target position in the machine coordinate system.

**Syntax**

```
int HIMC_Circle2D(
    int    ctrl_id,
    int    group_id,
    double center_x,
    double center_y,
    double end_x,
    double end_y,
    int    turns
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| center_x [in] | The value of the absolute center position in X coordinate. |
| center_y [in] | The value of the absolute center position in Y coordinate. |
| end_x [in] | The value of the absolute end position in X coordinate. |
| end_y [in] | The value of the absolute end position in Y coordinate. |
| turns [in] | Number of turns of circular path relative to the start point.<br>It determines the direction and the total angle of circular path. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

(1) Parameter “turns” represents the direction of circular trajectory’s rotation.
If “turns” >= 0, move counterclockwise; if “turns” < 0, move clockwise.

(2) When ||turns|| <= 1, the total moving angle of circular trajectory is < 360°.
If the total moving angle of circular trajectory is >= 360° (that is, one turn, or more than one turn), ||turns|| must be >= 2.

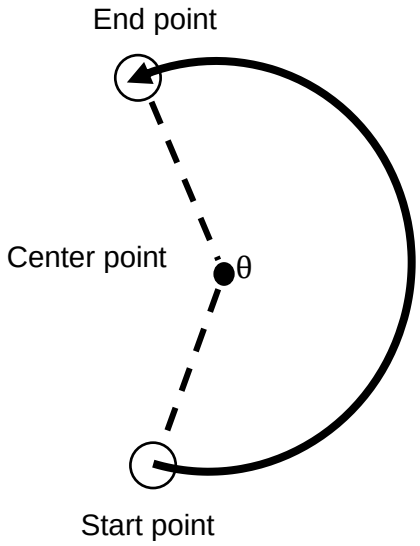
(3) The behavior of “turns = 0” and that of “turns =1” are the same when using this function.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_Circle2D |
| LabVIEW | HIMC Circle2D.vi |
| Python | Circle2D |




**Advantage**

No restriction to angles.

**Disadvantage**

Users cannot specify the border point (the farthest point in the movement). Therefore, the machine may not reach the border point.

The value of turns determines the direction of circular movement. ※ $\text{angle} = \theta + \text{turns} \times 360$

$\text{turns} \geq 0$ indicates C.C.W. direction, while $\text{turns} < 0$ indicates C.W. direction. Note that the movement of $\text{turns} = 0$ is the same as that of $\text{turns} = 1$. The following table takes $\theta = 210°$ for example.

| Turns | Calculation | Angle (Degree) |
|---|---|---|
| -2 | $210 - 2 \times 360°$ | $-510°$ |
| -1 | $210 - 1 \times 360°$ | $-150°$ |
| 0 | $210 + 0 \times 360°$ | $210°$ |
| 1 | $210 + 0 \times 360°$ | $210°$ |
| 2 | $210 + 1 \times 360°$ | $570°$ |

# 6.3 Group setting

## 6.3.1 HIMC_AddAxesToGrp

### Purpose

To add axes to an axis group with a specific sequence.

### Syntax

```
int HIMC_AddAxesToGrp(
    int ctrl_id,
    int group_id,
    int num_of_axes,
    int *p_axis_list,
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| num_of_axes [in] | Number of axes to be added to an axis group. (Maximum: 9) |
| p_axis_list [in] | A pointer to the buffer to store the sequence list of axes ID. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_AddAxesToGrp |
| LabVIEW | HIMC Add Axes To Grp.vi |
| Python | AddAxesToGrp |

## 6.3.2 HIMC_RemoveAxisFromGrp

### Purpose

To remove the last axis from an axis group.

### Syntax

```
int HIMC_RemoveAxisFromGrp(
    int ctrl_id,
    int group_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_RemoveAxisFromGrp |
| LabVIEW | HIMC Remove Axis From Grp.vi |
| Python | RemoveAxisFromGrp |

## 6.3.3 HIMC_SetupGroup

### Purpose

To set up an axis group with a specific sequence.

### Syntax

```
int HIMC_SetupGroup(
    int ctrl_id,
    int group_id,
    int num_of_axes,
    int *p_axis_list,
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| num_of_axes [in] | Number of axes. (Maximum: 9) |
| p_axis_list [in] | A pointer to the buffer to store the sequence list of axes ID. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetupGroup |
| LabVIEW | HIMC Setup Group.vi |
| Python | SetupGroup |

## 6.3.4 HIMC_UngrpAllAxes

### Purpose

To ungroup and disable an axis group.

### Syntax

```
int HIMC_UngrpAllAxes(
    int ctrl_id,
    int group_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_UngrpAllAxes |
| LabVIEW | HIMC Ungrp All Axes.vi |
| Python | UngrpAllAxes |

## 6.3.5 HIMC_GetGroupID

### Purpose

To get the axis group ID to which the axis belongs.

### Syntax

```
int HIMC_GetGroupID(
    int ctrl_id,
    int axis_id,
    int *p_group_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

p_group_id [out] A pointer to the buffer to receive the axis group ID to which the axis belongs.
If the value is -1, it indicates that the axis does not belong to any axis group.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGroupID |
| LabVIEW | HIMC Get Group ID.vi |
| Python | GetGroupID |

## 6.3.6 HIMC_SetGrpMotionProfile

### Purpose

To set TCP linear motion parameters for an axis group.

### Syntax

```
int HIMC_SetGrpMotionProfile(
    int    ctrl_id,
    int    group_id,
    double max_velocity,
    double max_acceleration,
    double max_deceleration,
    double smooth_time
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| max_velocity [in] | The maximum linear profile velocity for an axis group.<br>Parameter unit: mm/s<br>Input range: 0 ~ 5000 |
| max_acceleration [in] | The maximum linear profile acceleration for an axis group.<br>Parameter unit: $mm/s^2$<br>Input range: >0 ~ 50000 (acceleration cannot be 0) |
| max_deceleration [in] | The maximum linear profile deceleration for an axis group.<br>Parameter unit: $mm/s^2$<br>Input range: >0 ~ 50000 (deceleration cannot be 0) |
| smooth_time [in] | The linear profile smooth time for an axis group.<br>Parameter unit: ms<br>Input range: 0 ~ 500 |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

The default value of linear group motion profile is **[100, 500, 500, 50]** for velocity, acceleration, deceleration, and smooth time respectively.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetGrpMotionProfile |
| LabVIEW | HIMC Set Grp Motion Profile.vi |
| Python | SetGrpMotionProfile |

## 6.3.7 HIMC_SetGrpAngMotionProfile

### Purpose

To set TCP angular motion parameters for an axis group.

### Syntax

```
int HIMC_SetGrpAngMotionProfile(
    int    ctrl_id,
    int    group_id,
    double max_velocity,
    double max_acceleration,
    double max_deceleration,
    double smooth_time
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| max_velocity [in] | The maximum angular profile velocity for an axis group.<br>Parameter unit: deg/s<br>Input range: 0 ~ 7200 |
| max_acceleration [in] | The maximum angular profile acceleration for an axis group.<br>Parameter unit: deg/s$^2$<br>Input range: >0 ~ 72000 (acceleration cannot be 0) |
| max_deceleration [in] | The maximum angular profile deceleration for an axis group.<br>Parameter unit: deg/s$^2$<br>Input range: >0 ~ 72000 (deceleration cannot be 0) |
| smooth_time [in] | The angular profile smooth time for an axis group.<br>Parameter unit: ms<br>Input range: 0 ~ 500 |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

The default value of angular group motion profile is **[360, 1800, 1800, 50]** for velocity, acceleration, deceleration, and smooth time respectively.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetGrpAngMotionProfile |
| LabVIEW | HIMC Set Grp Ang Motion Profile.vi |
| Python | SetGrpAngMotionProfile |

## 6.3.8 HIMC_GetGrpKin

### Purpose

To get the kinematics type of an axis group.

### Syntax

```
int HIMC_GetGrpKin(
    int ctrl_id,
    int group_id,
    int *p_grp_kin
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

p_grp_kin [out] A pointer to the buffer to receive the kinematics type of an axis group.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpKin |
| LabVIEW | HIMC Get Grp Kin.vi |
| Python | GetGrpKin |

## 6.3.9 HIMC_SetGrpKin

### Purpose

To set the kinematics type of an axis group.

### Syntax

```
int HIMC_SetGrpKin(
    int ctrl_id,
    int group_id,
    int kin_type
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| kin_type [in] | The new kinematics type of an axis group. **Refer to section 6.1.3 for details.** |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpKin |
| LabVIEW | HIMC Set Grp Kin.vi |
| Python | SetGrpKin |

## 6.3.10 HIMC_GetGrpMaxVel

### Purpose

To get the maximum profile velocity of an axis group.

### Syntax

```
int HIMC_GetGrpMaxVel(
    int    ctrl_id,
    int    group_id,
    double *p_grp_vel
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

p_grp_vel [out] A pointer to the buffer to receive the maximum profile velocity of an axis group.
Parameter unit: mm/s or deg/s

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpMaxVel |
| LabVIEW | HIMC Get Grp Max Vel.vi |
| Python | GetGrpMaxVel |

## 6.3.11 HIMC_SetGrpVel

### Purpose

To set the maximum profile velocity of an axis group.

### Syntax

```
int HIMC_SetGrpVel(
    int    ctrl_id,
    int    group_id,
    double vel
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

vel [in] The new maximum profile velocity of an axis group.
Parameter unit: mm/s or deg/s
Input range: 0 ~ 5000

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpVel |
| LabVIEW | HIMC Set Grp Vel.vi |
| Python | SetGrpVel |

## 6.3.12 HIMC_GetGrpMaxAcc

### Purpose

To get the maximum profile acceleration of an axis group.

### Syntax

```
int HIMC_GetGrpMaxAcc(
    int    ctrl_id,
    int    group_id,
    double *p_grp_acc
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

p_grp_acc [out] A pointer to the buffer to receive the maximum profile acceleration of an axis group.
Parameter unit: $mm/s^2$ or $deg/s^2$

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpMaxAcc |
| LabVIEW | HIMC Get Grp Max Acc.vi |
| Python | GetGrpMaxAcc |

## 6.3.13 HIMC_SetGrpAcc

### Purpose

To set the maximum profile acceleration of an axis group.

### Syntax

```
int HIMC_SetGrpAcc(
    int    ctrl_id,
    int    group_id,
    double acc
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| acc [in] | The new maximum profile acceleration of an axis group.<br>Parameter unit: $mm/s^2$ or $deg/s^2$<br>Input range: >0 ~ 50000 (acceleration cannot be 0) |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpAcc |
| LabVIEW | HIMC Set Grp Acc.vi |
| Python | SetGrpAcc |

## 6.3.14 HIMC_SetGrpAccTime

### Purpose

To set the acceleration time of an axis group.

### Syntax

```
int HIMC_SetGrpAccTime(
    int    ctrl_id,
    int    group_id,
    double acc_time
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

acc_time [in] The acceleration time of an axis group.
Parameter unit: ms
Input range: nonzero positive value

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpAccTime |
| LabVIEW | HIMC Set Grp Acc Time.vi |
| Python | SetGrpAccTime |

## 6.3.15 HIMC_GetGrpMaxDec

### Purpose

To get the maximum profile deceleration of an axis group.

### Syntax

```
int HIMC_GetGrpMaxDec(
    int    ctrl_id,
    int    group_id,
    double *p_grp_dec
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

p_grp_dec [out] A pointer to the buffer to receive the maximum profile deceleration of an axis group.
Parameter unit: $mm/s^2$ or $deg/s^2$

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpMaxDec |
| LabVIEW | HIMC Get Grp Max Dec.vi |
| Python | GetGrpMaxDec |

## 6.3.16 HIMC_SetGrpDec

### Purpose

To set the maximum profile deceleration of an axis group.

### Syntax

```
int HIMC_SetGrpDec(
    int    ctrl_id,
    int    group_id,
    double dec
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

dec [in] The new maximum profile deceleration of an axis group.
Parameter unit: mm/s$^2$ or deg/s$^2$
Input range: >0 ~ 50000 (deceleration cannot be 0)

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpDec |
| LabVIEW | HIMC Set Grp Dec.vi |
| Python | SetGrpDec |

## 6.3.17 HIMC_SetGrpDecTime

### Purpose

To set the deceleration time of an axis group.

### Syntax

```
int HIMC_SetGrpDecTime(
    int    ctrl_id,
    int    group_id,
    double dec_time
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

dec_time [in] The deceleration time of an axis group.
Parameter unit: ms
Input range: nonzero positive value

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpDecTime |
| LabVIEW | HIMC Set Grp Dec Time.vi |
| Python | SetGrpDecTime |

## 6.3.18 HIMC_GetGrpSMTime

### Purpose

To get the profile smooth time of an axis group.

### Syntax

```
int HIMC_GetGrpSMTime(
    int    ctrl_id,
    int    group_id,
    double *p_grp_smooth_time
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| p_grp_smooth_time [out] | A pointer to the buffer to receive the profile smooth time of an axis group.<br>Parameter unit: ms |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpSMTime |
| LabVIEW | HIMC Get Grp SM Time.vi |
| Python | GetGrpSMTime |

## 6.3.19 HIMC_SetGrpSMTime

### Purpose

To set the profile smooth time of an axis group.

### Syntax

```
int HIMC_SetGrpSMTime(
    int    ctrl_id,
    int    group_id,
    double smooth_time
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

smooth_time [in] The new profile smooth time of an axis group.
Parameter unit: ms
Input range: 0 ~ 500

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpSMTime |
| LabVIEW | HIMC Set Grp SM Time.vi |
| Python | SetGrpSMTime |

## 6.3.20 HIMC_GetGrpCoordSys

### Purpose

To get the coordinate system of an axis group.

### Syntax

```
int HIMC_GetGrpCoordSys(
    int ctrl_id,
    int group_id,
    int *p_grp_coord_sys
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| p_grp_coord_sys [out] | A pointer to the buffer to receive the coordinate system of an axis group. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpCoordSys |
| LabVIEW | HIMC Get Coord Sys.vi |
| Python | GetGrpCoordSys |

## 6.3.21 HIMC_SetGrpCoordSys

### Purpose

To set the coordinate system of an axis group.

### Syntax

```
int HIMC_SetGrpCoordSys(
    int ctrl_id,
    int group_id,
    int coord_sys
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

coord_sys [in] The new coordinate system of an axis group. **Refer to section 6.1.2 for details.**
Example: 1. kCoord_MCS
2. kCoord_WCS1 | kCoord_PCS
3. kCoord_OFFSET | kCoord_WCS2 | kCoord_PCS

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpCoordSys |
| LabVIEW | HIMC Set Grp Coord Sys.vi |
| Python | SetGrpCoordSys |

## 6.3.22 HIMC_GetGrpBufferMode

### Purpose

To get the buffer mode of an axis group.

### Syntax

```
int HIMC_GetGrpBufferMode(
    int ctrl_id,
    int group_id,
    int *p_grp_buffer_mode
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| p_grp_buffer_mode [out] | A pointer to the buffer to receive the buffer mode of an axis group. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpBufferMode |
| LabVIEW | HIMC Get Buffer Mode.vi |
| Python | GetGrpBufferMode |

## 6.3.23 HIMC_SetGrpBufferMode

### Purpose

To set the buffer mode of an axis group.

### Syntax

```
int HIMC_SetGrpBufferMode(
    int ctrl_id,
    int group_id,
    int buffer_mode
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

buffer_mode [in] The new buffer mode of an axis group. **Refer to section 6.1.4 for details.**
Input range: 0 ~ 5

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpBufferMode |
| LabVIEW | HIMC Set Grp Buffer Mode.vi |
| Python | SetGrpBufferMode |

## 6.3.24 HIMC_GetGrpTransMode

### Purpose

To get the transition mode of an axis group.

### Syntax

```
int HIMC_GetGrpTransMode(
    int ctrl_id,
    int group_id,
    int *p_grp_trans_mode
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| p_grp_trans_mode [out] | A pointer to the buffer to receive the transition mode of an axis group. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpTransMode |
| LabVIEW | HIMC Get Trans Mode.vi |
| Python | GetGrpTransMode |

## 6.3.25 HIMC_SetGrpTransMode

### Purpose

To set the transition mode of an axis group.

### Syntax

```
int HIMC_SetGrpTransMode(
    int ctrl_id,
    int group_id,
    int trans_mode
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

trans_mode [in] The new transition mode of an axis group. **Refer to section 6.1.5 for details.**
Input range: 0 ~ 4

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpTransMode |
| LabVIEW | HIMC Set Grp Trans Mode.vi |
| Python | SetGrpTransMode |

## 6.3.26 HIMC_SetGrpTransPrm

### Purpose

To set the transition mode's parameters of an axis group.

### Syntax

```
int HIMC_SetGrpTransPrm(
    int    ctrl_id,
    int    group_id,
    double trans_vel,
    double trans_dis
    double trans_dev,
    double trans_curv
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| trans_vel [in] | The new transition mode's velocity parameter of an axis group.<br>**Refer to section 6.1.5 for details.** |
| trans_dis [in] | The new transition mode's distance parameter of an axis group.<br>**Refer to section 6.1.5 for details.** |
| trans_dev [in] | The new transition mode's maximum deviation parameter of an axis group.<br>**Refer to section 6.1.5 for details.** |
| trans_curv [in] | The new transition mode's curvature parameter of an axis group.<br>**Refer to section 6.1.5 for details.** |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetGrpTransPrm |
| LabVIEW | HIMC Set Grp Trans Prm.vi |
| Python | SetGrpTransPrm |

## 6.3.27 HIMC_GetGrpCmdNum

### Purpose

To get the number of commands of an axis group in the command buffer.

### Syntax

```
int HIMC_GetGrpCmdNum(
    int ctrl_id,
    int group_id,
    int *p_grp_cmd_num
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| p_grp_cmd_num [out] | A pointer to the buffer to receive the number of commands of an axis group in the command buffer. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpCmdNum |
| LabVIEW | HIMC Get Grp Cmd Num.vi |
| Python | GetGrpCmdNum |

## 6.3.28 HIMC_SetGrpVelScale

### Purpose

To set the velocity scale of axis group motion.

### Syntax

```
int HIMC_SetGrpVelScale(
    int    ctrl_id,
    int    group_id,
    double vel_scale
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

vel_scale [in] The new velocity scale of axis group motion.
Input range: 0 ~ 100

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpVelScale |
| LabVIEW | HIMC Set Grp Vel Scale.vi |
| Python | SetGrpVelScale |

## 6.3.29 HIMC_GetGrpVelScale

### Purpose

To get the velocity scale of axis group motion.

### Syntax

```
int HIMC_GetGrpVelScale(
    int    ctrl_id,
    int    group_id,
    double *p_grp_vel_scale
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis index. |
| p_grp_vel_scale [out] | A pointer to the buffer to receive the velocity scale of axis group motion.<br>Its range is from 0 to 100. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpVelScale |
| LabVIEW | HIMC Get Grp Vel Scale.vi |
| Python | GetGrpVelScale |

## 6.3.30 HIMC_GetGrpCoordTrans

### Purpose

To get the transformation parameters of axis group's coordinate system.

### Syntax

```
int HIMC_GetGrpCoordTrans(
    int    ctrl_id,
    int    group_id,
    int    coord_sys,
    double *trans_param
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis index.

coord_sys [in] Coordinate system.
**Refer to section 20.2 CoordSystem for details.**

trans_param [out] A pointer to a six-element array which contains the transformation parameters in 6-DOF {X, Y, Z, A, B, C}.
Parameter unit: mm for X, Y, Z; deg for A, B, C

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpCoordTrans |
| LabVIEW | HIMC Get Grp Coord Trans.vi |
| Python | GetGrpCoordTrans |

## 6.3.31 HIMC_SetGrpCoordTrans

### Purpose

To set the transformation parameters of axis group's coordinate system.

### Syntax

```
int HIMC_SetGrpCoordTrans(
    int    ctrl_id,
    int    group_id,
    int    coord_sys,
    double *trans_param
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis index.

coord_sys [in] Coordinate system.
**Refer to section 20.2 CoordSystem for details.**

trans_param [in] A pointer to a six-element array which contains the transformation parameters in 6-DOF {X, Y, Z, A, B, C}.
Parameter unit: mm for X, Y, Z; deg for A, B, C

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpCoordTrans |
| LabVIEW | HIMC Set Grp Coord Trans.vi |
| Python | SetGrpCoordTrans |

## 6.3.32 HIMC_GetGrpPoseCmd

### Purpose

To get the pose command of axis group's coordinate system.

### Syntax

```
int HIMC_GetGrpPoseCmd(
    int    ctrl_id,
    int    group_id,
    int    coord_sys,
    double *pose_cmd
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis index. |
| coord_sys [in] | Coordinate system.<br>**Refer to section 20.2 CoordSystem for details.** |
| pose_cmd [out] | A pointer to a six-element array which contains the pose command in 6-DOF {X, Y, Z, A, B, C}.<br>Parameter unit: mm for X, Y, Z; deg for A, B, C |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpPoseCmd |
| LabVIEW | HIMC Get Grp Pose Cmd.vi |
| Python | GetGrpPoseCmd |

## 6.3.33 HIMC_GetGrpPoseFb

### Purpose

To get the pose feedback of axis group's coordinate system.

### Syntax

```
int HIMC_GetGrpPoseFb(
    int    ctrl_id,
    int    group_id,
    int    coord_sys,
    double *pose_fb
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis index. |
| coord_sys [in] | Coordinate system.<br>**Refer to section 20.2 CoordSystem for details.** |
| pose_fb [out] | A pointer to a six-element array which contains the pose feedback in 6-DOF {X, Y, Z, A, B, C}.<br>Parameter unit: mm for X, Y, Z; deg for A, B, C |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGrpPoseFb |
| LabVIEW | HIMC Get Grp Pose Fb.vi |
| Python | GetGrpPoseFb |

# 6.4 Group status

## 6.4.1 HIMC_IsGrpEnabled

**Purpose**

To query the “enable” status of an axis group.

**Syntax**

```
int HIMC_IsGrpEnabled(
    int ctrl_id,
    int group_id,
    int *p_is_grp_enabled
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| p_is_grp_enabled [out] | A pointer to the buffer to receive the enable status of an axis group.<br>If the axis group is at the “GrpEnabled” state, the value will be 1. Otherwise, it will be 0. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_IsGrpEnabled |
| LabVIEW | HIMC Is Grp Enabled.vi |
| Python | IsGrpEnabled |

## 6.4.2 HIMC_IsGrpMoving

### Purpose

To query the “moving” status of an axis group. If the axis group is moving, PG (profile generator) continues outputting new positions.

### Syntax

```
int HIMC_IsGrpMoving(
    int ctrl_id,
    int group_id,
    int *p_is_grp_moving
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| p_is_grp_moving [out] | A pointer to the buffer to receive the moving status of an axis group.<br>If the axis group is at the “GrpMoving” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsGrpMoving |
| LabVIEW | HIMC Is Grp Moving.vi |
| Python | IsGrpMoving |

## 6.4.3 HIMC_IsGrpInPos

**Purpose**

To query the “in-position” status of an axis group. If the axis group is in-position, all axes in the group are in-position.

**Syntax**

```
int HIMC_IsGrpInPos(
    int ctrl_id,
    int group_id,
    int *p_is_grp_inpos
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| p_is_grp_inpos [out] | A pointer to the buffer to receive the in-position status of an axis group.<br>If the axis group is at the “GrpInPos” state, the value will be 1. Otherwise, it will be 0. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_IsGrpInPos |
| LabVIEW | HIMC Is Grp In Pos.vi |
| Python | IsGrpInPos |

## 6.4.4 HIMC_IsGrpErrorStop

### Purpose

To query whether the axis group is at the “error stop” state.

### Syntax

```
int HIMC_IsGrpErrorStop(
    int ctrl_id,
    int group_id,
    int *p_is_grp_errorstop
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| p_is_grp_errorstop [out] | A pointer to the buffer to receive the error stop status of an axis group.<br>If the axis group is at the “GrpErrorStop” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsGrpErrorStop |
| LabVIEW | HIMC Is Grp Error Stop.vi |
| Python | IsGrpErrorStop |

# 6.5 Advanced group motion control

## 6.5.1 HIMC_LineAbs

**Purpose**

To command an interpolated linear movement on an axis group toward an absolute position in the specific coordinate system.

**Syntax**

```
int HIMC_LineAbs(
    int ctrl_id,
    int group_id,
    CoordPosition *target_pos,
    MotionProfile *motion_profile,
    CoordSystem coord_sys,
    MotionBufferMode buff_mode,
    MotionTransitionMode trans_mode,
    TransPrm *trans_prm
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| target_pos [in] | A pointer to the buffer to store the 6-DOF target position of the tool center point (end-effector).<br>Parameter unit: mm for X, Y, Z; deg for A, B, C<br>**Refer to section 19.2 CoordPosition for details.** |
| motion_profile [in] | A pointer to the buffer to store motion profile settings of the tool center point (end-effector).<br>**Refer to section 19.3 MotionProfile for details.** |
| coord_sys [in] | Specify the applicable coordinate system.<br>**Refer to section 20.2 CoordSystem for details.** |
| buff_mode [in] | Specify the path buffer mode.<br>**Refer to section 20.3 MotionBufferMode for details.** |

trans_mode [in] Specify the path transition mode.
**Refer to section 20.4 MotionTransitionMode for details.**

trans_prm [in] Specify the pointer for specific transition mode.
**Refer to section 19.6 TransPrm for details.**

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_LineAbs |
| LabVIEW | HIMC Line Abs.vi |
| Python | LineAbs |

## 6.5.2 HIMC_LineRel

**Purpose**

To command an interpolated linear movement on an axis group toward a relative position in the specific coordinate system.

**Syntax**

```
int HIMC_LineRel(
    int ctrl_id,
    int group_id,
    CoordPosition *relative_dist,
    MotionProfile *motion_profile,
    CoordSystem coord_sys,
    MotionBufferMode buff_mode,
    MotionTransitionMode trans_mode,
    TransPrm *trans_prm
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| relative_dist [in] | A pointer to the buffer to store the relative distance for 6-DOF of the tool center point (end-effector).<br>Parameter unit: mm for X, Y, Z; deg for A, B, C<br>**Refer to section 19.2 CoordPositionfor details.** |
| motion_profile [in] | A pointer to the buffer to store motion profile settings of the tool center point (end-effector).<br>**Refer to section 19.3 MotionProfile for details.** |
| coord_sys [in] | Specify the applicable coordinate system.<br>**Refer to section 20.2 CoordSystem for details.** |
| buff_mode [in] | Specify the path buffer mode.<br>**Refer to section 20.3 MotionBufferMode for details.** |
| trans_mode [in] | Specify the path transition mode.<br>**Refer to section 20.4 MotionTransitionMode for details.** |

trans_prm [in] Specify the pointer for specific transition mode.

**Refer to section 19.6 TransPrm for details.**

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_LineRel |
| LabVIEW | HIMC Line Rel.vi |
| Python | LineRel |

## 6.5.3 HIMC_CircleAbs

**Purpose**

To command an interpolated circular movement on an axis group toward an absolute position in the specific coordinate system.

**Syntax**

```
int HIMC_CircleAbs(
    int ctrl_id,
    int group_id,
    CenterPosition *center_pos,
    NormalVector *normal_vector,
    int turns,
    CoordPosition *target_pos,
    MotionProfile *motion_profile,
    CoordSystem coord_sys,
    MotionBufferMode buff_mode,
    MotionTransitionMode trans_mode,
    TransPrm *trans_prm
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| center_pos [in] | A pointer to the buffer to store the position of center point.<br>Parameter unit: mm<br>**Refer to section 19.4 CenterPosition for details.** |
| normal_vector [in] | A pointer to the buffer to store normal vector of circle.<br>**Refer to section 19.5 NormalVector for details.** |
| turns [in] | Number of turns of circular path relative to the start point.<br>It determines the direction and the total angle of circular path. |
| target_pos [in] | A pointer to the buffer to store the 6-DOF target position of the tool center point (end-effector).<br>Parameter unit: mm for X, Y, Z; deg for A, B, C<br>**Refer to section 19.2 CoordPosition for details.** |

motion_profile [in] A pointer to the buffer to store motion profile settings of the tool center point (end-effector).
**Refer to section 19.3 MotionProfile for details.**

coord_sys [in] Specify the applicable coordinate system.
**Refer to section 20.2 CoordSystem for details.**

buff_mode [in] Specify the path buffer mode.
**Refer to section 20.3 MotionBufferMode for details.**

trans_mode [in] Specify the path transition mode.
**Refer to section 20.4 MotionTransitionMode for details.**

trans_prm [in] Specify the pointer for specific transition mode.
**Refer to section 19.6 TransPrm for details.**

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_CircleAbs |
| LabVIEW | HIMC Circle Abs.vi |
| Python | CircleAbs |

## 6.5.4 HIMC_CircleRel

**Purpose**

To command an interpolated circular movement on an axis group toward a relative position in the specific coordinate system.

**Syntax**

```
int HIMC_CircleRel(
    int ctrl_id,
    int group_id,
    CenterPosition *center_pos,
    NormalVector *normal_vector,
    int turns,
    CoordPosition *relative_dist,
    MotionProfile *motion_profile,
    CoordSystem coord_sys,
    MotionBufferMode buff_mode,
    MotionTransitionMode trans_mode,
    TransPrm *trans_prm
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| center_pos [in] | A pointer to the buffer to store the position of center point.<br>Parameter unit: mm<br>**Refer to section 19.4 CenterPosition for details.** |
| normal_vector [in] | A pointer to the buffer to store normal vector of circle.<br>**Refer to section 19.5 NormalVector for details.** |
| turns [in] | Number of turns of circular path relative to the start point.<br>It determines the direction and the total angle of circular path. |
| relative_dist [in] | A pointer to the buffer to store the relative distance for 6-DOF of the tool center point (end-effector).<br>Parameter unit: mm for X, Y, Z; deg for A, B, C<br>**Refer to section 19.2 CoordPosition for details.** |

motion_profile [in] A pointer to the buffer to store motion profile settings of the tool center point (end-effector).
**Refer to section 19.3 MotionProfile for details.**

coord_sys [in] Specify the applicable coordinate system.
**Refer to section 20.2 CoordSystem for details.**

buff_mode [in] Specify the path buffer mode.
**Refer to section 20.3 MotionBufferMode for details.**

trans_mode [in] Specify the path transition mode.
**Refer to section 20.4 MotionTransitionMode for details.**

trans_prm [in] Specify the pointer for specific transition mode.
**Refer to section 19.6 TrnsPrm for details.**

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_CircleRel |
| LabVIEW | HIMC Circle Rel.vi |
| Python | CircleRel |

# 7. GPIO functions


7. GPIO functions ..... 7-1
7.1 Overview ..... 7-2
7.1.1 GPIO variables ..... 7-2
7.2 Controller IO setting ..... 7-3
7.2.1 HIMC_SetGPO ..... 7-3
7.2.2 HIMC_ToggleGPO ..... 7-4
7.2.3 HIMC_SetAllGPO ..... 7-5
7.2.4 HIMC_SetGPIInvert ..... 7-6
7.2.5 HIMC_SetGPOInvert ..... 7-7
7.2.6 HIMC_BindEMO ..... 7-8
7.3 Slave IO setting ..... 7-9
7.3.1 HIMC_SetSlaveGpo ..... 7-9
7.3.2 HIMC_ToggleSlvGPO ..... 7-10
7.3.3 HIMC_SetSlvAllGPO ..... 7-11
7.4 Controller IO status ..... 7-12
7.4.1 HIMC_GetGPI ..... 7-12
7.4.2 HIMC_GetGPO ..... 7-13
7.4.3 HIMC_GetAllGPI ..... 7-14
7.4.4 HIMC_GetAllGPO ..... 7-15
7.5 Slave IO status ..... 7-16
7.5.1 HIMC_GetSlvGPI ..... 7-16
7.5.2 HIMC_GetSlvGPO ..... 7-17

# 7.1 Overview

HIMC provides 8 sets of general purpose input / output (GPIO) pins; the hardware delay time is within 1ms and the power is 24V. The slave device can connect to the controller via CoE communication and update its IO status. The number of IO depends on the slave device. With the functions provided in this chapter, such as “SetGPO” and “SetSlvGPO”, users can set the signal of output pin respectively for HIMC and slave. Besides, users can query the signal status of input / output pin. With iA Studio’s function module “Digital IO” (refer to section 4.4 in “iA Studio User Guide”), users can observe and set HIMC’s and slave’s input / output status.

HIMC’s digital input “I8” is E-stop signal (refer to section 3.3 in “HIMC Installation Guide”), which will be triggered when receiving rising-edge signal. At this time, all axes will be disabled and all HMPL tasks will be stopped.

**Note: After rising-edge signal is triggered, users can re-enable the axes or re-execute HMPL tasks.**

## 7.1.1 GPIO variables

Users can select the desired controller’s general purpose input/output system variables via Scope Manager in iA Studio (refer to section 4.8 in “iA Studio User Guide”). Detailed descriptions are shown in Table 7.1.1.1.

Table 7.1.1.1 Controller’s general purpose input / output variables

| Name | Variable | Unit | Description |
|---|---|---|---|
| HIMC GPO | himc_gpo_bits | N/A | State of controller’s general purpose output pin. |
| HIMC GPI | himc_gpi_bits | N/A | State of controller’s general purpose input pin. |

# 7.2 Controller IO setting

## 7.2.1 HIMC_SetGPO

### Purpose

To set the state of the controller's general purpose output.

### Syntax

```
int HIMC_SetGPO(
    int  ctrl_id,
    int  gpo_idx,
    char state
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| gpo_idx [in] | General purpose output index. |
| state [in] | The state to be set. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGPO |
| LabVIEW | HIMC Set GPO.vi |
| Python | SetGPO |

## 7.2.2 HIMC_ToggleGPO

### Purpose

To toggle the state of the controller's general purpose output.

### Syntax

```
int HIMC_ToggleGPO(
    int ctrl_id,
    int gpo_idx
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

gpo_idx [in] General purpose output index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ToggleGPO |
| LabVIEW | HIMC Toggle GPO.vi |
| Python | ToggleGPO |

## 7.2.3 HIMC_SetAllGPO

### Purpose

To set the states of the controller's multiple general purpose outputs.

### Syntax

```
int HIMC_SetAllGPO(
    int ctrl_id,
    int all_gpo_state
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

all_gpo_state [in] State value of all general purpose outputs.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetAllGPO |
| LabVIEW | HIMC Set All GPO.vi |
| Python | SetAllGPO |

## 7.2.4 HIMC_SetGPIInvert

### Purpose

To set the invert state of the controller's general purpose input.

### Syntax

```
int HIMC_SetGPIInvert(
    int  ctrl_id,
    int  gpi_idx,
    char invert
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| gpi_idx [in] | General purpose input index. |
| invert [in] | The invert state to be set. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGPIInvert |
| LabVIEW | HIMC Set GPI Invert.vi |
| Python | SetGPIInvert |

## 7.2.5 HIMC_SetGPOInvert

### Purpose

To set the invert state of the controller's general purpose output.

### Syntax

```
int HIMC_SetGPOInvert(
    int  ctrl_id,
    int  gpo_idx,
    char invert
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| gpo_idx [in] | General purpose output index. |
| invert [in] | The invert state to be set. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGPOInvert |
| LabVIEW | HIMC Set GPO Invert.vi |
| Python | SetGPOInvert |

## 7.2.6 HIMC_BindEMO

### Purpose

To set the general purpose input pin to be bound to E-Stop.

### Syntax

```
int HIMC_BindEMO(
    int ctrl_id,
    int gpi_idx
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

gpi_idx [in] General purpose input index. The default value is 8.
If it is set as 0, all general purpose input pins are not bound to E-Stop.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_BindEMO |
| LabVIEW | HIMC Bind EMO.vi |
| Python | BindEMO |

# 7.3 Slave IO setting

## 7.3.1 HIMC_SetSlvGPO

**Purpose**

To set the state of the slave's general purpose output.

**Syntax**

```
int HIMC_SetSlvGPO(
    int  ctrl_id,
    int  slv_slot_id,
    int  gpo_idx,
    int  on_off
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| gpo_idx [in] | General purpose output index. |
| on_off [in] | The state to be set. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

Users must configure Digital output object as PDO when using this function. For example, to set 0x60FE(Digital outputs) of the servo drive as PDO.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetSlvGPO |
| LabVIEW | HIMC Set Slv GPO.vi |
| Python | SetSlvGPO |

## 7.3.2 HIMC_ToggleSlvGPO

### Purpose

To toggle the state of the slave's general purpose output.

### Syntax

```
int HIMC_ToggleSlvGPO(
    int ctrl_id,
    int slv_slot_id,
    int gpo_idx
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| gpo_idx [in] | General purpose output index. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure Digital output object as PDO when using this function. For example, to set 0x60FE(Digital outputs) of the servo drive as PDO.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ToggleSlvGPO |
| LabVIEW | HIMC Toggle Slv GPO.vi |
| Python | ToggleSlvGPO |

## 7.3.3 HIMC_SetSlvAllGPO

### Purpose

To set the states of the slave's multiple general purpose outputs.

### Syntax

```
int HIMC_SetSlvAllGPO(
    int  ctrl_id,
    int  slv_slot_id,
    int  all_gpo_state
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored |
| all_gpo_state [in] | State value of all general purpose outputs. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure Digital output object as PDO when using this function. For example, to set 0x60FE(Digital outputs) of the servo drive as PDO.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSlvAllGPO |
| LabVIEW | HIMC Set Slv All GPO.vi |
| Python | SetSlvAllGPO |

# 7.4 Controller IO status

## 7.4.1 HIMC_GetGPI

### Purpose

To get the state of the controller's general purpose input.

### Syntax

```
int HIMC_GetGPI(
    int   ctrl_id,
    int   gpi_idx,
    char *p_state
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

gpi_idx [in] General purpose input index.

p_state [out] A pointer to the buffer to receive the state of the specific input.
If the input is at the "on" state, the value will be 1. Otherwise, it will be 0.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGPI |
| LabVIEW | HIMC Get GPI.vi |
| Python | GetGPI |

## 7.4.2 HIMC_GetGPO

### Purpose

To get the state of the controller's general purpose output.

### Syntax

```
int HIMC_GetGPO(
    int   ctrl_id,
    int   gpo_idx,
    char *p_state
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| gpo_idx [in] | General purpose output index. |
| p_state [out] | A pointer to the buffer to receive the state of the specific output.<br>If the output is at the "on" state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGPO |
| LabVIEW | HIMC Get GPO.vi |
| Python | GetGPO |

## 7.4.3 HIMC_GetAllGPI

### Purpose

To get the states of the controller's multiple general purpose inputs.

### Syntax

```
int HIMC_GetAllGPI(
    int  ctrl_id,
    int *p_state
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| p_state [out] | A pointer to the buffer to receive the state of the controller's general purpose input. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetAllGPI |
| LabVIEW | HIMC Get All GPI.vi |
| Python | GetAllGPI |

## 7.4.4 HIMC_GetAllGPO

### Purpose

To get the states of the controller's multiple general purpose outputs.

### Syntax

```
int HIMC_GetAllGPO(
    int  ctrl_id,
    int *p_state
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| p_state [out] | A pointer to the buffer to receive the state of the controller's general purpose output. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetAllGPO |
| LabVIEW | HIMC Get All GPO.vi |
| Python | GetAllGPO |

# 7.5 Slave IO status

## 7.5.1 HIMC_GetSlvGPI

### Purpose

To get the state of the slave's general purpose input.

### Syntax

```
int HIMC_GetSlvGPI(
    int  ctrl_id,
    int  slv_slot_id,
    int  gpi_idx,
    int *p_state
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| gpi_idx [in] | General purpose input index. |
| p_state [out] | A pointer to the buffer to receive the state of the specific input.<br>If the input is at the "on" state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure Digital input object as PDO when using this function. For example, to set 0x60FE(Digital inputs) of the servo drive as PDO.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSlvGPI |
| LabVIEW | HIMC Get Slv GPI.vi |
| Python | GetSlvGPI |

## 7.5.2 HIMC_GetSlvGPO

### Purpose

To get the state of the slave's general purpose output.

### Syntax

```
int HIMC_GetSlvGPO(
    int  ctrl_id,
    int  slv_slot_id,
    int  gpo_idx,
    int *p_state
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| gpi_idx [in] | General purpose output index. |
| p_state [out] | A pointer to the buffer to receive the state of the specific output.<br>If the output is at the "on" state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure Digital output object as PDO when using this function. For example, to set 0x60FE(Digital outputs) of the servo drive as PDO.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSlvGPO |
| LabVIEW | HIMC Get Slv GPO.vi |
| Python | GetSlvGPO |

(This page is intentionally left blank.)

# 8. AIO functions


8. AIO functions ..... 8-1
8.1 Overview ..... 8-2
8.2 Slave AIO setting ..... 8-3
8.2.1 HIMC_SetSlvAIType ..... 8-3
8.2.2 HIMC_SetSlvAOType ..... 8-4
8.2.3 HIMC_SetSlvAOHex ..... 8-5
8.2.4 HIMC_SetSlvAO ..... 8-6
8.3 Slave AIO status ..... 8-7
8.3.1 HIMC_GetSlvAIType ..... 8-7
8.3.2 HIMC_GetSlvAOType ..... 8-8
8.3.3 HIMC_GetSlvAIHex ..... 8-9
8.3.4 HIMC_GetSlvAI ..... 8-10
8.3.5 HIMC_GetSlvAOHex ..... 8-11
8.3.6 HIMC_GetSlvAO ..... 8-12
8.4 Slave AO bound to HIMC internal buffer variable ..... 8-13
8.4.1 HIMC_ SetSlvAOMonitor ..... 8-13
8.4.2 HIMC_SetSlvAOParam ..... 8-15
8.4.3 HIMC_GetSlvAOScale ..... 8-17
8.4.4 HIMC_GetSlvAOOffset ..... 8-18
8.4.5 HIMC_IsSlvAOBound ..... 8-19

# 8.1 Overview

With AIO functions, slaves with analog input (AI) or analog output (AO) function can read and set the related parameters. Among them, HMPL provides users with the setting of specifying conversion type between digital and analog. Detailed specifications are shown in Table 8.1.

Table 8.1.1 Definition of analog data conversion type

| Conversion type | Conversion description | Specification | Remark |
|---|---|---|---|
| KDAC_NONE | N/A | Use only Hex function to operate this conversion. | - |
| KDAC_N10_P10 | -10~10 | Convert analog Hex value to -10~10. | Common in analog voltage module |
| KDAC_P0_P10 | 0~10 | Convert analog Hex value to 0~10. | |
| KDAC_N5_P5 | -5~5 | Convert analog Hex value to -5~10. | |
| KDAC_P0_P5 | 0~5 | Convert analog Hex value to -0~10. | |
| KDAC_P0_P20 | 0~20 | Convert analog Hex value to 0~20. | Common in analog Current module |
| KDAC_P4_P20 | 4~20 | Convert analog Hex value to 4~20. | |
| KDAC_N20_P20 | -20~20 | Convert analog Hex value to -20~20. | |
| KDAC_SIGNED | High byte defines whether the sign is positive or negative. | 0 represents positive number in high byte; 1 represents negative number in high byte. | - |

# 8.2 Slave AIO setting

## 8.2.1 HIMC_SetSlvAIType

### Purpose

To set the conversion type of the slave's analog input value.

### Syntax

```
int HIMC_SetSlvAIType(
    int ctrl_id,
    int slv_slot_id,
    int ai_idx,
    int range_type
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| Ai idx [in] | Analog input channel. |
| range_type [in] | Analog data conversion type. Refer to Table 8.1 for details. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSlvAOType |
| LabVIEW | HIMC Set Slv AO Type.vi |
| Python | SetSlvAOType |

## 8.2.2 HIMC_SetSlvAOType

### Purpose

To set the conversion type of the slave's analog output value.

### Syntax

```
int HIMC_SetSlvAOType(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    int range_type
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

slv_slot_id [in] Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored.

ai idx [in] Analog output channel.

range_type [in] Analog data conversion type. Refer to Table 8.1 for details.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSlvAOType |
| LabVIEW | HIMC Set Slv AO Type.vi |
| Python | SetSlvAOType |

## 8.2.3 HIMC_SetSlvAOHex

### Purpose

To set the analog output Hex value of the slave.

### Syntax

```
int HIMC_SetSlvAOHex(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    int ao_hex_val
);
```

### .Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and its Slot ID. Slot ID could be ignored if there is no slot on the slave. |
| ao idx [in] | Analog output channel. |
| ao_hex_val [in] | Analog 32bit output value. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure Analog output object as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSlvAOHex |
| LabVIEW | HIMC Set Slv AO Hex.vi |
| Python | SetSlvAOHex |

## 8.2.4 HIMC_SetSlvAO

### Purpose

To set the analog output value of the slave.

### Syntax

```
int HIMC_SetSlvAO(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    double ao_val
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and its Slot ID. Slot ID could be ignored if there is no slot on the slave. |
| ao idx [in] | Analog output channel. |
| ao_value [in] | Analog output value. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure Analog output object as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSlvAO |
| LabVIEW | HIMC Set Slv AO.vi |
| Python | SetSlvAO |

# 8.3 Slave AIO status

## 8.3.1 HIMC_GetSlvAIType

### Purpose

To get the analog input type of the slave.

### Syntax

```
int HIMC_GetSlvAIType(
    int ctrl_id,
    int slv_slot_id,
    int ai_idx,
    int *p_ai_type
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and its Slot ID. Slot ID could be ignored if there is no slot on the slave. |
| ai_idx [in] | Analog input channel. |
| p_ai_type [out] | A pointer to the buffer to receive the analog input type of the slave**.**<br>**Refer to Table 8.1 for details.** |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSlvAIType |
| LabVIEW | HIMC Get Slv AI Type.vi |
| Python | GetSlvAIType |

## 8.3.2 HIMC_GetSlvAOType

### Purpose

To get the analog input type of the slave.

### Syntax

```
int HIMC_GetSlvAOType(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    int *p_ao_type
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and its Slot ID. Slot ID could be ignored if there is no slot on the slave. |
| ao_idx [in] | Analog output channel. |
| p_ao_type [out] | A pointer to the buffer to receive the analog output type of the slave.<br>**Refer to table 8.1 for details.** |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSlvAOType |
| LabVIEW | HIMC Get Slv AO Type.vi |
| Python | GetSlvAOType |

## 8.3.3 HIMC_GetSlvAIHex

### Purpose

To get the analog input Hex value of the slave.

### Syntax

```
int HIMC_GetSlvAIHex(
    int ctrl_id,
    int slv_slot_id,
    int ai_idx,
    int *p_ai_hex_val
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| ai_idx [in] | Analog input channel. |
| p_ai_hex_val [out] | A pointer to the buffer to receive the analog input Hex value. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure Analog input object as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSlvAIHex |
| LabVIEW | HIMC Get Slv AI Hex.vi |
| Python | GetSlvAIHex |

## 8.3.4 HIMC_GetSlvAI

### Purpose

To get the analog input of the slave.

### Syntax

```
int HIMC_GetSlvAI(
    int ctrl_id,
    int slv_slot_id,
    int ai_idx,
    double *p_ai_val
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| ai_idx [in] | Analog input channel. |
| p_ai_offset [out] | A pointer to the buffer to receive the analog input value. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure Analog input object as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSlvAI |
| LabVIEW | HIMC Get Slv AI.vi |
| Python | GetSlvAI |

## 8.3.5 HIMC_GetSlvAOHex

### Purpose

To get the analog output Hex value of the slave.

### Syntax

```
int HIMC_GetSlvAOHex(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    int *p_ao_hex_val
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| ao_idx [in] | Analog output channel. |
| p_ao_hex_val [out] | A pointer to the buffer to receive the analog output Hex value. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure Analog output object as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSlvAOHex |
| LabVIEW | HIMC Get Slv AOHex.vi |
| Python | GetSlvAOHex |

## 8.3.6 HIMC_GetSlvAO

### Purpose

To get the analog output value of the slave.

### Syntax

```
int HIMC_GetSlvAO(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    double *p_ao_val
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| slot_idx [in] | Index of analog output module installation slot. |
| ao_idx [in] | Analog output channel. |
| p_ao_val [out] | A pointer to the buffer to receive the analog output value. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure Analog output object as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetSlvAO |
| LabVIEW | HIMC Get Slv AO.vi |
| Python | GetSlvAO |

# 8.4 Slave AO bound to HIMC internal buffer variable

## 8.4.1 HIMC_ SetSlvAOMonitor

### Purpose

To set the controller variable to be bound to analog output.

### Syntax

```
int HIMC_SetSlvAOMonitor(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    int var_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

slv_slot_id [in] Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored.

ao_idx [in] Analog output channel.

var_id [in] Controller variable and axis ID. The following tables show the definition of controller variable and axis ID.
**Refer to section 16.1.2 for more controller variables.**
Example: HMPL_AXIS_0 | HMPL_REF_VEL

| Controller variable | Definition | Controller variable | Definition |
|---|---|---|---|
| HMPL_REF_POS | Reference position of axis | HMPL_POS_FB | Position feedback of axis |
| HMPL_REF_VEL | Reference velocity of axis | HMPL_VEL_FB | Velocity feedback of axis |
| HMPL_REF_ACC | Reference acceleration of axis | HMPL_ACC_FB | Acceleration feedback of axis |
| | | HMPL_CUR_FB | Current feedback of axis |

| Axis ID | Definition |
|---|---|
| HMPL_AXIS_0 | Axis 0 |
| HMPL_AXIS_1 | Axis 1 |
| … | … |
| HMPL_AXIS_15 | Axis 15 |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSlvAOMonitor |
| LabVIEW | HIMC Set Slv AO Monitor.vi |
| Python | SetSlvAOMonitor |

## 8.4.2 HIMC_SetSlvAOParam

### Purpose

To set the slave's analog output bound to controller variable.

### Syntax

```
int HIMC_SetSlvAOParam(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    int ao_en_bind,
    double ao_scale,
    double ao_offset
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| ao_idx [in] | Analog output channel. |
| ao_en_bind [in] | 0: The function of analog output bound to controller variable is OFF (default)<br>1: The function of analog output bound to controller variable is ON |
| ao_scale [in] | Scale of analog output and controller variable. |
| ao_offset [in] | Offset of analog output and controller variable. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSlvAOParam |
| LabVIEW | HIMC Set Slv AO Param.vi |
| Python | SetSlvAOParam |

## 8.4.3 HIMC_GetSlvAOScale

### Purpose

To set scale of analog output and controller variable.

### Syntax

```
int HIMC_SetSlvAOScale(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    double *p_ao_scale
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| ao_idx [in] | Analog output channel. |
| p_ao_scale [in] | A pointer to the buffer to receive scale of analog output and controller variable. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetSlvAOScale |
| LabVIEW | HIMC Set Slv AO Scale.vi |
| Python | SetSlvAOScale |

## 8.4.4 HIMC_GetSlvAOOffset

**Purpose**

To get the slave's offset of analog output and controller variable.

**Syntax**

```
int HIMC_SetSlvAOOffset(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    double *p_ao_offset
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| ao_idx [in] | Analog output channel. |
| p_ao_offset [in] | A pointer to the buffer to receive offset of analog output and controller variable. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_GetSlvAOOffset |
| LabVIEW | HIMC Get Slv AO Offset.vi |
| Python | GetSlvAOOffset |

## 8.4.5 HIMC_IsSlvAOBound

**Purpose**

To query whether the slave's analog output is bound to controller variable.

**Syntax**

```
int HIMC_IsSlvAOBound(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    int *p_is_bound
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_slot_id [in] | Slave ID and Slot ID. If there is no slot on the slave, Slot ID can be ignored. |
| ao_idx [in] | Analog output channel. |
| p_ao_bound [in] | A pointer to the buffer to receive whether the slave's analog output is bound to controller variable. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_GetSlvAOBound |
| LabVIEW | HIMC Get Slv AO Bound.vi |
| Python | GetSlvAO Bound |

(This page is intentionally left blank.)

# 9. User Table functions


9. User Table functions ..... 9-1
9.1 Overview ..... 9-2
9.2 HIMC_SetUserTable ..... 9-3
9.3 HIMC_GetUserTable ..... 9-4
9.4 HIMC_SetTableValue ..... 9-5
9.5 HIMC_GetTableValue ..... 9-6
9.6 HIMC_SaveUserTable ..... 9-7
9.7 HIMC_LoadUserTable ..... 9-8

# 9.1 Overview

HIMC provides users with free memory space, which can store up to 512,000 double-type variable data (500K Bytes). With the functions provided in this chapter, users can access memory space. The written values will be stored to controller's random access memory (RAM). With function "HIMC_SaveUserTable", User Table's data in memory space will be saved to HIMC's hard disk space. After HIMC is power cycled, with function "HIMC_LoadUserTable", the stored data will be copied to User Table's memory space again.

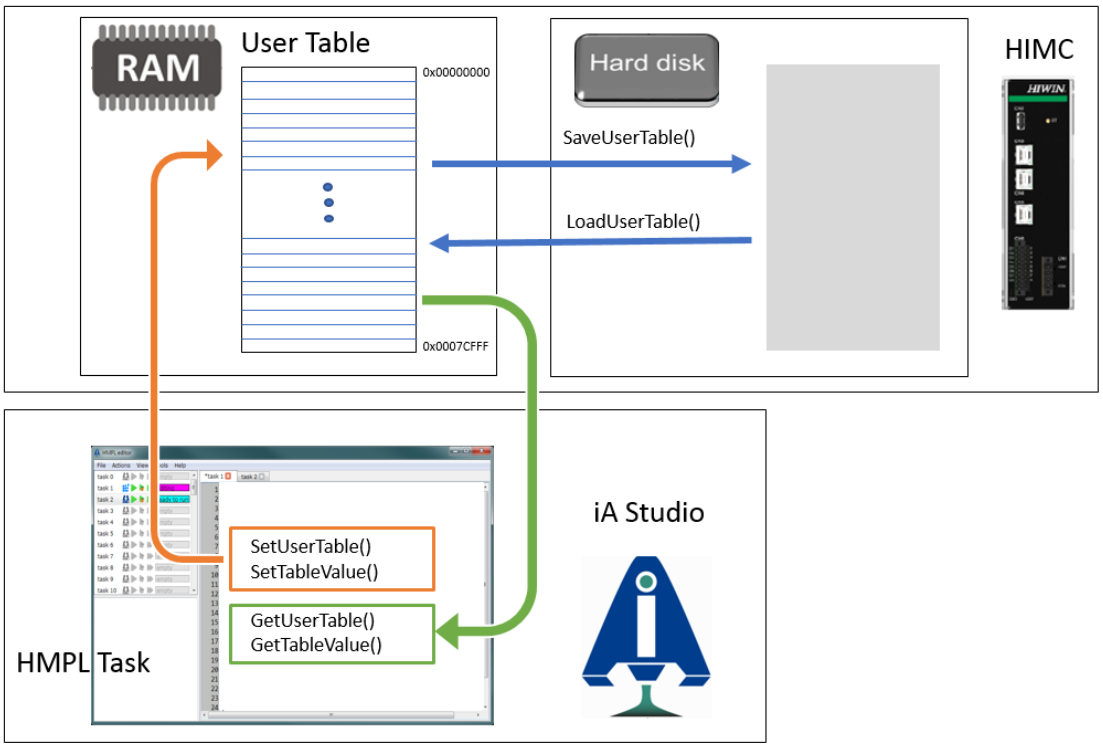



Figure 9.1.1

**Note: Users can access User Table's variable values via Table Viewer in iA Studio (refer to section 4.11 in "iA Studio User Guide"), including loading and saving to HIMC's memory and hard disk.**

Attention:
The error map used by dynamic error compensation functions is stored in User Table's memory space. When enabling dynamic error compensation, users should ensure the access to other User Table values will not affect the built error compensation values.

# 9.2 HIMC_SetUserTable

**Purpose**

To set user table data to the controller.

**Syntax**

```
int HIMC_SetUserTable(
    int    ctrl_id,
    double *p_user_table_data,
    int    start_idx,
    int    number_of_doubles_to_write
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| p_user_table_data [in] | A pointer to the buffer to write the data to user table. |
| start_idx [in] | Start index of user table. |
| number_of_doubles_to_write [in] | Number of user table to be written. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetUserTable |
| LabVIEW | HIMC Set User Table.vi |
| Python | SetUserTable |

# 9.3 HIMC_GetUserTable

## Purpose

To get user table data from the controller.

## Syntax

```
int HIMC_GetUserTable(
    int    ctrl_id,
    double *p_user_table_data,
    int    start_idx,
    int    number_of_doubles_to_read
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| p_user_table_data [out] | A pointer to the buffer to receive the data from user table. |
| start_idx [in] | Start index of user table. |
| number_of_doubles_to_read [in] | Number of user table to be read. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetUserTable |
| LabVIEW | HIMC Get User Table.vi |
| Python | GetUserTable |

# 9.4 HIMC_SetTableValue

## Purpose

To write data to the specific index of user table.

## Syntax

```
int HIMC_SetTableValue(
    int    ctrl_id,
    int    index,
    double value
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| index [in] | Index of user table. |
| value [in] | Input data. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetTableValue |
| LabVIEW | HIMC Set Table Value.vi |
| Python | SetTableValue |

# 9.5 HIMC_GetTableValue

## Purpose

To get data from the specific index of user table.

## Syntax

```
int HIMC_GetTableValue(
    int    ctrl_id,
    int    index,
    double *value
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| index [in] | Index of user table. |
| value [out] | A pointer to the buffer to receive the data. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetTableValue |
| LabVIEW | HIMC Get Table Value.vi |
| Python | GetTableValue |

# 9.6 HIMC_SaveUserTable

## Purpose

Store user table data in RAM to permanent memory.

## Syntax

```
int HIMC_SaveUserTable(
    int ctrl_id,
    int start_idx,
    int num_data
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| start_idx [in] | Start index of user table. |
| num_data [in] | Number of elements to be stored. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SaveUserTable |
| LabVIEW | HIMC Save User Table.vi |
| Python | SaveUserTable |

# 9.7 HIMC_LoadUserTable

## Purpose

Load user table data from permanent memory to RAM.

## Syntax

```
int HIMC_LoadUserTable(
    int ctrl_id,
    int start_idx,
    int num_data
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| start_idx [in] | Start index of user table. |
| num_data [in] | Number of elements to be stored. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_LoadUserTable |
| LabVIEW | HIMC Load User Table.vi |
| Python | LoadUserTable |

# 10. Position Trigger functions


10. Position Trigger functions ..... 10-1
10.1 Overview ..... 10-2
10.1.1 PT variables ..... 10-2
10.1.2 Flow of using PT function ..... 10-4
10.2 HIMC_EnablePT ..... 10-5
10.3 HIMC_DisablePT ..... 10-6
10.4 HIMC_IsPTEnabled ..... 10-7
10.5 HIMC_SetPosTriggerConfig ..... 10-8

# 10.1 Overview

HIMC position trigger function can only be used with HIWIN Mikrosystem servo drive. With HMPL commands, users can operate PT (position trigger) related functions. Before operating PT related functions, please consult HIWIN Mikrosystem or local distributors for compatible drives.

**Note: The requirements for HIWIN Mikrosystem servo drives to use PT related functions are (1) digital encoder (2) complete homing procedure first.**

## 10.1.1 PT variables

PT related functions are operated based on the variables listed in Table 10.1.1.1.

Table 10.1.1.1

| Name | Type | Unit | Description | HIMC API function |
|---|---|---|---|---|
| status | int | true / false | Status of PT function, which indicates whether PT is still functioning. | HIMC_EnablePT<br>HIMC_DisablePT<br>HIMC_IsPTEnabled |
| start position | double | mm or deg | Start position of PT function. PT output signal train starts at this point. | HIMC_ SetPosTriggerConfig |
| end position | double | mm or deg | End position of PT function. No more PT output signal is sent out after this point. | |
| interval | double | mm or deg | Position interval between consecutive PT outputs. | |
| pulse width | int | ns | The width of each PT output signal.<br>For E1 series servo drive, the range is from 20 ns to 80,000 ns, with the minimum increment of 20 ns. For example, 20, 40, ... 80,000 ns. | |

In Figure 10.1.1.1, the polarity is set to be “active high”.

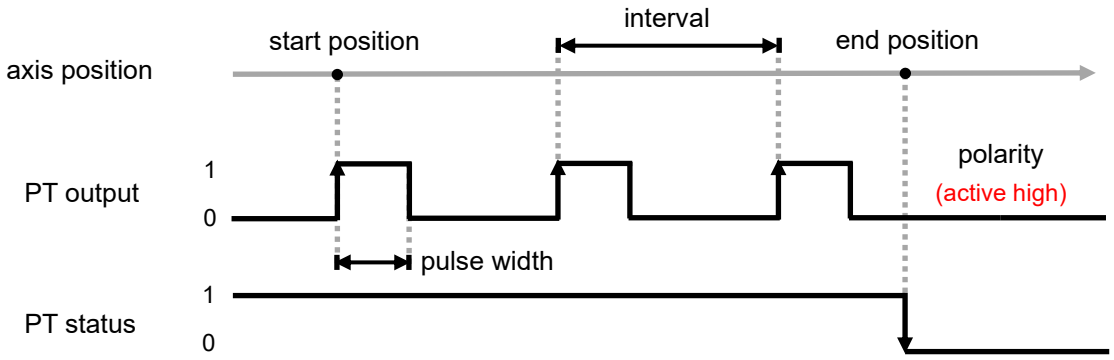



Figure 10.1.1.1

In Figure 10.1.1.2, the polarity is set to be “active low”.

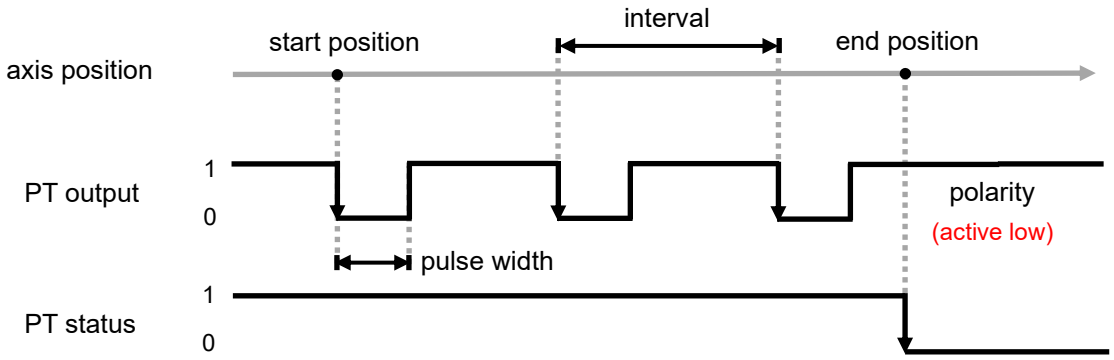



Figure 10.1.1.2

Limitation:

The interval of PT function and the velocity of axis must fit in the formula “Velocity < Interval x Position sampling rate”. If the interval is set as 100 um and the position sampling rate is 16 K, the velocity must be less than 1600 mm/s.

**Note: To adjust PT function’s output polarity (active high / low), go to servo drive’s HMI for setting. After saving the setting, power cycle servo drive to make the output polarity be effective.**

## 10.1.2 Flow of using PT function

◆ Fixed interval PT function

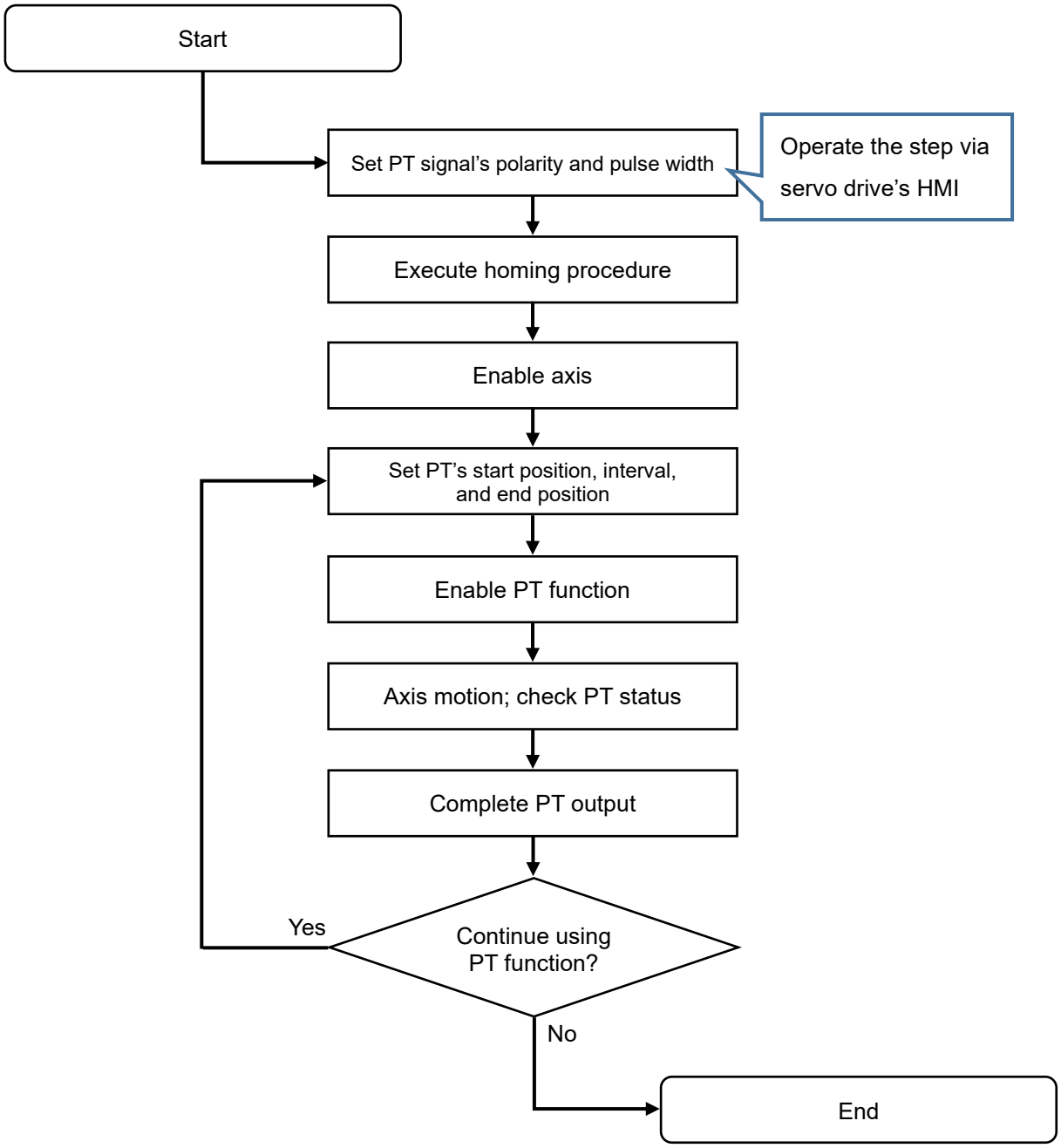



Figure 10.1.2.1

## 10.2 HIMC_EnablePT

### Purpose

To enable the position trigger function of an axis.

### Syntax

```
int HIMC_EnablePT(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_EnablePT |
| LabVIEW | HIMC Enable PT.vi |
| Python | EnablePT |

# 10.3 HIMC_DisablePT

## Purpose

To disable the position trigger function of an axis.

## Syntax

```
int HIMC_DisablePT(
    int ctrl_id,
    int axis_id
);
```

## Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_DisablePT |
| LabVIEW | HIMC Disable PT.vi |
| Python | DisablePT |

## 10.4 HIMC_IsPTEnabled

### Purpose

To query whether the position trigger function is enabled.

### Syntax

```
int HIMC_IsPTEnabled(
    int ctrl_id,
    int axis_id,
    int *p_is_pt_enabled
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_pt_enabled [out] | A pointer to the buffer to receive the position trigger enabled state.<br>If the axis is at the “PTEnabled” state, the value will be 1. Otherwise, it will be 0. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsPTEnabled |
| LabVIEW | HIMC Is PT Enabled.vi |
| Python | IsPTEnabled |

## 10.5 HIMC_SetPosTriggerConfig

### Purpose

To set the position trigger configuration to an axis.

### Syntax

```
int HIMC_SetPosTriggerConfig(
    int ctrl_id,
    int axis_id,
    PosTriggerPar *pos_trigger_par
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| pos_trigger_par [in] | A pointer to the buffer which contains position trigger configuration set to an axis. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetPosTriggerConfig |
| LabVIEW | HIMC Set Pos Trigger Config.vi |
| Python | SetPosTriggerConfig |

# 11. Touch Probe functions


11. Touch Probe functions ..... 11-1
11.1 Overview ..... 11-2
11.2 HIMC_EnableTouchProbe ..... 11-3
11.3 HIMC_DisableTouchProbe ..... 11-4
11.4 HIMC_IsTouchProbeEnabled ..... 11-5
11.5 HIMC_IsTouchProbeTriggered ..... 11-6
11.6 HIMC_GetTouchProbePos ..... 11-7
11.7 HIMC_SetTouchProbeFunc ..... 11-8

# 11.1 Overview

Touch probe function is a latch function, used in homing procedure (applicable to AC motor, direct drive motor and linear motor). It captures encoder's position feedback value with the edge triggering of encoder input signal. As Figure 11.1.1 shows, when the servo drive passes by its encoder index signal, touch probe function will be triggered, and the position of the index signal will be recorded. With touch probe function, users can query the controller whether touch probe function is triggered, and the controller can get the position of the index signal recorded by latch.

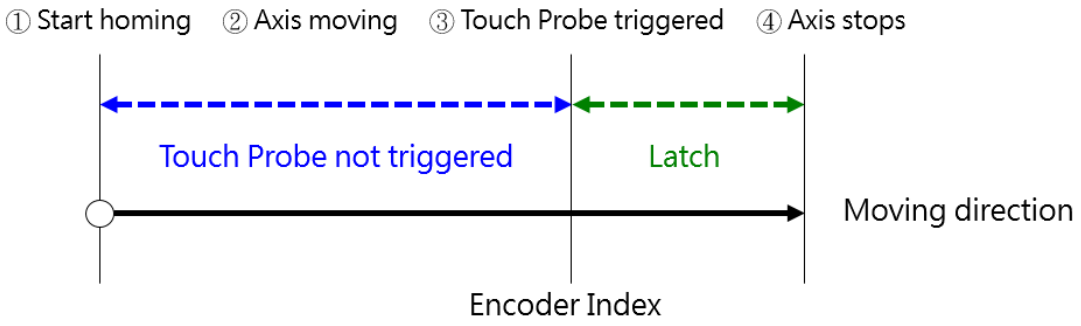



Figure 11.1.1

## 11.2 HIMC_EnableTouchProbe

### Purpose

To enable the touch probe function of an axis.

### Syntax

```
int HIMC_EnableTouchProbe(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure object 0x60B8(Touch probe function) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_EnableTouchProbe |
| LabVIEW | HIMC Enable Touch Probe.vi |
| Python | EnableTouchProbe |

## 11.3 HIMC_DisableTouchProbe

### Purpose

To disable the touch probe function of an axis.

### Syntax

```
int HIMC_DisableTouchProbe(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Users must configure object 0x60B8(Touch probe function) as PDO when using this function.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_DisableTouchProbe |
| LabVIEW | HIMC Disable Touch Probe.vi |
| Python | DisableTouchProbe |

# 11.4 HIMC_IsTouchProbeEnabled

**Purpose**

To query whether the touch probe function is enabled.

**Syntax**

```
int HIMC_IsTouchProbeEnabled(
    int ctrl_id,
    int axis_id,
    int *p_is_probe_enabled
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_probe_enabled [out] | A pointer to the buffer to receive the touch probe enabled state.<br>If the axis is at the “TouchProbeEnabled” state, the value will be 1.<br>Otherwise, it will be 0. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

Users must configure object 0x60B9(Touch probe status) as PDO when using this function.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_IsTouchProbeEnabled |
| LabVIEW | HIMC Is Touch Probe Enabled.vi |
| Python | IsTouchProbeEnabled |

# 11.5 HIMC_IsTouchProbeTriggered

## Purpose

To query whether the touch probe is triggered.

## Syntax

```
int HIMC_IsTouchProbeTriggered(
    int ctrl_id,
    int axis_id,
    int *p_is_probe_triggered
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_probe_triggered [out] | A pointer to the buffer to receive the touch probe triggered state.<br>If the axis is at the “TouchProbeTriggered” state, the value will be 1.<br>Otherwise, it will be 0. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Remark

Users must configure object 0x60B9(Touch probe status) as PDO when using this function.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_IsTouchProbeTriggered |
| LabVIEW | HIMC Is Touch Probe Triggered.vi |
| Python | IsTouchProbeTriggered |

## 11.6 HIMC_GetTouchProbePos

**Purpose**

To get the touch probe position of an axis.

**Syntax**

```
int HIMC_GetTouchProbePos(
    int    ctrl_id,
    int    axis_id,
    double *p_get_probe_pos
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_get_probe_pos [out] | A pointer to the buffer to receive the touch probe position of an axis.<br>Parameter unit: mm or deg |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

Users must configure the corresponding Touch probe object as PDO when using this function such as 0x60BA(Touch probe 1 positive edge).

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_GetTouchProbePos |
| LabVIEW | HIMC Get Touch Probe Pos.vi |
| Python | GetTouchProbePos |

# 11.7 HIMC_SetTouchProbeFunc

**Purpose**

To set touch probe function.

**Syntax**

```
int HIMC_SetTouchProbeFunc(
    int    ctrl_id,
    int    axis_id,
    int    cont_trigger,
    int    detect_edge
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| tp_source [in] | Touch probe source.<br>Input range: 0 (touch probe1, default), 1 (touch probe 2) |
| tp_source [in] | Touch probe trigger mode.<br>Input range: 0 (single trigger, default), 1 (continuous trigger) |
| tp_source [in] | Touch probe source.<br>Input range: 0 (rising-edge detection, default), 1 (falling-edge detection) |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetTouchProbeFunc |
| LabVIEW | HIMC Set Touch Probe Func.vi |
| Python | SetTouchProbeFunc |

# 12. Dynamic Error Compensation functions

12. Dynamic Error Compensation functions ..... 12-1
12.1 Overview ..... 12-2
12.2 HIMC_EnableComp ..... 12-4
12.3 HIMC_DisableComp ..... 12-5
12.4 HIMC_SetupComp ..... 12-6
12.5 HIMC_SetupComp2D ..... 12-8
12.6 HIMC_SetupComp3D ..... 12-10
12.7 HIMC_GetCompPos ..... 12-12
12.8 HIMC_SetCompAlgType ..... 12-13

# 12.1 Overview

HIMC provides dynamic 1D / 2D / 3D error compensation function. Based on related error mearsurement and calculation results, users can establish an error map and do the setting on HIMC. The setting parameters include the compensated axis, reference axis, interval of map points, base of map points, number of map points and compensation value of each map point. As for the setting of compensation value, the memory space of HIMC User Table will be used and the first ID position of error map will be provided to User Table.

**Note 1: Refer to chapter 9 for the usage and the description of User Table.**

**Note 2: Before enabling dynamic error compensation, axis must complete homing to fix the coordinate positions of compensated axis and reference axis.**

Users can choose the same axis to be both compensated axis and reference axis, or choose multiple different axes to be the reference axes of the compensated axis. For example, the compensated axis is Z axis, and the reference axes are X axis and Y axis. The compensation value of the compensated axis will change according to the moving position of the reference axis. During axis motion, the established error map will calculate the compensation command value between map points with linear interpolation, so that the compensation value can keep continuous. When the position of the axis exceeds the range established by error map, HIMC will take the nearest map point's compensation value to be its compensation command, as Figure 12.1.1 shows.

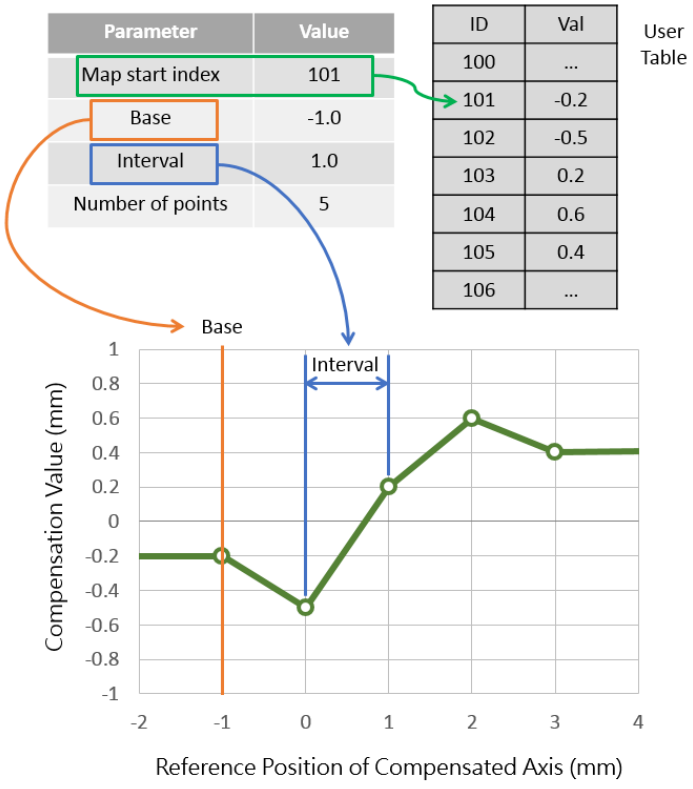



Figure 12.1.1

After enabling dynamic error compensation, in controller's axis control command, the output reference position will add the displacement to be compensated to eliminate the known measured error. As Figure 3.1.1 shows, the relationship can be described as:

Reference position + Position compensation = Position output

After enablilng dynamic error compensation, users can observe variables via Scope Manager in iA Studio.

- Compensation value: Axis → Motion Variable → Position Compensation
- Reference position (without compensation): Axis → Motion Variable → Reference Position
- Reference position (with compensation): Axis → Motion Variable → Position Output

Limit:

In HIMC, compensation command does not go through the profile generator, and the maximum compensation value preset by the controller is 1 mm. If the compensation value is larger than 1 mm, the system will display an error message to remind users.

When enabling dynamic error compensation, the reference coordinate to be compensated must be fixed. Therefore, users cannot change the home offset of the axis.

## 12.2 HIMC_EnableComp

### Purpose

To enable the dynamic error compensation of an axis.

### Syntax

```
int HIMC_EnableComp(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

This function is not applicable when the axis is enabled.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_EnableComp |
| LabVIEW | HIMC Enable Comp.vi |
| Python | EnableComp |

## 12.3 HIMC_DisableComp

### Purpose

To disable the dynamic error compensation of an axis.

### Syntax

```
int HIMC_DisableComp(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

(1) The reference position of the axis will be reset as current feedback.
(2) This function is not applicable when the axis is enabled.
(3) The setting of the dynamic error compensation will be cleared.
To enable the dynamic error compensation again, users need to reset the setting.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_DisableComp |
| LabVIEW | HIMC Disable Comp.vi |
| Python | DisableComp |

## 12.4 HIMC_SetupComp

**Purpose**

To set up one-dimensional dynamic error compensation of an axis.

**Syntax**

```
int HIMC_SetupComp(
    int    ctrl_id,
    int    axis_id,
    int    start_idx,
    double base_val,
    double interval,
    int    num_pt,
    int    ref_axis_id
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| start_idx [in] | Start index of map point in the user table. |
| base_val [in] | Base value (the smallest value of map input)<br>Parameter unit: mm or deg |
| interval [in] | Constant interval between adjacent map points.<br>Parameter unit: mm or deg |
| num_pt [in] | Number of map points. |
| ref_axis_id [in] | Index of reference axis. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetupComp |
| LabVIEW | HIMC Setup Comp.vi |
| Python | SetupComp |

## 12.5 HIMC_SetupComp2D

**Purpose**

To set up two-dimensional dynamic error compensation of an axis.

**Syntax**

```
int HIMC_SetupComp2D(
    int    ctrl_id,
    int    axis_id,
    int    start_idx,
    double *base_val,
    double *interval,
    int    *num_pt,
    int    *ref_axis_id
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| start_idx [in] | Start index of map point in the user table. |
| base_val [in] | A pointer to a two-element array which contains each dimension's base value (the smallest value of map input).<br>Parameter unit: mm or deg |
| interval [in] | A pointer to a two-element array which contains each dimension's constant interval between adjacent map points.<br>Parameter unit: mm or deg |
| num_pt [in] | A pointer to a two-element array which contains each dimension's number of map points. |
| ref_axis_id [in] | A pointer to a two-element array which contains each dimension's index of reference axis. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetupComp2D |
| LabVIEW | HIMC Setup Comp2D.vi |
| Python | SetupComp2D |

## 12.6 HIMC_SetupComp3D

**Purpose**

To set up three-dimensional dynamic error compensation of an axis.

**Syntax**

```
int HIMC_SetupComp3D(
    int    ctrl_id,
    int    axis_id,
    int    start_idx,
    double *base_val,
    double *interval,
    int    *num_pt,
    int    *ref_axis_id
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| start_idx [in] | Start index of map point in the user table. |
| base_val [in] | A pointer to a three-element array which contains each dimension's base value (the smallest value of map input).<br>Parameter unit: mm or deg |
| interval [in] | A pointer to a three-element array which contains each dimension's constant interval between adjacent map points.<br>Parameter unit: mm or deg |
| num_pt [in] | A pointer to a three-element array which contains each dimension's number of map points. |
| ref_axis_id [in] | A pointer to a three-element array which contains each dimension's index of reference axis. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.4 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetupComp3D |
| LabVIEW | HIMC Setup Comp3D.vi |
| Python | SetupComp3D |

# 12.7 HIMC_GetCompPos

## Purpose

To get the error compensation value of an axis sent from the controller to the servo drive.

## Syntax

```
int HIMC_GetCompPos(
    int     ctrl_id,
    int     group_id,
    double *p_comp_pos
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| p_comp_pos [out] | A pointer to the buffer to receive the error compensation value of an axis. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.3 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetCompPos |
| LabVIEW | HIMC Get Comp Pos.vi |
| Python | GetCompPos |

# 12.8 HIMC_SetCompAlgType

## Purpose

To set the interpolation method of dynamic error compensation of an axis.

## Syntax

```
int HIMC_SetCompAlgType(
    int ctrl_id,
    int axis_id,
    int alg_type
);
```

## Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

alg_type [in] The interpolation method of dynamic error compensation.
0: first-order linear interpolation (defualt)
1: three-order spline interpolation

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Remark

Three-dimensional dynamic error compensation does not support three-order spline interpolation.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetCompAlgType |
| LabVIEW | HIMC Set Comp Alg Type.vi |
| Python | SetCompAlgType |

(This page is intentionally left blank.)

# 13. Filter functions

13. Filter functions ..... 13-1
13.1 Overview ..... 13-2
13.2 HIMC_EnableAxisVsf ..... 13-3
13.3 HIMC_DisableAxisVsf ..... 13-4
13.4 HIMC_SetAxisVsf ..... 13-5
13.5 HIMC_EnableAxisInShape ..... 13-7
13.6 HIMC_DisableAxisInShape ..... 13-8
13.7 HIMC_SetAxisInShape ..... 13-9
13.8 HIMC_EnableGrpInShape ..... 13-11
13.9 HIMC_DisableGrpInShape ..... 13-12
13.10 HIMC_SetGrpInShape ..... 13-13

# 13.1 Overview

Filter functions are used to revise profile generator's position command. Currently, HMPL provides three kinds of filters, smooth time, vibration suppression filter (VSF), and input shaping filter (InShape).

Smooth time makes the motor accelerate smoothly to achieve smooth movement, while VSF and InShape suppresses the vibration of the motor (especially when the load of the mechanism is cantilever) during the movement. By tuning "frequency" and "damping ratio", the effect of vibration suppression can be achieved.

VSF and InShape cannot be used at the same time, but either of them can be used with smooth time.

Besides, when it comes to coordinated motion, Axis InShape function is useless. Users must adopt Group InShape function to suppress the vibration.

**Note: Using filters will increase move time and decrease debounce time.**

## 13.2 HIMC_EnableAxisVsf

**Purpose**

To enable VSF filter of an axis.

**Syntax**

```
int HIMC_EnableAxisVsf(
    int ctrl_id,
    int axis_id
);
```

**Parameter**

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

This function is not applicable when the motor is moving. Otherwise, the motor will generate an unexpected vibration.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_EnableAxisVsf |
| LabVIEW | HIMC Enable Axis Vsf.vi |
| Python | EnableAxisVsf |

## 13.3 HIMC_DisableAxisVsf

### Purpose

To disable VSF filter of an axis.

### Syntax

```
int HIMC_DisableAxisVsf(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_DisableAxisVsf |
| LabVIEW | HIMC Disable Axis Vsf.vi |
| Python | DisableAxisVsf |

## 13.4 HIMC_SetAxisVsf

**Purpose**

To set VSF filter's parameters of an axis.

**Syntax**

```
int HIMC_SetAxisVsf(
    int ctrl_id,
    int axis_id,
    double frequency,
    double damping_ratio
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| frequency [in] | System frequency.<br>Parameter unit: Hz<br>Input range: 0.1 ~ 200 |
| damping_ratio [in] | Damping ratio.<br>Input range: 0.7 ~ 1.5 (**1.0** is recommended) |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

This function is not applicable when the motor is moving. Otherwise, the motor will generate an unexpected vibration.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetAxisVsf |
| LabVIEW | HIMC Set Axis Vsf.vi |
| Python | SetAxisVsf |

## 13.5 HIMC_EnableAxisInShape

**Purpose**

To enable InShape filter of an axis.

**Syntax**

```
int HIMC_EnableAxisInShape(
    int ctrl_id,
    int axis_id
);
```

**Parameter**

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

This function is not applicable when the motor is moving. Otherwise, the motor will generate an unexpected vibration.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_EnableAxisInshape |
| LabVIEW | HIMC Enable Axis In Shape.vi |
| Python | EnableAxisInShape |

## 13.6 HIMC_DisableAxisInShape

### Purpose

To disable InShape filter of an axis.

### Syntax

```
int HIMC_DisableAxisInShape(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_DisableAxisInshape |
| LabVIEW | HIMC Disable Axis In Shape.vi |
| Python | DisableAxisInShape |

# 13.7 HIMC_SetAxisInShape

**Purpose**

To set InShape filter's parameters of an axis.

**Syntax**

```
int HIMC_SetAxisInShape(
    int ctrl_id,
    int axis_id,
    double frequency,
    double damping_ratio,
    ShaperMode shaper_type
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl |
| axis_id [in] | Axis index. |
| frequency [in] | System frequency.<br>Parameter unit: Hz<br>Input range: 3.0 ~ 300 |
| damping_ratio [in] | Damping ratio.<br>Input range: 0.0 ~ 0.3 |
| shaper_type [in] | There are two shaper types, **Shaper_Normal** and **Shaper_Robust**.<br>Shaper_Robust is more robust than Shaper_Normal, but Shaper_Normal is strong enough to suppress vibration. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

(1) This function is not applicable when the motor is moving. Otherwise, the motor will generate an unexpected vibration.

(2) The default value for frequency and damping ratio is 5.5Hz and 0.03 respectively.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetAxisInshape |
| LabVIEW | HIMC Set Axis In Shape.vi |
| Python | SetAxisInShape |

## 13.8 HIMC_EnableGrpInShape

### Purpose

To enable InShape filter of an axis group.

### Syntax

```
int HIMC_EnableGrpInShape(
    int ctrl_id,
    int group_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl

group_id [in] Axis group index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

This function is not applicable when the motor is moving. Otherwise, the motor will generate an unexpected vibration.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_EnableGrpInShape |
| LabVIEW | HIMC Enable Grp In Shape.vi |
| Python | EnableGrpInShape |

# 13.9 HIMC_DisableGrpInShape

## Purpose

To disable InShape filter of an axis group.

## Syntax

```
int HIMC_DisableGrpInShape(
    int ctrl_id,
    int group_id
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl |
| group_id [in] | Axis group index. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_DisableGrpInShape |
| LabVIEW | HIMC Disable Grp In Shape.vi |
| Python | DisableGrpInShape |

# 13.10 HIMC_SetGrpInShape

**Purpose**

To set InShape filter's parameters of an axis group.

**Syntax**

```
int HIMC_SetGrpInShape(
    int ctrl_id,
    int group_id,
    double frequency,
    double damping_ratio,
    ShaperMode shaper_type
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl |
| group_id [in] | Axis group index. |
| frequency [in] | System frequency.<br>Parameter unit: Hz<br>Input range: 3.0 ~ 300 |
| damping_ratio [in] | Damping ratio.<br>Input range: 0.0 ~ 0.3 |
| shaper_type [in] | There are two shaper types, **Shaper_Normal** and **Shaper_Robust**.<br>Shaper_Robust is more robust than Shaper_Normal, but Shaper_Normal is strong enough to suppress vibration. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

(1) This function is not applicable when the motor is moving. Otherwise, the motor will generate an unexpected vibration.

(2) The default value for frequency and damping ratio is 5.5Hz and 0.03 respectively.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetGrpInShape |
| LabVIEW | HIMC Set Grp In Shape.vi |
| Python | SetGrpInShape |

# 14. HMPL Task functions


14. HMPL Task functions ..... 14-1
14.1 Overview ..... 14-2
14.2 HIMC_StartTask ..... 14-3
14.3 HIMC_StartTaskFunc ..... 14-4
14.4 HIMC_StopTask ..... 14-5
14.5 HIMC_StopAllTask ..... 14-6
14.6 HIMC_IsTaskStop ..... 14-7
14.7 HIMC_LoadHMPLTask ..... 14-8

## 14.1 Overview

HIMC has 64 built-in HMPL tasks for users to practice motion profile commands based on application. In any HMPL task, users can start or stop other HMPL tasks with HMPL task function. When a HMPL task is being executed, users cannot ask it to be re-executed; instead, users should wait until the execution of the task is done and the task enters “stop” status. However, users can query whether the HMPL task is currently being executed, and control the order of multiple HMPL tasks for the application accordingly.

## 14.2 HIMC_StartTask

### Purpose

To start the execution of a HMPL task.

### Syntax

```
int HIMC_StartTask(
    int ctrl_id,
    int task_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

task_id [in] HMPL task ID.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_StartTask |
| LabVIEW | HIMC Start Task.vi |
| Python | StartTask |

## 14.3 HIMC_StartTaskFunc

### Purpose

To start the execution of a function in a HMPL task.

### Syntax

```
int HIMC_StartTaskFunc(
    int ctrl_id,
    int task_id,
    char *func_name
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

task_id [in] HMPL task ID.

func_name [in] A pointer to the buffer to store the function name in the HMPL task.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_StartTaskFunc |
| LabVIEW | HIMC Start Task Func.vi |
| Python | StartTaskFunc |

# 14.4 HIMC_StopTask

## Purpose

To stop the execution of a HMPL task.

## Syntax

```
int HIMC_StopTask(
    int ctrl_id,
    int task_id
);
```

## Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

task_id [in] HMPL task ID.

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_StopTask |
| LabVIEW | HIMC Stop Task.vi |
| Python | StopTask |

# 14.5 HIMC_StopAllTask

## Purpose

To stop the execution of all HMPL tasks (the caller included).

## Syntax

```
int HIMC_StopAllTask(
    int ctrl_id
);
```

## Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_StopAllTask |
| LabVIEW | HIMC Stop All Task.vi |
| Python | StopAllTask |

# 14.6 HIMC_IsTaskStop

**Purpose**

To query whether the execution of a HMPL task is stopped.

**Syntax**

```
int HIMC_IsTaskStop(
    int ctrl_id,
    int task_id,
    int *isStop
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| task_id [in] | HMPL task ID. |
| isStop [out] | A pointer to the buffer to receive the status of the HMPL task.<br>If the HMPL task is at the “TaskStop” state, the value will be 1. Otherwise, it will be 0. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_IsTaskStop |
| LabVIEW | HIMC Is Task Stop.vi |
| Python | IsTaskStop |

# 14.7 HIMC_LoadHMPLTask

**Purpose**

To load HMPL file to the controller.

**Syntax**

```
int HIMC_LoadHMPLTask(
    int ctrl_id,
    int task_id,
    const char *file_name
);
```

**Parameter**

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

task_id [in] HMPL task ID.

file_name [in] A pointer to the buffer to store the HMPL file path to be loaded.

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

This function must be used with the executable file “HMPL_compiler.exe” in “iA_Studio” folder. Copy the executable file and place it in the “bin\Debug” and “bin\Release” folder of the user-developed program.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Executable | HMPL_compiler.exe |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_LoadHMPLTask |
| LabVIEW | HIMC Load HMPL Task.vi |
| Python | LoadHMPLTask |

# 15. Callback functions

15. Callback functions ..... 15-1
15.1 HIMC_SetHmplEvtCallback ..... 15-2
15.2 HIMC_SetErrorCallback ..... 15-3
15.3 HIMC_SetHmplMsgEvtCallback ..... 15-4

# 15.1 HIMC_SetHmplEvtCallback

## Purpose

To register a callback function to capture the event sent by HMPL task.

## Syntax

```
int HIMC_SetHmplEvtCallback(
    int ctrl_id,
    HMPLEventCBFuncPtr hmpl_event_cb_func_ptr
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| hmpl_event_cb_func_ptr [in] | A pointer to callback function.<br>The prototype of the function is “void func(int arg)”. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetHmplEvtCallback |
| LabVIEW | -- |
| Python | SetHmplEvtCallback |

## 15.2 HIMC_SetErrorCallback

### Purpose

To register a callback function to capture the error ID sent by the controller.

### Syntax

```
int HIMC_SetErrorCallback(
    int ctrl_id,
    HimcErrorCBFuncPtr himc_error_cb_func_ptr
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| himc_error_cb_func_ptr [in] | A pointer to callback function.<br>The prototype of the function is “void func(int arg)”. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

Refer ro chapter 5 in “iA Studio User Guide” for error ID.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetErrorCallback |
| LabVIEW | -- |
| Python | SetErrorCallback |

## 15.3 HIMC_SetHmplMsgEvtCallback

### Purpose

To register a callback function to capture the string message sent by HMPL task.

### Syntax

```
int HIMC_SetHmplMsgEvtCallback(
    int ctrl_id,
    HMPLMsgEventCBFuncPtr hmpl_msg_cb_func_ptr
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| hmpl_msg_cb_func_ptr [in] | A pointer to callback function.<br>The prototype of the function is “void func(const char *text)”. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Remark

The maximum length of the string message is 128 characters.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetHmplMsgEvtCallback |
| LabVIEW | -- |
| Python | SetHmplMsgEvtCallback |

# 16. Variable and Function Operation functions


16. Variable and Function Operation functions ..... 16-1
16.1 Overview ..... 16-2
16.1.1 Controller variables list ..... 16-3
16.2 Servo drive variable operation ..... 16-7
16.2.1 HIMC_ReadSDO ..... 16-7
16.2.2 HIMC_WriteSDO ..... 16-9
16.2.3 HIMC_ReadPDO ..... 16-10
16.2.4 HIMC_WritePDO ..... 16-11
16.2.5 HIMC_ForceWritePDO ..... 16-12
16.2.6 HIMC_ReleasePDO ..... 16-13
16.3 Controller variable operation ..... 16-14
16.3.1 HIMC_GetVariableByID ..... 16-14
16.3.2 HIMC_SetVariableByID ..... 16-15
16.3.3 HIMC_GetVariableListByID ..... 16-16
16.3.4 HIMC_SetVariableListByID ..... 16-17
16.3.5 HIMC_GetGlobalVariables ..... 16-18
16.3.6 HIMC_SetGlobalVariables ..... 16-19

## 16.1 Overview

HIMC provides users with variable operation functions for servo drive and controller. For servo drive's variable operation, servo drive Object Dictionary index must be given to access its value, and data exchange must be performed via CoE communication. As for controller's variable operation, specific variable's addressing ID must be given based on controller's variable ID list for the access. The definition of parameters of controller is given in section 16.1.1.

Attention:
If there is no requirement for specific purposes, it is recommended for users to access relevant system variables with relevant HMI and functions. When using variable operation functions, users should ensure the safety of accessing variables and entering values.

## 16.1.1 Controller variables list

HIMC takes 32 bits as controller variable's addressing ID. Its type is 0x□□□□□□□□, where "0x" indicates the value is in hexadecimal system. With variable operation functions, users can access system variables, axis variables and axis group variables provided by HIMC. The rule of addressing ID is explained as follows:

1. The 1st and 2nd value of addressing ID indicates "category of controller variable". 0x00□□□□□□ belongs to system variable; 0x83□□□□□□ belongs to axis variable; 0x82□□□□□□ belongs to axis group variable.

2. The 3rd and 4th value of addressing ID indicates "axis ID or axis group ID". For example, axis variable 0x8302□□□□ is a variable storing axis index 02, while axis group variable 0x8201□□□□ is a variable storing axis group index 01.

3. The 5th to 8th value of addressing ID indicates "addressing location of controller's system, axis or axis group variable". Refer to Table 16.1.1.1 to Table 16.1.1.3 for parameters list and description.

Table 16.1.1.1

| System Variables | | |
|---|---|---|
| Addressing ID | Variable Name | Description |
| 0x0000012c | HCV_ID_fclk | System execution clock (increase 1 count per 250 us) |
| 0x0000012e | HCV_ID_timeInMs | System execution time (ms) |
| 0x000007d0 | HCV_ID_user_table | double[512000] array variable that users can freely use |
| 0x00002328 | HCV_ID_ltest0 | int variable that users can freely use |
| 0x00002329 | HCV_ID_ltest1 | int variable that users can freely use |
| 0x0000232a | HCV_ID_ltest2 | int variable that users can freely use |
| 0x0000232b | HCV_ID_ltest3 | int variable that users can freely use |
| 0x0000232c | HCV_ID_ltest4 | int variable that users can freely use |
| 0x0000232d | HCV_ID_ltest5 | int variable that users can freely use |
| 0x0000232e | HCV_ID_ltest6 | int variable that users can freely use |
| 0x0000232f | HCV_ID_ltest7 | int variable that users can freely use |
| 0x00002330 | HCV_ID_ltest8 | int variable that users can freely use |
| 0x00002331 | HCV_ID_ltest9 | int variable that users can freely use |
| 0x0000235a | HCV_ID_dtest0 | double variable that users can freely use |
| 0x0000235b | HCV_ID_dtest1 | double variable that users can freely use |
| 0x0000235c | HCV_ID_dtest2 | double variable that users can freely use |
| 0x0000235d | HCV_ID_dtest3 | double variable that users can freely use |
| 0x0000235e | HCV_ID_dtest4 | double variable that users can freely use |
| 0x0000235f | HCV_ID_dtest5 | double variable that users can freely use |
| 0x00002360 | HCV_ID_dtest6 | double variable that users can freely use |
| 0x00002361 | HCV_ID_dtest7 | double variable that users can freely use |

| System Variables | | |
|---|---|---|
| Addressing ID | Variable Name | Description |
| 0x00002362 | HCV_ID_dtest8 | double variable that users can freely use |
| 0x00002363 | HCV_ID_dtest9 | double variable that users can freely use |
| 0x0000238c | HCV_ID_mtest | double[10] array variable that users can freely use |

Table 16.1.1.2

| Axis Variables | | |
|---|---|---|
| Addressing ID | Variable Name | Description |
| 0x83□□0015 | HCV_ID_motion_type | Motion type |
| 0x83□□0033 | HCV_ID_pos_tr | In-position convergence radius |
| 0x83□□0034 | HCV_ID_pos_tr_t | In-position settling time |
| 0x83□□0065 | HCV_ID_sw_RL | Software right limit |
| 0x83□□0066 | HCV_ID_sw_LL | Software left limit |
| 0x83□□0067 | HCV_ID_vel_lim | Maximum velocity limit |
| 0x83□□0068 | HCV_ID_acc_lim | Maximum acceleration limit |
| 0x83□□0069 | HCV_ID_dec_lim | Maximum deceleration limit |
| 0x83□□0079 | HCV_ID_max_pos_err | Position error limit |
| 0x83□□007a | HCV_ID_max_comp_lim | Position compensation limit |
| 0x83□□00a0 | HCV_ID_home_status | Homing state |
| 0x83□□00a1 | HCV_ID_home_method | Homing method |
| 0x83□□00a2 | HCV_ID_home_fast_vel | Fast homing velocity |
| 0x83□□00a3 | HCV_ID_home_slow_vel | Slow homing velocity |
| 0x83□□00a4 | HCV_ID_home_timeout | Homing delay time |
| 0x83□□00a5 | HCV_ID_home_acc | Homing acceleration |
| 0x83□□00a6 | HCV_ID_home_offset | Homing position offset |
| 0x83□□00d3 | HCV_ID_max_vel | Target velocity |
| 0x83□□00d4 | HCV_ID_max_acc | Target acceleration |
| 0x83□□00d5 | HCV_ID_max_dec | Target deceleration |
| 0x83□□00d7 | HCV_ID_sm_factor | Smooth time |
| 0x83□□00db | HCV_ID_vel_scale | Velocity scale (0~100) |
| 0x83□□00dd | HCV_ID_p2p_del | P2P motion waiting time |
| 0x83□□00de | HCV_ID_p2p_pos1 | P2P position 1 |
| 0x83□□00df | HCV_ID_p2p_pos2 | P2P position 2 |
| 0x83□□00e0 | HCV_ID_p2p_repeat | Repeat P2P motion |
| 0x83□□00e1 | HCV_ID_rlt_dist | Relative move distance |
| 0x83□□00e2 | HCV_ID_en_motionManager | Motion axis selection of Motion Manager |
| 0x83□□00e3 | HCV_ID_acc_time | Acceleration time |
| 0x83□□00e4 | HCV_ID_dec_time | Deceleration time |
| 0x83□□00e9 | HCV_ID_map_io_type | Error compensation type |

| Axis Variables | | |
|---|---|---|
| Addressing ID | Variable Name | Description |
| 0x83□□0117 | HCV_ID_rollover_turns | Rollover turns |
| 0x83□□0119 | HCV_ID_rollover_val | Rollover value |
| 0x83□□0193 | HCV_ID_gant_pair | Gantry pair ID of gantry configuration |
| 0x83□□01f7 | HCV_ID_en_delay | Time out for axis enabling |
| 0x83□□01ff | HCV_ID_fb_ratio_pos | Servo drive position resolution; length unit (denominator) |
| 0x83□□0200 | HCV_ID_fb_ratio_cnt | Servo drive position resolution; unit: count (numerator) |
| 0x83□□0209 | HCV_ID_fb_curr_ratio_curr | Servo drive current resolution; current unit (denominator) |
| 0x83□□020a | HCV_ID_fb_curr_ratio_cnt | Servo drive current resolution; unit: count (numerator) |
| 0x83□□0213 | HCV_ID_rotor_inertia | Rotor intertia ratio of the motor |
| 0x83□□0214 | HCV_ID_force_constant | Torque constant of the motor |
| 0x83□□0263 | HCV_ID_last_err | Axis error code |
| 0x83□□03c2 | HCV_ID_gear_ratio | Gear ratio |

**Note: Symbols □□ will be the axis ID in hexadecimal format. For example, 01 stands for axis index 01; 0f stands for axis index 15.**

Table 16.1.1.3

| Axis Group Variables | | |
|---|---|---|
| Addressing ID | Variable Name | Description |
| 0x82□□0002 | HCV_ID_grp_num_axis | Number of axes for axis group |
| 0x82□□00c9 | HCV_ID_grp_lin_vel_lim | Velocity limit for axis group’s linear motion |
| 0x82□□00ca | HCV_ID_grp_lin_acc_lim | Acceleration limit for axis group’s linear motion |
| 0x82□□00cb | HCV_ID_grp_lin_dec_lim | Deceleration limit for axis group’s linear motion |
| 0x82□□00d4 | HCV_ID_grp_ang_vel_lim | Velocity limit for axis group’s rotary motion |
| 0x82□□00d5 | HCV_ID_grp_ang_acc_lim | Acceleration limit for axis group’s rotary motion |
| 0x82□□00d6 | HCV_ID_grp_ang_dec_lim | Deceleration limit for axis group’s rotary motion |
| 0x82□□00d0 | HCV_ID_grp_lin_vel | Target velocity for axis group |
| 0x82□□00d1 | HCV_ID_grp_lin_acc | Target acceleration for axis group |
| 0x82□□00d2 | HCV_ID_grp_lin_dec | Target deceleration for axis group |
| 0x82□□00d3 | HCV_ID_grp_lin_sf | Smooth time for axis group |
| 0x82□□00f0 | HCV_ID_grp_lin_acc_time | Target acceleration time for axis group |
| 0x82□□00f1 | HCV_ID_grp_lin_dec_time | Target deceleration time for axis group |
| 0x82□□00e7 | HCV_ID_grp_ang_vel | Target velocity for axis group’s rotary motion |
| 0x82□□00e8 | HCV_ID_grp_ang_acc | Target acceleration for axis group’s rotary motion |
| 0x82□□00e9 | HCV_ID_grp_ang_dec | Target deceleration for axis group’s rotary motion |
| 0x82□□00ea | HCV_ID_grp_ang_sf | Smooth time for axis group’s rotary motion |
| 0x82□□00eb | HCV_ID_grp_ang_acc_time | Target acceleration time for axis group’s rotary motion |

| Axis Group Variables | | |
|---|---|---|
| Addressing ID | Variable Name | Description |
| 0x82□□00ec | HCV_ID_grp_ang_dec_time | Target deceleration time for axis group's rotary motion |
| 0x82□□00e4 | HCV_ID_grp_coord_sys | Coordinate system for axis group |
| 0x82□□00e5 | HCV_ID_grp_buffer_mode | Buffer mode for axis group |
| 0x82□□00e6 | HCV_ID_grp_trans_mode | Transition mode for axis group |
| 0x82□□00e7 | HCV_ID_grp_trans_vel | Transition velocity for axis group |
| 0x82□□00e8 | HCV_ID_grp_trans_dis | Tramsition distance for axis group |
| 0x82□□00f6 | HCV_ID_grp_trans_dev | Tramsition deviation for axis group |
| 0x82□□00f7 | HCV_ID_grp_trans_curvature | Tramsition curvature for axis group |
| 0x82□□00eb | HCV_ID_grp_vel_scale | Velocity scale (0~100) for axis group |
| 0x82□□00da | HCV_ID_grp_shaper_fr | InShape filter's frequency for axis group |
| 0x82□□00db | HCV_ID_grp_shaper_xi | InShape filter's damping ratio for axis group |
| 0x82□□038f | HCV_ID_grp_last_err | Axis group error code |

**Note: Symbols □□ will be the axis group ID in hexadecimal format. For example, 01 stands for axis group index 01; 0f stands for axis group index 15.**

# 16.2 Servo drive variable operation

## 16.2.1 HIMC_ReadSDO

**Purpose**

To read the object value of the slave through SDO.

**Syntax**

```
int HIMC_ReadSDO(
    int ctrl_id,
    int slv_id,
    int obj_index,
    int obj_subindex,
    int size_bytes,
    long long *value
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_id [in] | Slave index. |
| obj_index [in] | Index of the slave object. |
| obj_subindex [in] | Subindex of the slave object. |
| obj_subindex [in] | Byte length of the slave object. |
| value [out] | A pointer to the buffer to store the read object value. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_ReadSDO |

| | |
|---|---|
| LabVIEW | HIMC Read SDO.vi |
| Python | ReadSDO |

## 16.2.2 HIMC_WriteSDO

### Purpose

To write the object value of the slave through SDO.

### Syntax

```
int HIMC_WriteSDO(
    int ctrl_id,
    int slv_id,
    int obj_index,
    int obj_subindex,
    int size_bytes,
    long long value
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_id [in] | Slave index. |
| obj_index [in] | Index of the slave object. |
| obj_subindex [in] | Subindex of the slave object. |
| size_bytes [in] | Byte length of the slave object. |
| value [in] | The object value to be written. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_Write SDO.vi |
| LabVIEW | HIMC Write SDO.vi |
| Python | WriteSDO |

## 16.2.3 HIMC_ReadPDO

### Purpose

To read PDO object value of the slave through PDO.

### Syntax

```
int HIMC_ReadPDO(
    int ctrl_id,
    int slv_id,
    int obj_index,
    int obj_subindex,
    long long *value
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_id [in] | Slave index. |
| obj_index [in] | Index of the slave object. |
| obj_subindex [in] | Subindex of the slave object. |
| value [out] | A pointer to the buffer to store the read object value. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ReadPDO |
| LabVIEW | HIMC Read PDO.vis |
| Python | ReadPDO |

## 16.2.4 HIMC_WritePDO

### Purpose

To write PDO object value of the slave through PDO.

### Syntax

```
int HIMC_WritePDO(
    int ctrl_id,
    int slv_id,
    int obj_index,
    int obj_subindex,
    long long value
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_id [in] | Slave index. |
| obj_index [in] | Index of the slave object. |
| obj_subindex [in] | Subindex of the slave object. |
| value [in] | The object value to be written. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_WritePDO |
| LabVIEW | HIMC Write PDO.vi |
| Python | WritePDO |

## 16.2.5 HIMC_ForceWritePDO

### Purpose

To force write PDO object of the slave through PDO.

### Syntax

```
int HIMC_ForceWritePDO(
    int ctrl_id,
    int slv_id,
    int obj_index,
    int obj_subindex,
    long long value
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_id [in] | Slave index. |
| obj_index [in] | Index of the slave object. |
| obj_subindex [in] | Subindex of the slave object. |
| value [in] | The object value to be written. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ForceWritePDO |
| LabVIEW | HIMC ForceWrite PDO.vi |
| Python | ForceWritePDO |

## 16.2.6 HIMC_ReleasePDO

### Purpose

To release force written PDO object and use it with ForceWritePDO.

### Syntax

```
int HIMC_ReleasePDO(
    int ctrl_id,
    int slv_id,
    int obj_index,
    int obj_subindex
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| slv_id [in] | Slave index. |
| obj_index [in] | Index of the slave object. |
| obj_subindex [in] | Subindex of the slave object. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ReleasePDO |
| LabVIEW | HIMC Release PDO.vi |
| Python | ReleasePDO |

# 16.3 Controller variable operation

## 16.3.1 HIMC_GetVariableByID

### Purpose

To get the variable value of the controller by ID.

### Syntax

```
int HIMC_GetVariableByID(
    int    ctrl_id,
    int    var_id,
    double *p_val
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

var_id [in] HIMC controller variable ID.
**Refer to section 16.1.1 for the definition.**

p_val [out] A pointer to the buffer to receive the value of the variable.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetVariableByID |
| LabVIEW | HIMC Get Variable By ID.vi |
| Python | GetVariableByID |

## 16.3.2 HIMC_SetVariableByID

### Purpose

To set the variable value of the controller by ID.

### Syntax

```
int HIMC_SetVariableByID(
    int    ctrl_id,
    int    var_id,
    double val
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

var_id [in] HIMC controller variable ID.
**Refer to section 16.1.1 for the definition.**

val [in] The new value of the variable.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetVariableByID |
| LabVIEW | HIMC Set Variable By ID.vi |
| Python | SetVariableByID |

## 16.3.3 HIMC_GetVariableListByID

### Purpose

To get values of list variables of the controller by ID.

### Syntax

```
int HIMC_GetVariableListByID(
    int    ctrl_id,
    int    *p_var_id,
    int    num,
    double *p_val
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

p_var_id [in] A pointer to the buffer to store the HIMC controller variable IDs of list variables.
**Refer to section 16.1.1 for the definition.**

num [in] Number of variables.

p_val [out] A pointer to the buffer to receive the values of list variables.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetVariableListByID |
| LabVIEW | HIMC Get Variable List By ID.vi |
| Python | GetVariableListByID |

## 16.3.4 HIMC_SetVariableListByID

### Purpose

To set values to list variables of the controller by ID.

### Syntax

```
int HIMC_SetVariableListByID(
    int    ctrl_id,
    int    *p_var_id,
    int    num,
    double *p_val
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

p_var_id [in] A pointer to the buffer to store the HIMC controller variable IDs of list variables.
**Refer to section 16.1.1 for the definition.**

num [in] Number of variables.

p_val [in] A pointer to the buffer to store the new values of list variables.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetVariableListByID |
| LabVIEW | HIMC Set Variable List By ID.vi |
| Python | SetVariableListByID |

## 16.3.5 HIMC_GetGlobalVariables

### Purpose

To get values of list global variables of the controller.

### Syntax

```
int HIMC_GetGlobalVariables(
    int    ctrl_id,
    char   **pp_var_name_array,
    int    length,
    double *p_output_array
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| pp_var_name_array [in] | A pointer to the buffer to store the names of list global variables. |
| length [in] | Number of global variables. |
| p_output_array [out] | A pointer to the buffer to receive the values of list global variables. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Executable | HMPL_compiler.exe |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGlobalVariables |
| LabVIEW | HIMC Get Global Variables.vi |
| Python | GetGlobalVariables |

## 16.3.6 HIMC_SetGlobalVariables

### Purpose

To set values to list global variables of the controller.

### Syntax

```
int HIMC_SetGlobalVariables(
    int    ctrl_id,
    char   **pp_var_name_array,
    int    length,
    double *p_input_array
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| pp_var_name_array [in] | A pointer to the buffer to store the names of list global variables. |
| length [in] | Number of global variables. |
| p_input_array [in] | A pointer to the buffer to store the new values of list global variables. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Executable | HMPL_compiler.exe |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetGlobalVariables |
| LabVIEW | HIMC Get Global Variables.vi |
| Python | GetGlobalVariables |

(This page is intentionally left blank.)

# 17. HIMC Error functions


17. HIMC Error functions ..... 17-1
17.1 Overview ..... 17-2
17.1.1 System error messages ..... 17-3
17.1.2 Axis error massages ..... 17-6
17.1.3 Group error messages ..... 17-9
17.2 HIMC_GetLastError ..... 17-11
17.3 HIMC_GetAxisLastErr ..... 17-12
17.4 HIMC_ClearAxisLastErr ..... 17-13
17.5 HIMC_GetGrpLastErr ..... 17-14
17.6 HIMC_ClearGrpLastErr ..... 17-15
17.7 HIMC_GetDriveErr ..... 17-16
17.8 HIMC_GetErrorInformation ..... 17-17

# 17.1 Overview

HIMC offers 32-bit error codes to represent the related error messages. With the functions provided in this chapter, users can get or clear the error code of system, axis and axis group. (The error codes, names and descriptions of each category are listed in section 17.1.1 to 17.1.3.) The type of error code is 0x□□□□□□□□, where “0x” indicates the value is in hexadecimal system. Its rule is the same as that of controller variable addressing ID, which is explained as follows:

1. The 1st and 2nd value of error code indicates “category of controller variable”. 0x00□□□□□□ belongs to system variable; 0x83□□□□□□ belongs to axis variable; 0x82□□□□□□ belongs to axis group variable.
2. The 3rd and 4th value of error code indicates “axis ID or axis group ID”. For example, axis variable 0x8302□□□□ is a variable storing axis index 02, while axis group variable 0x8201□□□□ is a variable storing axis group index 01.
3. The 5th to 8th value of error code indicates “variable ID”. Refer to section 17.1.1 to 17.1.3 for details.

**Note: Since the return value of function is in decimal system, users must convert it to hexadecimal system by themselves to get the correct error code.**

## 17.1.1 System error messages

Table 17.1.1.1

| System Error Codes | | |
|---|---|---|
| Error Code | Error Name | Description |
| 0x00000001 | eERR_HCV_ID_NOT_FOUND | The variable ID was not found. |
| 0x00000002 | eERR_DATA_EXCEEDED | The requested data is out of range. |
| 0x00000003 | eERR_HCV_IS_READ_ONLY | Read-only parameter. |
| 0x00000004 | eERR_HCV_VALUE_OUT_OF_RANGE | The input value is out of range. |
| 0x00000064 | eERR_EMERGENCY_STOP | Emergency stop activated. Disable all axes and stop all tasks. |
| 0x00000100 | eERR_MAIL_BOX_BUSY | The mailbox between controller and slave is busy. |
| 0x00000101 | eERR_VAR_NOT_IN_SLV_DB | The slave variable was not found. |
| 0x00000102 | eERR_VAR_NOT_REGYET | The slave variable cannot be read. |
| 0x00000103 | eERR_READ_VAR_NO_RECV | There was no response from slave. |
| 0x00000104 | eERR_PREV_SLV_CMD_NOT_FIN | The previous command to slave is not finished. |
| 0x00000105 | eERR_SLV_ID_INVALID | The slave ID is invalid. |
| 0x00000106 | eERR_PDO_NUM_EXCEED | The number of PDO is out of range. |
| 0x00000107 | eERR_NOT_VALID_TASKID | The task ID is invalid. |
| 0x00000108 | eERR_TASK_IS_RUNNING | The task is already running. |
| 0x00000109 | eERR_FUNC_NOT_IN_TASK | The function was not found in task. |
| 0x0000010a | eERR_TASK_EMPTY | The task is empty. |
| 0x0000010b | eERR_TASK_NOT_RUNNING | The task is not running. |
| 0x0000012c | eERR_NIC_INIT_TOUT | The network port of mega-ulink is not ready. |
| 0x0000012d | eERR_HARDWARE_MISMATCH | The hardware is unrecognized. |
| 0x0000012e | eERR_SLAVE_NUM_MISMATCH | The number of slaves is different from configuration. |
| 0x0000012f | eERR_INVALID_PDO | The PDO is invalid. |
| 0x00000130 | eERR_INVALID_MCK_CNFG | The configuration of motion kernel is invalid. |
| 0x00000138 | eERR_HIMC_LOAD_CONFIG_FAIL | Load configuration from SSD failed. Please save it again. |
| 0x00000139 | eERR_HIMC_SAVE_CONFIG_FAIL | Store configuration to HIMC failed. Please save it again. |
| 0x0000013a | eERR_HIMC_SAVE_CONFIG_COPY_FAIL | Store configuration to HIMC failed. Cannot save file into SAVE folder. |
| 0x0000013b | eERR_HIMC_SAVE_UPDATE_PRM_TIMEOUT | Store configuration to HIMC failed. Update Prm values timeout. |
| 0x0000013c | eERR_ETHERCAT_LICENSE_MISMATCH | The authorization of EtherCAT is invalid. |
| 0x000001f4 | eERR_ISR_NOT_STABLE | The period of interrupt is not stable. |
| 0x000001f5 | eERR_MCK_OVERLOAD | The motion kernel is overloaded. |
| 0x000001f6 | eERR_ISR_OVERLOAD | The CPU is overloaded. |
| 0x00001388 | eERR_HMPL_INVALID_ARG | The arguments are invalid in HMPL. |
| 0x00001389 | eERR_HMPL_INVALID_PTR | The pointer is invalid in HMPL. |
| 0x0000138a | eERR_HMPL_STACK_OVERFLOW | Stack overflow in HMPL. |
| 0x0000138b | eERR_HMPL_ILLEGAL_MEM_OP | The operation of memory is illegal in HMPL. |
| 0x0000138c | eERR_HMPL_MOTION_NOT_READY | Motion function should be called in synchronized state. |
| 0x0000138d | eERR_HMPL_STR_TOO_LONG | String length is out of range. |

| System Error Codes | | |
|---|---|---|
| Error Code | Error Name | Description |
| 0x0000138e | eERR_HMPL_INVALID_STR_FORMAT | String format is invalid. |
| 0x0000138f | eERR_HMPL_ARG_OUT_OF_RANGE | The argument is out of range. |
| 0x00001392 | eERR_HMPL_ASCII_AGENT_RUNNING | ASCII agent is already running. |
| 0x0000139c | eERR_HMPL_CANNOT_RUN_IN_DEBUG | The function cannot run in debug mode. |
| 0x000013a6 | eERR_HMPL_TOO_MANY_BRK_POINT | There are too many break points in the task. |
| 0x000013ec | eERR_HMPL_MUTEX_LOCK_TWICE | Cannot lock the same mutex twice in the same task. |
| 0x00001450 | eERR_HMPL_INVALID_SYS_TIME_MEMORY | Buffer too small, minimum size must be 30 Byte. |
| 0x00001451 | eERR_HMPL_NOT_SUPPORTED | This HMPL function not supported for this platform. |
| 0x00001452 | eERR_HMPL_CLIENT_NOT_CONNECTED | Cannot send as client disconnected. |
| 0x0000176f | eERR_HMPL_INTERNAL_ERROR | HMPL internal error. |
| 0x00001770 | eERR_HMPL_EXEC_FAILED | HMPL function execution failed. |
| 0x00001771 | eERR_HMPL_ASM_LOAD_FAILED | HMPL compilation failed, assembly file empty or not generated. |
| 0x00001772 | eERR_HMPL_STARTTASK_TIMEOUT | HMPL StartTask function timeout. |
| 0x00001773 | eERR_HMPL_STOPTASK_TIMEOUT | HMPL StopTask function timeout. |
| 0x000017d4 | eERR_ASCII_CONNECT_TIMEOUT | ASCII client connection timeout. |
| 0x000017d5 | eERR_ASCII_CONNECT_FAILED | ASCII client connection failed. Please check ip and port. |
| 0x000017d6 | eERR_ASCII_MULTI_CONNECTING | Multiple ASCII clients connecting at the same time. |
| 0x000017d7 | eERR_ASCII_MULTI_DISCONNECTING | Multiple ASCII clients disconnecting at the same time. |
| 0x000017d8 | eERR_ASCII_DISCONNECT_TIMEOUT | ASCII client disconnection timeout. |
| 0x000017de | eERR_ASCII_RECV_TIMEOUT | ASCII client receive timeout. Please try again later. |
| 0x000017df | eERR_ASCII_RECV_FAIL | ASCII client receive failed. Please check if the connection is still alive. |
| 0x000017e0 | eERR_ASCII_MULTI_RECVING | Multiple ASCII clients receiving at the same time. |
| 0x000017e8 | eERR_ASCII_SEND_TIMEOUT | ASCII client send timeout. Please try again later. |
| 0x000017e9 | eERR_ASCII_SEND_FAIL | ASCII client send failed. Please check if the connection is still alive. |
| 0x000017ea | eERR_ASCII_MULTI_SENDING | Multiple ASCII clients sending at the same time. |
| 0x00001838 | eERR_MODBUS_CONNECT_TIMEOUT | Modbus client connection timeout. |
| 0x00001839 | eERR_MODBUS_CONNECT_FAILED | Modbus client connection failed. Please check ip. |
| 0x0000183a | eERR_MODBUS_MULTI_CONNECTING | Multiple Modbus clients connecting at the same time. |
| 0x0000183b | eERR_MODBUS_MULTI_DISCONNECTING | Multiple Modbus clients disconnecting at the same time. |
| 0x0000183c | eERR_MODBUS_DISCONNECT_TIMEOUT | Modbus client disconnection timeout. |
| 0x0000183d | eERR_MODBUS_DATALENGTH_ERR | Modbus client's read/write data number exceeds the limitation. |
| 0x0000183e | eERR_MODBUS_SOCKET_BUSY | Modbus client deals with two or more commands at the same time. |
| 0x0000183f | eERR_MODBUS_JOB_TIMEOUT | Modbus client job execution timeout. Please try again later. |

| System Error Codes | | |
|---|---|---|
| Error Code | Error Name | Description |
| 0x00001840 | eERR_MODBUS_JOB_FAIL | Modbus client job execution failed. Please check if the connection is still alive. |

## 17.1.2 Axis error massages

The following error codes appear due to an error or invalid operation in an axis. Symbols □□ will be the axis ID in hexadecimal format. For example, 01 stands for axis index 01; 0f stands for axis index 15.

Table 17.1.2.1

| Axis Error Codes | | |
|---|---|---|
| Error Code | Error Name | Description |
| 0x83□□000a | eERR_AXIS_CMD_UNKOWN | The command name is unknown. |
| 0x83□□001e | eERR_AXIS_CMD_QUEUE_FULL | Axis command queue is full. |
| 0x83□□0064 | eERR_AXIS_CMD_INVALID_STATE | The axis is unable to execute the command in current motion state. |
| 0x83□□006e | eERR_AXIS_CMD_INVALID_ENABLED | The command is not allowed while enabled. |
| 0x83□□0078 | eERR_AXIS_CMD_INVALID_DISABLED | The command is not allowed while disabled. |
| 0x83□□0082 | eERR_AXIS_CMD_INVALID_MOVING | The axis is unable to execute the command while moving. |
| 0x83□□008c | eERR_AXIS_CMD_INVALID_STOPPING | The command is invalid when axis stops moving. |
| 0x83□□0096 | eERR_AXIS_CMD_INVALID_ERROR_STATE | The command is invalid when axis is in ErrorStop state. |
| 0x83□□00a0 | eERR_AXIS_CMD_INVALID_IN_SYNC | The command is invalid when axis is in synchronized motion state. |
| 0x83□□00aa | eERR_AXIS_CMD_INVALID_GEAR_MASTER | The command is invalid when axis is the gear master axis. |
| 0x83□□00b4 | eERR_AXIS_CMD_INVALID_PP_MODE | The command is invalid when axis is in PP mode. |
| 0x83□□00be | eERR_AXIS_CMD_INVALID_MAP_SWITCHING | The command is invalid when axis is switching the compensation map. |
| 0x83□□00c8 | eERR_AXIS_CMD_INVALID_INPUTSHAPING_ENABLED | The axis is unable to execute the command when position command shaping function is activated. |
| 0x83□□00d2 | eERR_AXIS_CMD_INVALID_COMP_ENABLED | The axis is unable to execute the command when dynamic compensation is enabled. |
| 0x83□□00dc | eERR_AXIS_CMD_INVALID_GANTRY_MODE | The axis is unable to execute the command in gantry mode. |
| 0x83□□00e6 | eERR_AXIS_CMD_INVALID_GROUPED | The command is not allowed when axis is in an axis group. |
| 0x83□□00f0 | eERR_AXIS_CMD_INVALID_CONTROL_MODE | The command is invalid in current control mode. |
| 0x83□□00fa | eERR_AXIS_CMD_INVALID_OP_MODE | The operational mode is invalid. |
| 0x83□□0104 | eERR_AXIS_CMD_INVALID_BUFFER_MODE | The axis buffer mode is invalid. |
| 0x83□□010e | eERR_AXIS_CMD_INVALID_TP_ENABLED | The command is not allowed when Touch Probe is enabled. |
| 0x83□□012c | eERR_AXIS_CMD_INVALID_PARAMETER | The parameter of axis command is invalid. |
| 0x83□□0136 | eERR_AXIS_CMD_INVALID_POS | Axis target position is out of allowable range. |
| 0x83□□0140 | eERR_AXIS_CMD_INVALID_VEL | Axis velocity setting is out of allowable range. |
| 0x83□□014a | eERR_AXIS_CMD_INVALID_ACC | Axis acceleration setting is out of allowable range. |
| 0x83□□0154 | eERR_AXIS_CMD_INVALID_DEC | Axis deceleration setting is out of allowable range. |
| 0x83□□015e | eERR_AXIS_CMD_INVALID_JERK | Axis jerk setting is out of allowable range. |
| 0x83□□0168 | eERR_AXIS_CMD_INVALID_SM_TIME | Axis smooth time setting is out of allowable range. |

| Axis Error Codes | | |
|---|---|---|
| Error Code | Error Name | Description |
| 0x83□□0172 | eERR_AXIS_CMD_INVALID_KILL_DEC | Axis kill deceleration setting is out of allowable range. |
| 0x83□□017c | eERR_AXIS_CMD_INVALID_VEL_SCALE | Axis velocity scale setting is out of allowable range. |
| 0x83□□0190 | eERR_AXIS_COMP_NOT_CNFG | Axis dynamic compensation settings have not been configured properly. |
| 0x83□□01c2 | eERR_AXIS_CMD_INVALID_MASTER_SLAVE_CONNECTION | Master-slave relationship setting is invalid. |
| 0x83□□01cc | eERR_AXIS_CMD_INVALID_SLAVE_ID | Slave ID setting is invalid. |
| 0x83□□01d6 | eERR_AXIS_CMD_INVALID_GEAR_RATIO | The gear ratio setting of slave axis is out of allowable range. |
| 0x83□□01f4 | eERR_AXIS_CMD_INVALID_ROLLOVER_POS | Invalid axis rollover position, should be a positive value. |
| 0x83□□03f2 | eERR_AXIS_DRIVE_FAULT | The drive has reported a fault. Please check the corresponding error message in the drive. |
| 0x83□□03fc | eERR_AXIS_DRIVE_ABNORMAL_DISABLE | The drive is abnormally disabled. |
| 0x83□□0406 | eERR_AXIS_DRIVE_ENABLE_TOUT | It took too long to enable the drive. |
| 0x83□□0410 | eERR_AXIS_DRIVE_CLEAR_ERROR_TOUT | It took too long to clear drive error. |
| 0x83□□041a | eERR_AXIS_DRIVE_DISABLE_TOUT | It took too long to disable the drive. |
| 0x83□□0424 | eERR_AXIS_DRIVE_HOME_TOUT | It took too long to home the axis. |
| 0x83□□042e | eERR_AXIS_DRIVE_HOME_FAILED | Axis homing error. Please check error code from drive. |
| 0x83□□0456 | eERR_AXIS_VEL_LIMIT | The reference velocity has exceeded the velocity limit. |
| 0x83□□0460 | eERR_AXIS_ACC_LIMIT | The reference acceleration has exceeded the acceleration limit. |
| 0x83□□046a | eERR_AXIS_CURR_LIMIT | The current command has exceeded the current limit. |
| 0x83□□0474 | eERR_AXIS_DAMPINGRATIO_LIMIT | The damping ratio setting of axes is out of allowable range. |
| 0x83□□047e | eERR_AXIS_FREQUENCY_LIMIT | The frequency setting of axis is out of allowable range. |
| 0x83□□07da | eERR_AXIS_SWRL | Axis reference position reached right software limit. |
| 0x83□□07e4 | eERR_AXIS_SWLL | Axis reference position reached left software limit. |
| 0x83□□07ee | eERR_AXIS_HWRL | Axis right hardware limit signal triggered. |
| 0x83□□07f8 | eERR_AXIS_HWLL | Axis left hardware limit signal triggered. |
| 0x83□□0802 | eERR_AXIS_COMP_LIMIT | Axis compensation position has exceeded maximum compensation limit. |
| 0x83□□083e | eERR_AXIS_PERR | Axis position error has exceeded the protection limit. Please first check if there is any mechanical interference for motor motion. |
| 0x83□□0848 | eERR_AXIS_VERR | Axis velocity error has exceeded the protection limit. Please first check if there is any mechanical interference for motor motion. |
| 0x83□□08a2 | eERR_AXIS_PVT_MOTION_VEL_LIMIT | Velocity of axis PVT motion has exceeded the protection limit. Please first check if the given parameters are valid. |

| Axis Error Codes | | |
|---|---|---|
| Error Code | Error Name | Description |
| 0x83□□08ac | eERR_AXIS_PVT_MOTION_ACC_LIMIT | Acceleration of axis PVT motion has exceeded the protection limit. Please first check if the given parameters are valid. |
| 0x83□□08b6 | eERR_AXIS_PVT_MOTION_INVALID_TIME | Time sequence of axis PVT motion is invalid. Please first check if the given parameters are valid. |
| 0x83□□0bb8 | eERR_AXIS_CTRL_ERR | Axis internal control error. |
| 0x83□□0fa0 | eERR_AXIS_CMD_GEAR_DISABLED | Gear command is not allowed while gear is disabled. |
| 0x83□□0fa1 | eERR_AXIS_CMD_INVALID_AXIS_IN_CAM | Gear command is invalid when axis is in cam |
| 0x83□□1388 | eERR_CAM_CMD_INVALID_ENGAGE_WINDOW | Cam engage window is out of allowable range. |
| 0x83□□1389 | eERR_CAM_CMD_INVALID_ENGAGE_POSITION | Cam engage position is out of cam table domain. |
| 0x83□□138a | eERR_CAM_CMD_INVALID_MASTER_SCALE_FACTOR | Cam master scale factor is out of allowable range. |
| 0x83□□138b | eERR_CAM_CMD_INVALID_CAM_SCALE_FACTOR | Cam scale factor is out of allowable range. |
| 0x83□□138c | eERR_CAM_CMD_INVALID_CAMTABLE_ID | Cam table ID is out of allowable range. |
| 0x83□□138d | eERR_CAM_CMD_INVALID_DISENGAGE_WINDOW | Cam disengage window is out of allowable range. |
| 0x83□□138e | eERR_CAM_CMD_INVALID_DISENGAGE_POSITION | Cam disengage position is out of allowable range. |
| 0x83□□138f | eERR_CAM_CMD_INVALID_OPERATION_IN_ENGAGED_STATE | Cam command is invalid in engage state. |
| 0x83□□1390 | eERR_CAM_CMD_INVALID_START_MODE | Cam start mode does not correspond to a valid enumeration value. |
| 0x83□□1391 | eERR_CAM_CMD_INVALID_MOVE_MODE | Cam move mode does not correspond to a valid enumeration value. |
| 0x83□□1392 | eERR_CAM_ENGAGED_FAILED | CamMaster may pass through the engage window because engage window is too small. |
| 0x83□□1393 | eERR_CAM_CMD_NOTINUSE | End of profile mode is not in use right now |
| 0x83□□1394 | eERR_CAM_CMD_CAM_DISABLED | Cam command is not allowed while cam is disabled |
| 0x83□□1395 | eERR_CAM_CMD_INVALID_AXIS_NOT_INCAM | Cam command is invalid when axis is not in cam |
| 0x83□□1396 | eERR_CAM_CMD_INVALID_AXIS_NOT_IN_DISENGAGED | Cam command is invalid when axis is not in disengaged |
| 0x83□□1397 | eERR_CAM_CMD_INVALID_AXIS_IN_GEAR | Cam command is invalid when axis is in gear |

## 17.1.3 Group error messages

The following error codes appear due to an error or invalid operation in an axis group. Symbols □□ will be the axis group ID in hexadecimal format. For example, 01 stands for axis group index 01; 0f stands for axis group index 15.

Table 17.1.3.1

| Axis Group Error Codes | | |
|---|---|---|
| Error Code | Error Name | Description |
| 0x82□□000a | eERR_CRD_CMD_UNKNOWN | The axis group command is unknown. |
| 0x82□□0028 | eERR_CRD_CMD_AXIS_DUPLICATED | Could not add the axis since it is already in the group. |
| 0x82□□0032 | eERR_CRD_CMD_GRP_SIZE_EMPTY | The axis group is empty. |
| 0x82□□003c | eERR_CRD_CMD_GRP_SIZE_FULL | The axis group is full and cannot hold any more axis. |
| 0x82□□0046 | eERR_CRD_CMD_INVALID_MOVING | The command is invalid while the axis group is moving. |
| 0x82□□0050 | eERR_CRD_CMD_INVALID_DISABLED | The command is invalid while the axis group is disabled. |
| 0x82□□005a | eERR_CRD_CMD_INVALID_INPUTSHAPING_PARAMETER_INCOMPLETE | The parameters of axis group inshape function is incomplete. |
| 0x82□□001e | eERR_CRD_CMD_INVALID_KIN_SETTING | The kinematics type setting is invalid. |
| 0x82□□001f | eERR_CRD_CMD_INVALID_SPECIFIC_KIN | The command is invalid when axis group is in specific kinematics type. |
| 0x82□□006e | eERR_CRD_CMD_INVALID_STATE | The axis group is unable to execute the command in current motion state. |
| 0x82□□0078 | eERR_CRD_CMD_QUEUE_FULL | Please wait till the last command is done. |
| 0x82□□0082 | eERR_CRD_CMD_GRP_AXIS_INVALID | The group axis is invalid. |
| 0x82□□00d2 | eERR_CRD_CMD_INVALID_POS | The axis group target position or orientation is out of allowable range. |
| 0x82□□00dc | eERR_CRD_CMD_INVALID_LIN_VEL | The linear velocity setting of axis group is out of allowable range. |
| 0x82□□00e6 | eERR_CRD_CMD_INVALID_LIN_ACC | The linear acceleration setting of axis group is out of allowable range. |
| 0x82□□00f0 | eERR_CRD_CMD_INVALID_LIN_DEC | The linear deceleration setting of axis group is out of allowable range. |
| 0x82□□00fa | eERR_CRD_CMD_INVALID_LIN_JERK | The linear jerk setting of axis group is out of allowable range. |
| 0x82□□0104 | eERR_CRD_CMD_INVALID_LIN_SM_TIME | The linear smooth time setting of axis group is out of allowable range. |
| 0x82□□010e | eERR_CRD_CMD_INVALID_DAMPINGRATIO | The damping ratio setting of axis group is out of allowable range. |
| 0x82□□0118 | eERR_CRD_CMD_INVALID_FREQUENCY | The frequency setting of axis group is out of allowable range. |
| 0x82□□0140 | eERR_CRD_CMD_INVALID_ANG_VEL | The angular velocity setting of axis group is out of allowable range. |
| 0x82□□014a | eERR_CRD_CMD_INVALID_ANG_ACC | The angular acceleration setting of axis group is out of allowable range. |
| 0x82□□0154 | eERR_CRD_CMD_INVALID_ANG_DEC | The angular deceleration setting of axis group is out of allowable range. |
| 0x82□□015e | eERR_CRD_CMD_INVALID_ANG_JERK | The angular jerk setting of axis group is out of allowable range. |
| 0x82□□0168 | eERR_CRD_CMD_INVALID_ANG_SM_TIME | The angular smooth time setting of axis group is out of allowable range. |

| Axis Group Error Codes | | |
|---|---|---|
| Error Code | Error Name | Description |
| 0x82□□0190 | eERR_CRD_CMD_INVALID_VEL_SCALE | The velocity scale of axis group is out of allowable range. |
| 0x82□□019a | eERR_CRD_CMD_INVALID_TRANS_VEL | The transition velocity of axis group is invalid. |
| 0x82□□01a4 | eERR_CRD_CMD_INVALID_TRANS_DIS | The transition distance of axis group is invalid. |
| 0x82□□01a5 | eERR_CRD_CMD_INVALID_TRANS_DEV | The transition deviation of axis group is invalid. |
| 0x82□□01a6 | eERR_CRD_CMD_INVALID_TRANS_CURVE | The transition curvature of axis group is invalid. |
| 0x82□□01b8 | eERR_CRD_CMD_TRANS_MODE_UNKNOWN | The path transition mode name is unknown. |
| 0x82□□01c2 | eERR_CRD_CMD_COORD_SYS_UNKNOWN | The coordinate system is unknown. |
| 0x82□□01cc | eERR_CRD_CMD_BLEND_MODE_UNKNOWN | The path blending mode name is unknown. |
| 0x82□□01fe | eERR_CRD_CMD_LIN_INVALID_PARAM | The parameters are invalid for linear path planning. |
| 0x82□□0262 | eERR_CRD_CMD_CIRC_INVALID_PARAM | The parameters are invalid for circular path planning. |
| 0x82□□026c | eERR_CRD_CMD_CIRC_INVALID_CENTER | The center position of circular path is too close to start / end point. |
| 0x82□□0276 | eERR_CRD_CMD_CIRC_ANGLE_SMALL | The central angle of circular path is too small. |
| 0x82□□0280 | eERR_CRD_CMD_CIRC_INVALID_RADIUS | The radius of circular path is invalid. |
| 0x82□□028a | eERR_CRD_CMD_CIRC_INVALID_COORD | The coordinate system of circular path is invalid. |
| 0x82□□02c6 | eERR_CRD_CMD_BEZIER_INVALID_PARAM | The parameters are invalid for Bezier curve path planning. |
| 0x82□□02d0 | eERR_CRD_CMD_BSPLINE_INVALID_PARAM | The parameters are invalid for BSpline curve path planning. |
| 0x82□□02da | eERR_CRD_CMD_CURVE_INVALID_STARTPOS | The start position is invalid for curve path planning. |
| 0x82□□02e4 | eERR_CRD_CMD_COORD_INVALID_PARAM | The parameters are invalid for coordinate transformation. |
| 0x82□□02ee | eERR_CRD_CMD_NURBS_INVALID_PARAM | The parameters are invalid for NURBS curve path planning. |
| 0x82□□03f2 | eERR_CRD_AXIS_ABNORMALLY_DISABLED | One or more axes in the axis group are abnormally disabled. |
| 0x82□□03fc | eERR_CRD_AXIS_SWL | One of the axes in axis group touches software limit. |

## 17.2 HIMC_GetLastError

### Purpose

To get the latest error code of the controller.

### Syntax

```
int HIMC_GetLastError(
    int ctrl_id,
    int *p_error_code
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| p_error_ code [out] | A pointer to the buffer to receive the latest error code of the controller.<br>**Refer to section 17.1.1 for the definition.** |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetLastError |
| LabVIEW | HIMC Get Last Error.vi |
| Python | GetLastError |

# 17.3 HIMC_GetAxisLastErr

## Purpose

To get the lastest error code of an axis.

## Syntax

```
int HIMC_GetAxisLastErr(
    int ctrl_id,
    int axis_id,
    int *err_code
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| err_code [out] | A pointer to the buffer to receive the lastest error code of an axis.<br>**Refer to section 17.1.2 for the definition.** |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetAxisLastErr |
| LabVIEW | HIMC Get Axis Last Err.vi |
| Python | GetAxisLastErr |

## 17.4 HIMC_ClearAxisLastErr

### Purpose

To clear the lastest error code of an axis.

### Syntax

```
int HIMC_ClearAxisLastErr(
    int ctrl_id,
    int axis_id
);
```

### Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ClearAxisLastErr |
| LabVIEW | HIMC Clear Axis Last Err.vi |
| Python | ClearAxisLastErr |

# 17.5 HIMC_GetGrpLastErr

## Purpose

To get the lastest error code of an axis group.

## Syntax

```
int HIMC_GetGrpLastErr(
    int ctrl_id,
    int group_id,
    int *err_code
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| group_id [in] | Axis group index. |
| err_code [out] | A pointer to the buffer to receive the lastest error code of an axis group.<br>**Refer to section 17.1.3 for the definition.** |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ClearAxisLastErr |
| LabVIEW | HIMC Clear Axis Last Err.vi |
| Python | ClearAxisLastErr |

# 17.6 HIMC_ClearGrpLastErr

## Purpose

To clear the lastest error code of an axis group.

## Syntax

```
int HIMC_ClearGrpLastErr(
    int ctrl_id,
    int group_id
);
```

## Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

group_id [in] Axis group index.

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_ClearGrpLastErr |
| LabVIEW | HIMC Clear Grp Last Err.vi |
| Python | ClearGrpLastErr |

# 17.7 HIMC_GetDriveErr

## Purpose

To get the error code of a drive.

## Syntax

```
int HIMC_GetDriveErr(
    int ctrl_id,
    int axis_id,
    int *err_code
);
```

## Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis Index. |
| err_code [in] | A pointer to the buffer to receive the error code of a drive. |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Remark

Users must configure object 0x603F(Error code) as PDO when using this function.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetDriveErr |
| LabVIEW | HIMC Get Drive Err.vi |
| Python | GetDriveErr |

## 17.8 HIMC_GetErrorInformation

**Purpose**

To get the information of a given error ID.

**Syntax**

```
int HIMC_GetErrorInfomation(
    int error_id,
    char *p_name,
    int name_buff_len,
    int *p_name_actual_len,
    char *p_description,
    int description_buff_len,
    int *p_description_actual_len
);
```

**Parameter**

| | |
|---|---|
| error_id [in] | Specify an error ID. |
| p_name [out] | A pointer to the buffer to receive the error name of the given error ID. |
| name_buff_len [in] | Specify the buffer's maximum number of the characters to receive the error name. |
| p_name_actual_len [out] | A pointer to the buffer to receive the error name's actual number of the characters. (Exclude the terminating null character.) |
| p_description [out] | A pointer to the buffer to receive the error description of the given error ID. |
| description_buff_len [in] | Specify the buffer's maximum number of the characters to receive the error description. |
| p_description_actual_len [out] | A pointer to the buffer to receive the error description's actual number of the characters. (Exclude the terminating null character.) |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

## Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_GetErrorInformation |
| LabVIEW | HIMC Get Error Information.vi |
| Python | GetErrorInformation |

# 18. Homing functions


18. Homing functions ..... 18-1
18.1 Overview ..... 18-2
18.2 HIMC_MoveHome ..... 18-9
18.3 HIMC_SetHomeMethod ..... 18-10
18.4 HIMC_SetHomeSwitchVel ..... 18-12
18.5 HIMC_SetHomeZeroVel ..... 18-13
18.6 HIMC_SetHomeAcc ..... 18-14
18.7 HIMC_SetHomeOffset ..... 18-15
18.8 HIMC_SetHomeTimeout ..... 18-16
18.9 HIMC_IsHomed ..... 18-17
18.10 HIMC_IsHoming ..... 18-18

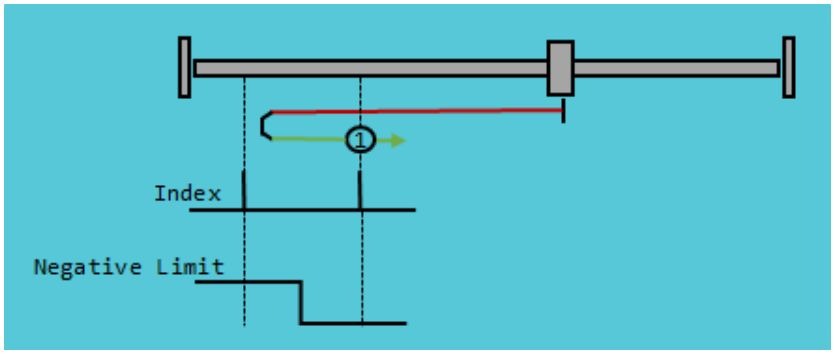
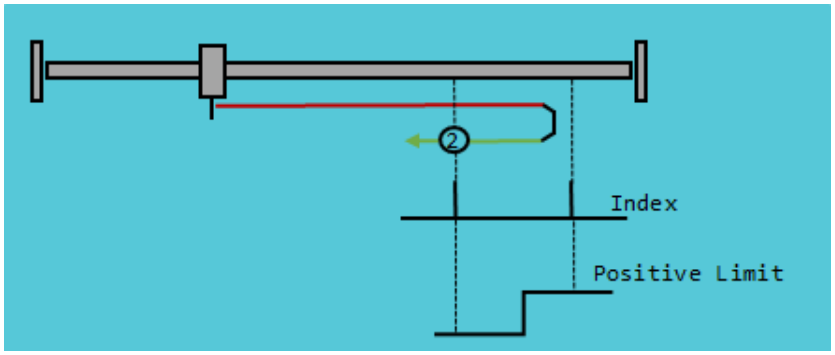
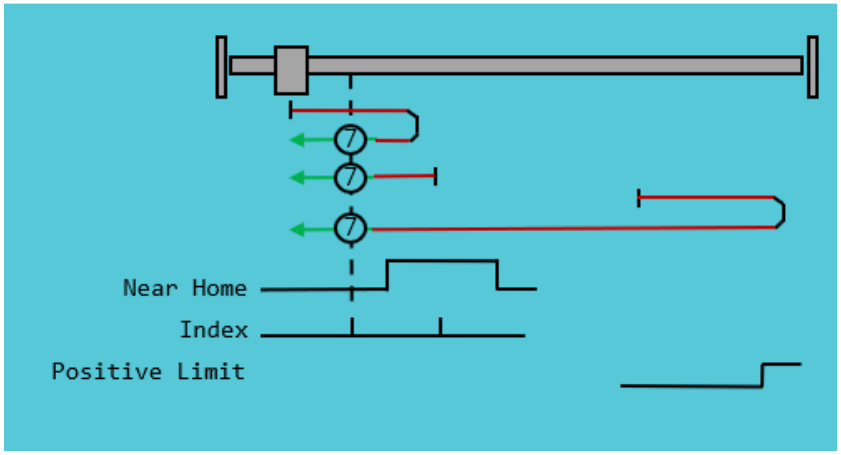
# 18.1 Overview

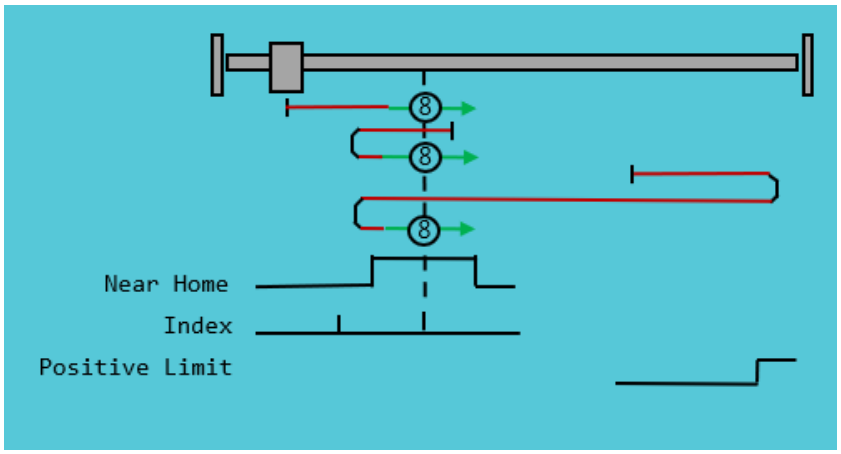
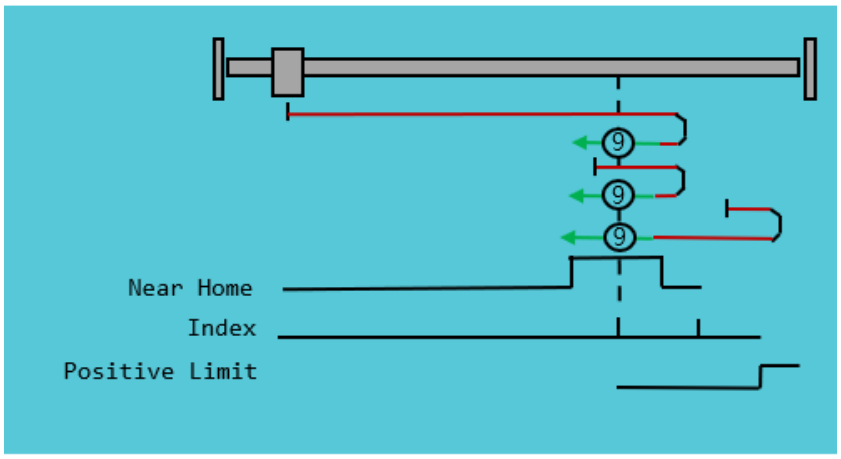
HIMC supports homing method CiA 402, which allows users to set the homing method of each axis based on stage configuration. All homing methods are listed in Table 18.1.1, and the detailed diagrams and descriptions are shown in Table 18.1.2.

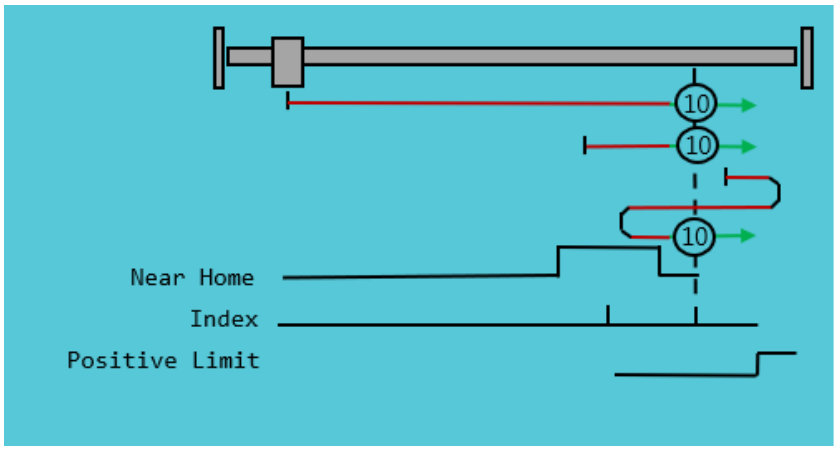
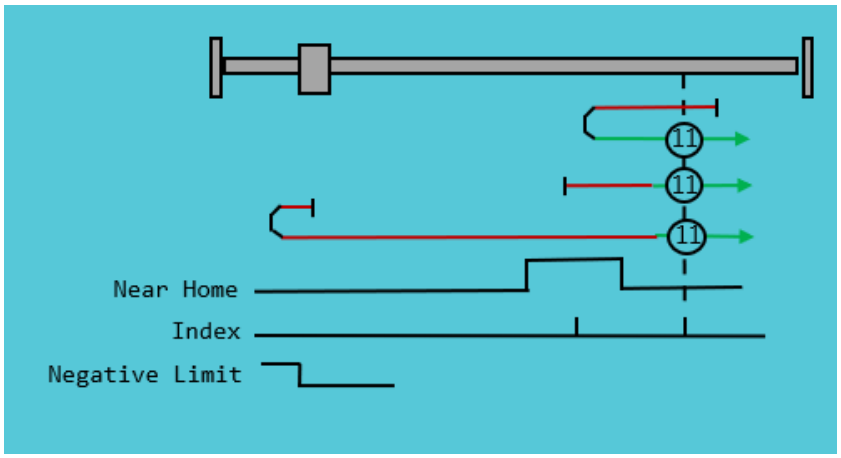
Table 18.1.1

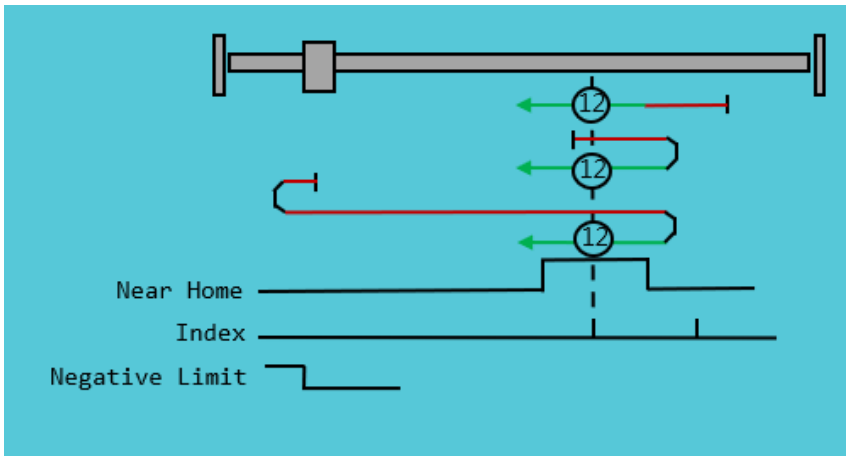
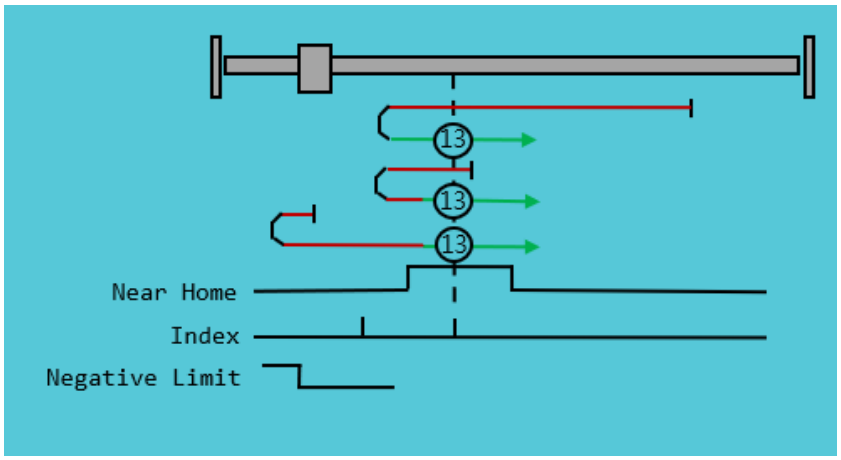
| HMPL definition | Description |
|---|---|
| HOME_METHOD_1 | Search for negative limit in negative direction first, and search for index in positive direction. |
| HOME_METHOD_2 | Search for positive limit in positive direction first, and search for index in negative direction. |
| HOME_METHOD_7 | Search for index on the left of near home rising-edge in positive direction. |
| HOME_METHOD_8 | Search for index on the right of near home rising-edge in positive direction. |
| HOME_METHOD_9 | Search for index on the left of near home falling-edge in positive direction. |
| HOME_METHOD_10 | Search for index on the right of near home falling-edge in positive direction. |
| HOME_METHOD_11 | Search for index on the right of near home rising-edge in negative direction. |
| HOME_METHOD_12 | Search for index on the left of near home rising-edge in negative direction. |
| HOME_METHOD_13 | Search for index on the right of near home falling-edge in negative direction. |
| HOME_METHOD_14 | Search for index on the left of near home falling-edge in negative direction. |
| HOME_METHOD_17 | Search for negative limit in negative direction first, and move to the position of home offset. |
| HOME_METHOD_18 | Search for positive limit in positive direction first, and move to the position of home offset. |
| HOME_METHOD_19 | Search for negative limit in negative direction first, and search for positive limit in positive direction. Finally, move to the middle of positive and negative limit. |
| HOME_METHOD_33 | Search for index in negative direction first, and move to the position of home offset. |
| HOME_METHOD_34 | Search for index in positive direction first, and move to the position of home offset. |
| HOME_METHOD_37 | Set the current position of the motor as home position. |

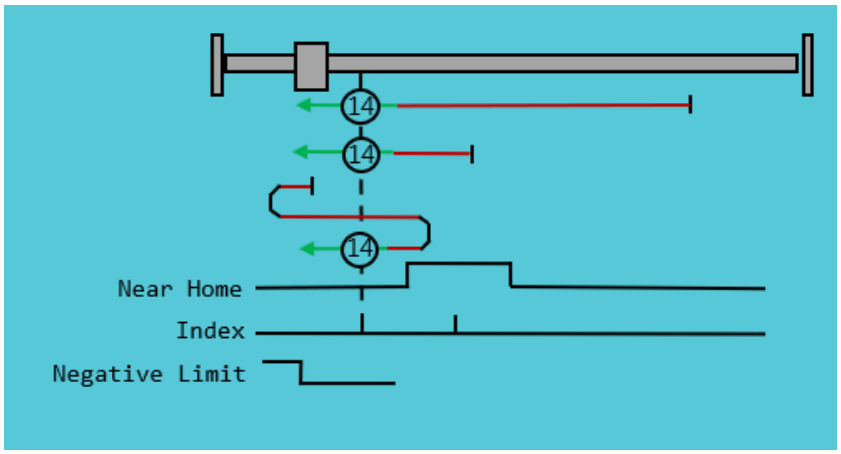
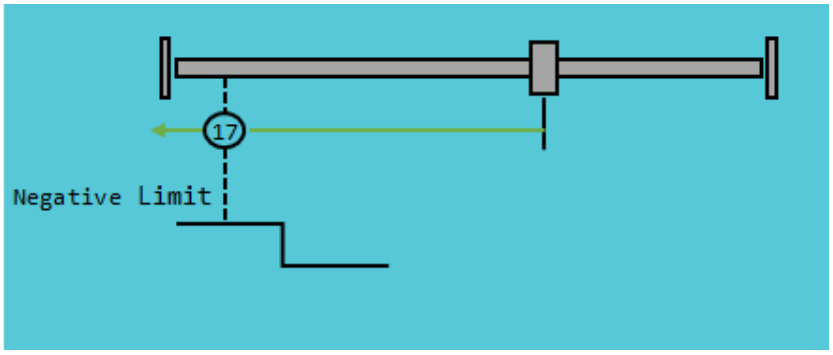
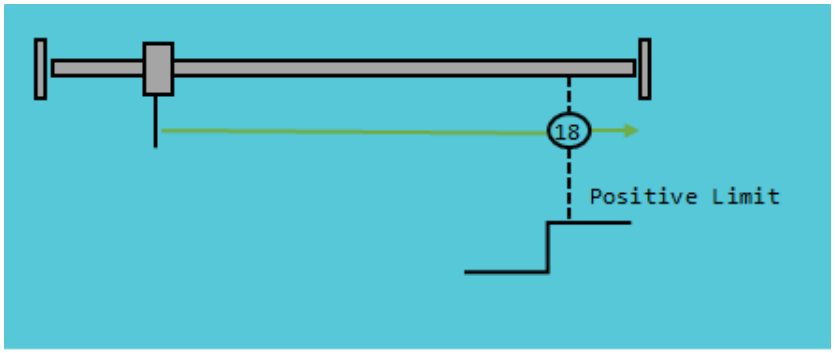
Table 18.1.2

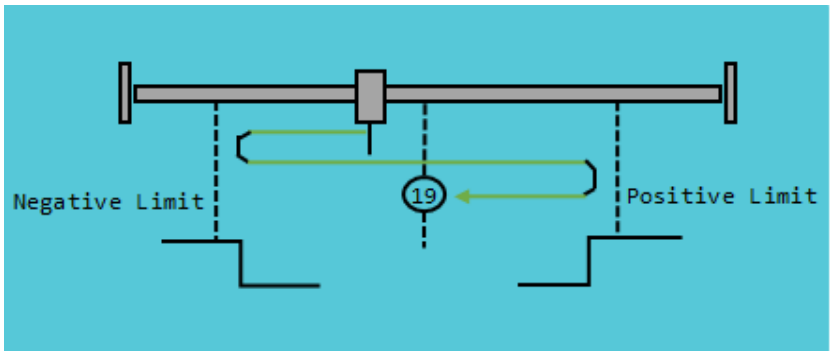
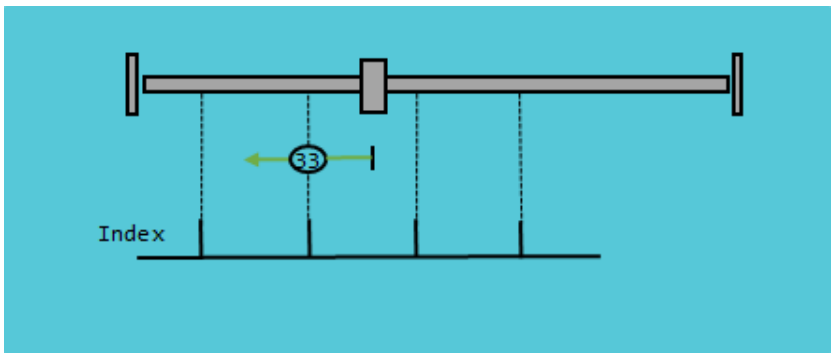
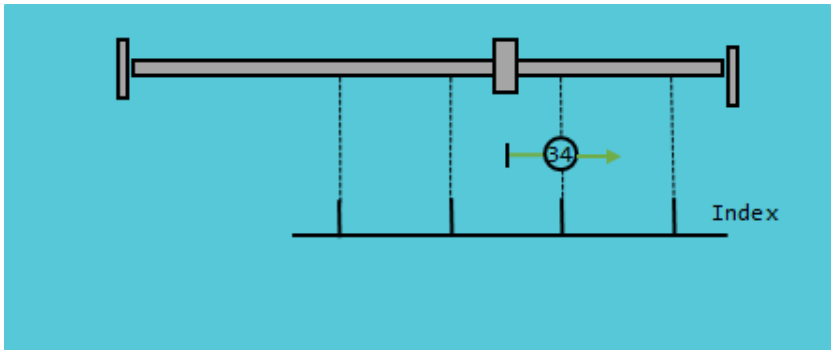
| HMPL definition | Homing procedure |
|---|---|
| HOME_METHOD_1 |  <br>Search for negative limit in negative direction at fast velocity first, and search for index in positive direction at slow velocity. Take the position of index as home position, and move to home offset at slow velocity. |
| HOME_METHOD_2 |  <br>Search for positive limit in positive direction at fast velocity first, and search for index in negative direction at slow velocity. Take the position of index as home position, and move to home offset at slow velocity. |
| HOME_METHOD_7 |  <br>1. (not on near home) Search for near home rising-edge in positive direction first, and search for index in negative direction.<br>2. (on near home) Search for near home falling-edge in negative direction first, and search for index in negative direction.<br>3. (not on near home) Search for positive limit in positive direction first, and search for near home falling-edge in negative direction. Finally, search for index in negative direction. |

| HMPL definition | Homing procedure |
|---|---|
| HOME_METHOD_8 |  |
| | 1. (not on near home) Search for near home rising-edge in positive direction first, and search for index in positive direction.<br>2. (on near home) Search for near home falling-edge in negative direction first, and search for index in positive direction.<br>3. (not on near home) Search for positive limit in positive direction first, and search for near home falling-edge in negative direction. Finally, search for index in positive direction. |
| HOME_METHOD_9 |  |
| | 1. (not on near home) Search for near home falling-edge in positive direction first, and search for index in negative direction.<br>2. (on near home) Search for near home falling-edge in positive direction first, and search for index in negative direction.<br>3. (not on near home) Search for positive limit in positive direction first, and search for near home rising-edge in negative direction. Finally, search for index in negative direction. |

| HMPL definition | Homing procedure |
|---|---|
| HOME_METHOD_10 |  |
| | 1. (not on near home) Search for near home falling-edge in positive direction first, and search for index in positive direction.<br>2. (on near home) Search for near home falling-edge in positive direction first, and search for index in positive direction.<br>3. (not on near home) Search for positive limit in positive direction first, and search for near home rising-edge in negative direction. Finally, search for index in positive direction. |
| HOME_METHOD_11 |  |
| | 1. (not on near home) Search for near home rising-edge in negative direction first, and search for index in positive direction.<br>2. (on near home) Search for near home falling-edge in positive direction first, and search for index in positive direction.<br>3. (not on near home) Search for negative limit in negative direction first, and search for near home falling-edge in positive direction. Finally, search for index in positive direction. |

| HMPL definition | Homing procedure |
|---|---|
| HOME_METHOD_12 |  |
| | 1. (not on near home) Search for near home rising-edge in negative direction first, and search for index in negative direction.<br>2. (on near home) Search for near home falling-edge in positive direction first, and search for index in negative direction.<br>3. (not on near home) Search for negative limit in negative direction first, and search for near home falling-edge in positive direction. Finally, search for index in negative direction. |
| HOME_METHOD_13 |  |
| | 1. (not on near home) Search for near home falling-edge in negative direction first, and search for index in positive direction.<br>2. (on near home) Search for near home falling-edge in negative direction first, and search for index in positive direction.<br>3. (not on near home) Search for negative limit in negative direction first, and search for near home rising-edge in positive direction. Finally, search for index in positive direction. |

| HMPL definition | Homing procedure |
|---|---|
| HOME_METHOD_14 |  |
| | 1. (not on near home) Search for near home falling-edge in negative direction first, and search for index in negative direction.<br>2. (on near home) Search for near home falling-edge in negative direction first, and search for index in negative direction.<br>3. (not on near home) Search for negative limit in negative direction first, and search for near home rising-edge in positive direction. Finally, search for index in negative direction. |
| HOME_METHOD_17 |  |
| | Search for negative limit in negative direction at slow velocity. Take the position of negative limit as home position, and move to home offset at slow velocity. |
| HOME_METHOD_18 |  |
| | Search for positive limit in positive direction at slow velocity. Take the position of positive limit as home position, and move to home offset at slow velocity. |

| HMPL definition | Homing procedure |
|---|---|
| HOME_METHOD_19 |  |
| | Search for negative limit in negative direction at slow velocity first, and search for positive limit in positive direction at slow velocity. Move to the middle at slow velocity, and take the middle position as home position. |
| HOME_METHOD_33 |  |
| | Search for index in negative direction at slow velocity. Take index as home position, and move to home offset at slow velocity. |
| HOME_METHOD_34 |  |
| | Search for index in positive direction at slow velocity. Take index as home position, and move to home offset at slow velocity. |

**Note: Homing procedure does not support simulator.**

## 18.2 HIMC_MoveHome

**Purpose**

To execute homing procedure of an axis.

**Syntax**

```
int HIMC_MoveHome(
    int    ctrl_id,
    int    axis_id
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Remark**

Users must configure object 0x6060(Mode of operation) and object 0x6061(Mode of operation display) as PDO when using this function.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_MoveHome |
| LabVIEW | HIMC Move Home.vi |
| Python | MoveHome |

# 18.3 HIMC_SetHomeMethod

## Purpose

To set the homing method of homing procedure.

## Syntax

```
int HIMC_SetHomeMethod(
    int    ctrl_id,
    int    axis_id,
    int    method
);
```

## Parameter

ctrl_id [in] A controller ID for HIWIN Motion Controller.
It must be obtained by calling HIMC_ConnectCtrl.

axis_id [in] Axis index.

method [in] Description of HMPL definition for homing method index.
**Refer to Table 18.1.1 and Table 18.1.2 for details.**
The default value is HOME_METHOD_33.

| Homing method index | Description of HMPL definition | Homing method index | Description of HMPL definition |
|---|---|---|---|
| 1 | HOME_METHOD_1 | 13 | HOME_METHOD_13 |
| 2 | HOME_METHOD_2 | 14 | HOME_METHOD_14 |
| 7 | HOME_METHOD_7 | 17 | HOME_METHOD_17 |
| 8 | HOME_METHOD_8 | 18 | HOME_METHOD_18 |
| 9 | HOME_METHOD_9 | 19 | HOME_METHOD_19 |
| 10 | HOME_METHOD_10 | 33 | HOME_METHOD_33 |
| 11 | HOME_METHOD_11 | 34 | HOME_METHOD_34 |
| 12 | HOME_METHOD_12 | 37 | HOME_METHOD_37 |

## Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetHomeMethod |
| LabVIEW | HIMC Set Home Method.vi |
| Python | SetHomeMethod |

## 18.4 HIMC_SetHomeSwitchVel

### Purpose

To set fast homing velocity of homing procedure.

### Syntax

```
int HIMC_SetHomeSwitchVel(
    int     ctrl_id,
    int     axis_id,
    double  fast_vel
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| fast_vel [in] | Fast homing velocity, the default value is 20.<br>Parameter unit: mm/s or deg/s. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetHomeSwitchVel |
| LabVIEW | HIMC Set Home Switch Vel.vi |
| Python | SetHomeSwitchVel |

# 18.5 HIMC_SetHomeZeroVel

**Purpose**

To set slow homing velocity of homing procedure.

**Syntax**

```
int HIMC_SetHomeZeroVel(
    int     ctrl_id,
    int     axis_id,
    double  slow_vel
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| slow_vel [in] | Slow homing velocity, the default value is 5.<br>Parameter unit: mm/s or deg/s. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetHomeZeroVel |
| LabVIEW | HIMC Set Home Zero Vel.vi |
| Python | SetHomeZeroVel |

## 18.6 HIMC_SetHomeAcc

### Purpose

To set homing acceleration of homing procedure.

### Syntax

```
int HIMC_SetHomeAcc(
    int     ctrl_id,
    int     axis_id,
    double  acc
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| acc [in] | Homing acceleration, the default value is 2000.<br>Parameter unit: mm/s or deg/s. |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | `HIMC_API.h` |
| Library | `HIMC_API.lib` |
| DLL | `HIMC_API.dll` |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetHomeAcc |
| LabVIEW | HIMC Set Home Acc.vi |
| Python | SetHomeAcc |

# 18.7 HIMC_SetHomeOffset

**Purpose**

To set the home offset of homing procedure.

**Syntax**

```
int HIMC_SetHomeOffset(
    int     ctrl_id,
    int     axis_id,
    double  offset
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| offset [in] | Home offset. The default value is 0.<br>Parameter unit: mm or deg |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_SetHomeOffset |
| LabVIEW | HIMC Set Home Offset.vi |
| Python | SetHomeOffset |

## 18.8 HIMC_SetHomeTimeout

### Purpose

To set the timeout of homing procedure.

### Syntax

```
int HIMC_SetHomeTimeout(
    int     ctrl_id,
    int     axis_id,
    int     timeout
);
```

### Parameter

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| timeout [in] | Timeout. The default value is 120,000.<br>Parameter unit: ms |

### Return value

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | HIMC_SetHomeTimeout |
| LabVIEW | HIMC Set Home Timeout.vi |
| Python | SetHomeTimeout |

# 18.9 HIMC_IsHomed

**Purpose**

To query whether the axis has completed homing procedure.

**Syntax**

```
int HIMC_IsHomed(
    int     ctrl_id,
    int     axis_id,
    int    *p_is_homed
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_homed [in] | A pointer to the buffer to receive whether the axis has completed homing procedure.<br>If the axis has completed homing procedure, the value will be 1. Otherwise, it will be 0. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_IsHomed |
| LabVIEW | HIMC Is Homed.vi |
| Python | IsHomed |

# 18.10 HIMC_IsHoming

**Purpose**

To query whether the axis is operating homing procedure.

**Syntax**

```
int HIMC_IsHoming(
    int     ctrl_id,
    int     axis_id,
    int    *p_is_homing
);
```

**Parameter**

| | |
|---|---|
| ctrl_id [in] | A controller ID for HIWIN Motion Controller.<br>It must be obtained by calling HIMC_ConnectCtrl. |
| axis_id [in] | Axis index. |
| p_is_homing [in] | A pointer to the buffer to receive whether the axis is operating homing procedure.<br>If the axis is operating homing procedure, the value will be 1. Otherwise, it will be 0. |

**Return value**

It will return an **int** value **0** if the function succeeds, a **nonzero** value if the function fails.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | HIMC_IsHoming |
| LabVIEW | HIMC Is Homing.vi |
| Python | IsHoming |

# 19. Data structures

19. Data structures ..... 19-1
19.1 ComInfo ..... 19-2
19.2 CoordPosition ..... 19-3
19.3 MotionProfile ..... 19-4
19.4 CenterPosition ..... 19-5
19.5 NormalVector ..... 19-6
19.6 TransPrm ..... 19-7
19.7 PosTriggerPar ..... 19-8

# 19.1 ComInfo

## Purpose

It defines the connection type and its information.

## Syntax

```
typedef struct {
    ComType type;
    struct {
        char ip[20];
        char port[12];
    } TCP_IP;

    struct {
        char com_port_name[80];
        int  baud_rate;
    } RS232;

    struct {
        char autoExecExe;
    } Simulator;
} ComInfo;
```

## Member

| | |
|---|---|
| ComType | Connection type. |
| TCP_IP | A structure which includes network connecting parameter, ip and port. |
| RS232 | A structure which includes RS232 connecting parameter, com port name and baud rate. |
| Simulator | A structure which includes Simulator connecting parameter and autoExecExe. |

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | ComInfo |
| LabVIEW | -- |
| Python | ComInfo |

# 19.2 CoordPosition

**Purpose**

It defines the positions or distances of the coordinate motion.

**Syntax**

```
typedef struct {
    double x_pos;
    double y_pos;
    double z_pos;
    double a_pos;
    double b_pos;
    double c_pos;
} CoordPosition, *PCoordPosition;
```

**Member**

| | |
|---|---|
| x_pos | Linear position X of the end-point. Unit: mm |
| y_pos | Linear position Y of the end-point. Unit: mm |
| z_pos | Linear position Z of the end-point. Unit: mm |
| a_pos | Orientation angle A of the end-point. Unit: deg |
| b_pos | Orientation angle B of the end-point. Unit: deg |
| c_pos | Orientation angle C of the end-point. Unit: deg |

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | CoordPosition |
| LabVIEW | -- |
| Python | CoordPosition |

## 19.3 MotionProfile

### Purpose

It defines the motion profile settings.

### Syntax

```
typedef struct {
    double max_vel;
    double max_acc;
    double max_dec;
    double smooth_time;
} MotionProfile, *PMotionProfile;
```

### Member

max_vel
The maximum tangential velocity of linear motion.
Unit: mm/s or deg/s
Range: 0 ~ 5000

max_acc
The maximum tangential acceleration of linear motion.
Unit: $mm/s^2$ or $deg/s^2$
Range: >0 ~ 50000 (acceleration cannot be 0)

max_dec
The maximum tangential deceleration of linear motion.
Unit: $mm/s^2$ or $deg/s^2$
Range: >0 ~ 50000 (deceleration cannot be 0)

smooth_time
Smooth time of linear motion.
Unit: ms
Range: 0 ~ 500

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | MotionProfile |
| LabVIEW | -- |
| Python | MotionProfile |

# 19.4 CenterPosition

**Purpose**

It defines the center position settings.

**Syntax**

```
typedef struct {
    double x_pos;
    double y_pos;
    double z_pos;
} CenterPosition, *PCenterPosition;
```

**Member**

| | |
|---|---|
| x_pos | Position of X axis. Unit: mm |
| y_pos | Position of Y axis. Unit: mm |
| z_pos | Position of Z axis. Unit: mm |

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | CenterPosition |
| LabVIEW | -- |
| Python | CenterPosition |

# 19.5 NormalVector

## Purpose

It defines the normal vector settings.

## Syntax

```
typedef struct {
    double x_vector;
    double y_vector;
    double z_vector;
} NormalVector, *PNormalVector;
```

## Member

| | |
|---|---|
| x_vector | X direction vector. |
| y_vector | Y direction vector. |
| z_vector | Z direction vector. |

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | NormalVector |
| LabVIEW | -- |
| Python | NormalVector |

# 19.6 TransPrm

**Purpose**

It defines the position trigger settings.

**Syntax**

```
typedef struct {
    double trans_vel,
    double trans_dis,
    double trans_dev;
    double trans_curv;
} TransPrm, *PTransPrm;
```

**Member**

| | |
|---|---|
| trans_vel | The new transition mode's velocity parameter of an axis group.<br>Unit: mm/s |
| trans_dis | The new transition mode's distance parameter of an axis group.<br>Unit: mm |
| trans_dev | The new transition mode's maximum deviation parameter of an axis group.<br>Unit: mm |
| trans_curv | The new transition mode's curvature parameter of an axis group.<br>Unit: $mm^{-1}$ |

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 3.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | TransPrm |
| LabVIEW | -- |
| Python | TransPrm |

# 19.7 PosTriggerPar

**Purpose**

It defines the position trigger settings.

**Syntax**

```
typedef struct {
    double start_pos;
    double end_pos;
    double interval;
    int pulse_width;
} PosTriggerPar, *PPosTriggerPar;
```

**Member**

start_pos Start position of PT function.
Unit: mm or deg

end_pos End position of PT function.
Unit: mm or deg

interval Position interval between consecutive PT outputs.
Unit: mm or deg

pulse_width The width of each PT output signal.
Unit: ns

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | PosTriggerPar |
| LabVIEW | -- |
| Python | PosTriggerPar |

# 20. Enumerations

20. Enumerations ..... 20-1
20.1 ComType ..... 20-2
20.2 CoordSystem ..... 20-3
20.3 MotionBufferMode ..... 20-5
20.4 MotionTransitionMode ..... 20-6
20.5 ShaperMode ..... 20-7

# 20.1 ComType

## Definition

Connection type enumeration

## Syntax

```
typedef enum {
    COM_TYPE_TCPIP,
    COM_TYPE_RS232,
    COM_TYPE_SIMULATOR
} ComType;
```

## Member

| | |
|---|---|
| COM_TYPE_TCPIP | The connection type is TCPIP. |
| COM_TYPE_RS232 | The connection type is RS232. |
| COM_TYPE_SIMULATOR | The connection type is simulator. |

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | ComType |
| LabVIEW | -- |
| Python | ComType |

# 20.2 CoordSystem

**Definition**

Coordinate systems

**Syntax**

```
typedef enum {
    kCoord_ACS = 0,
    kCoord_MCS = 1,
    kCoord_PCS = 2,
    kCoord_GLOBAL = 3,
    kCoord_WCS1 = 1 << 8,
    kCoord_WCS2 = 2 << 8,
    kCoord_WCS3 = 3 << 8,
    kCoord_WCS4 = 4 << 8,
    kCoord_WCS5 = 5 << 8,
    kCoord_WCS6 = 6 << 8,
    kCoord_WCS7 = 7 << 8,
    kCoord_WCS8 = 8 << 8,
    kCoord_WCS9 = 9 << 8,
    kCoord_WCS10 = 10 << 8,
    kCoord_WCS11 = 11 << 8,
    kCoord_WCS12 = 12 << 8,
    kCoord_WCS13 = 13 << 8,
    kCoord_WCS14 = 14 << 8,
    kCoord_WCS15 = 15 << 8,
    kCoord_OFFSET = 1 << 15
} CoordSystem;
```

**Description**

Refer to section 6.1.2.

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 2.0 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | CoordSystem |
| LabVIEW | -- |
| Python | CoordSystem |

# 20.3 MotionBufferMode

## Definition

Buffer modes between adjacent coordinate motion segments

## Syntax

```
typedef enum {
    kBM_Buffered = 0,
    kBM_Aborting = 1,
    kBM_BlendingLow = 2,
    kBM_BlendingPrevious = 3,
    kBM_BlendingNext = 4,
    kBM_BlendingHigh = 5
} MotionBufferMode;
```

## Description

Refer to section 6.1.4.

## Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

## Corresponding name of other API environment

| | |
|---|---|
| C# | MotionBufferMode |
| LabVIEW | -- |
| Python | MotionBufferMode |

## 20.4 MotionTransitionMode

### Definition

Transition modes between adjacent coordinate motion segments

### Syntax

```
typedef enum {
    kTM_NONE = 0,
    kTM_StartVelocity = 1,
    kTM_ConstantVelocity = 2,
    kTM_CornerDistance = 3,
    kTM_MaxCornerDeviation = 4,
    kTM_PLCOpenReserved_05 = 5,
    kTM_PLCOpenReserved_06 = 6,
    kTM_PLCOpenReserved_07 = 7,
    kTM_PLCOpenReserved_08 = 8,
    kTM_PLCOpenReserved_09 = 9
} MotionTransitionMode;
```

### Description

Refer to section 6.1.5.

### Requirement

| | |
|---|---|
| Minimum supported version | iA Studio 0.23 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

### Corresponding name of other API environment

| | |
|---|---|
| C# | MotionTransitionMode |
| LabVIEW | -- |
| Python | MotionTransitionMode |

# 20.5 ShaperMode

**Definition**

Filter modes of input shaping filter (InShape)

**Syntax**

```
typedef enum {
    Shaper_Normal = 0,
    Shaper_Robust
} ShaperMode;
```

**Member**

| | |
|---|---|
| Shaper_Normal | Normal input shaping filter. |
| Shaper_Robust | Robust input shaping filter. |

**Requirement**

| | |
|---|---|
| Minimum supported version | iA Studio 1.1 |
| Header | HIMC_API.h |
| Library | HIMC_API.lib |
| DLL | HIMC_API.dll |

**Corresponding name of other API environment**

| | |
|---|---|
| C# | ShaperMode |
| LabVIEW | -- |
| Python | ShaperMode |

(This page is intentionally left blank.)

# 21. Appendix

21. Appendix ..... 21-1
21.1 API error codes ..... 21-2

# 21.1 API error codes

The following error codes appear when accessing the controller by API.

Table 21.1.1

| API Error Codes | | |
|---|---|---|
| Error Code | Error Name | Description |
| 0x01000000 | eERR_API_COMM_ERR | An error occurred when communicating with the controller. |
| 0x0100000a | eERR_API_CONNECT_FAIL | Cannot connect to controller. |
| 0x01000014 | eERR_API_TOUT | This operation returned because the time-out period expired. |
| 0x0100001e | eERR_API_ACCESS_REJECT | The request was rejected. |
| 0x01000028 | eERR_API_FIFO_MISMATCH | Fatal API error. |
| 0x01000032 | eERR_API_FIFO_FULL | The network is busy. |
| 0x0100003c | eERR_API_HIMC_NOT_READY | The himc is not ready. |
| 0x01000046 | eERR_API_PROTOCOL_MISMATCH | Fatal API error. |
| 0x01000050 | eERR_API_INPUT_ARG_ERR | The arguments are invalid. |
| 0x0100005a | eERR_API_NOT_SUPPORT | The API is not supported for this version. |
| 0x01000064 | eERR_API_BUSY | The API is busy. |
| 0x0100006e | eERR_API_FILE_TRANS_FAIL | The file transmission failed. |
| 0x01000078 | eERR_API_ID_NOT_FOUND | The connection ID was not found, maybe not connected yet. |
| 0x01000082 | eERR_API_SLV_DB_NOT_READY | The slaves are not ready. |
| 0x0100008c | eERR_API_SLV_ID_INVALID | The slave ID is invalid. |
| 0x01000096 | eERR_API_INVALID_VAR_ID | The variable ID is invalid. |
| 0x010000a0 | eERR_API_VAR_VAL_OUT_OF_RANGE | The value is out of range. |
| 0x010000a5 | eERR_API_VAR_IS_READ_ONLY | The variable is read only. |
| 0x010000aa | eERR_API_FS_ACCESS_DENIED | Unable to access file system, please check your permission. |
| 0x010000b4 | eERR_API_TASK_ID_INVALID | The task ID is invalid. |
| 0x010000be | eERR_API_TASK_EMPTY | The task is empty. |
| 0x010000c3 | eERR_API_TASK_FUNC_NOT_FOUND | Cannot find the function. |
| 0x010000c8 | eERR_API_TASK_NOT_RUNNING | The task is not running. |
| 0x010000d2 | eERR_API_TASK_IS_RUNNING | The task is already running. |
| 0x010000d7 | eERR_API_TOO_MANY_BRK_POINT | There are too many break points in the task. |
| 0x010000dc | eERR_API_INVALID_ERROR_ID | The error ID is invalid. |
| 0x010000e6 | eERR_API_INSUFFICIENT_BUFFER | Insufficient buffer. |
| 0x010000f0 | eERR_API_STR_TOO_LONG | String length is out of range. |
| 0x000000fa | eERR_API_HIMC_VERSION_MISMATCH | The API is not compatible with this controller version. |
| 0x010003e8 | eERR_API_MOTION_ERROR | Motion control error. Please check error log. |
| 0x0100270f | eERR_API_FATAL | Fatal API error. |