

[Instruction Set](#) > [Multi-Axis Coordinated Motion Instructions](#) > Motion Calculate Transform Position with Orientation (MCTPO)

Motion Calculate Transform Position with Orientation (MCTPO)

This information applies to the Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, and GuardLogix 5580 controllers.

Use the MCTPO instruction to calculate the position of a point in one coordinate system to the equivalent point in a second coordinate system.

Available Languages

Ladder Diagram



Function Block


This instruction is not available in function block.

Structured Text

MCTPO(CartesianSystem, RobotSystem, MotionControl, WorkFrame, ToolFrame, Direction, ReferencePosition, TransformPosition, RobotConfiguration, TurnsCounter);

Operands

Important: Do not use the same tag name for more than one instruction in the same program. Do not write to any instruction output tag under any circumstances.

 **ATTENTION:** If instruction operands are changed while in Run mode, the pending edits must be accepted and the controller mode cycled from Program to Run for the changes to take effect.

Configuration

The following table provides the operands used to configure the instruction. These operands cannot be changed at runtime.

Operand	Data Type	Format	Description
Cartesian System	COORDINATE_SYSTEM	Tag	Cartesian coordinate system used to program the moves.
Robot System	COORDINATE_SYSTEM	Tag	Non-Cartesian coordinate system that controls the actual equipment.
Motion Control	MOTION_INSTRUCTION	Tag	The control tag for the instruction



- ▷ [Quick Start Steps](#)
- ▷ [Logix Designer](#)
- ▷ [Module Information](#)
- ◀ [Instruction Set](#)
 - [Logix 5000 Controllers Instruction and Application Considerations](#)
 - [Logix Designer Application Instruction Set](#)
 - [Interpret the Attribute Tables](#)
 - [Array Concepts](#)
- ▷ [CIP Axis Attributes](#)
- ▷ [Module Configuration Attributes](#)
 - [Bit Addressing](#)
 - [Common Attributes](#)
 - [Data Conversions](#)
 - [Elementary data types](#)
 - [LINT data types](#)
 - [Floating Point Values](#)
 - [Immediate values](#)
 - [Index Through Arrays](#)
 - [Math Status Flags](#)
 - [Motion Error Codes \(.ERR\)](#)
 - [Structures](#)
- ▷ [Equipment Sequence instructions](#)
- ▷ [Equipment Phase Instructions](#)
- ▷ [Alarm Instructions](#)
- ▷ [Advanced Math Instructions](#)
- ▷ [Array \(File\)/Misc Instructions](#)
- ▷ [Array \(File\)/Shift Instructions](#)
- ▷ [ASCII Conversion Instructions](#)
- ▷ [ASCII Serial Port Instructions](#)
- ▷ [ASCII String Instructions](#)
- ▷ [Bit Instructions](#)
- ▷ [Compare Instructions](#)
- ▷ [Debug Instructions](#)
- ▷ [Drives Instructions](#)
- ▷ [Drive Safety Instructions](#)
- ▷ [For/Break Instructions](#)
- ▷ [Filter Instructions](#)
- ▷ [Function Block Attributes](#)
- ▷ [Structured Text Attributes](#)
- ▷ [Compute/Math Instructions](#)
- ▷ [Move/Logical Instructions](#)
- ▷ [Input/Output Instructions](#)
- ▷ [License Instructions](#)
- ▷ [Math Conversion Instructions](#)
- ▷ [Metal Form Instructions](#)
- ▷ [Motion Configuration Instructions](#)
- ▷ [Motion Event Instructions](#)
- ▷ [Motion Group Instructions](#)
- ▷ [Motion Move Instructions](#)
- ▷ [Motion State Instructions](#)

Work Frame	POSITION_DATA	Immediate Tag	<p>Work Frame offsets are the offsets used to locate the user Work frame of the Robot relative to the origin of the Robot base frame. These offsets consist of an XYZ and RxRyRz value. This allows the programs to be written in the user work space or world frame and transformed to the Robot base frame.</p> <p>To rotate the target position around the X, Y, or Z axis or offset the target position along the X, Y, or Z axis of the Robot base coordinate system, enter the degrees of rotation into the Rx, Ry, and Rz tag members in units of degrees of rotation. Enter the offset distances into X, Y, and Z tag members in coordination units. Set the ID member to a value greater than or equal to zero.</p> <p>To maintain the work frame at the robot base frame, leave structured values at zero, or set the operand tag value to zero.</p>
Tool Frame	POSITION_DATA	Immediate Tag	<p>Tool Center Point (TCP) offsets are the offsets used to locate the tool center relative to the center of the End of Arm. These offsets consist of an XYZ and RxRyRz value.</p> <p>To have the target position account for an attached tool having translation and/or orientation offsets, enter the tool offset distances into X, Y, and Z tag members in coordination units. Enter the degrees of tool rotation into the Rx, Ry, and Rz tag members in units of degrees of rotation. Set the ID member to a value greater than or equal to zero.</p> <p>To have the target position reflect only the point at the end-of-arm, leave the structure values at zero, or set the operand tag value to zero.</p>

For further information of configuring coordinate systems refer to the Motion Coordinate System User Manual publication MOTION-UM002.

Affects Math Status Flags

No

Major/Minor Faults

None specific to this instruction. See Index Through Arrays for array-indexing faults.

Execution

Ladder Diagram

Condition/State	Action Taken
Prescan	Same as Rung-condition-in is false.
Rung-condition-in is false	The .EN, .DN, .ER are cleared to false.
Rung-condition-in is true and .EN bit is false	The .EN bit is set to true and the instruction executes.
Rung-condition-in is true and .EN bit is true	N/A

- ◀ [Multi-Axis Coordinated Motion Instructions](#)
- [Master Driven Coordinated Control \(MDCC\)](#)
- [Motion Calculate Transform Position \(MCTP\)](#)
- [Motion Coordinated Change Dynamics \(MCCD\)](#)
- [Motion Coordinated Circular Move \(MCCM\)](#)
- [Motion Coordinated Transform with Orientation \(MCTO\)](#)
- [Motion Coordinated Path Move \(MCPM\)](#)
- [Motion Calculate Transform Position with Orientation \(MCTPO\)](#)
- [Motion Coordinated Linear Move \(MCLM\)](#)
- [Motion Coordinated Shutdown \(MCSD\)](#)
- [Motion Coordinated Shutdown Reset \(MCSR\)](#)
- [Motion Coordinated Stop \(MCS\)](#)
- [Motion Coordinated Transform \(MCT\)](#)
- [Speed, acceleration, deceleration, and jerk enumerations for coordinated motion](#)
- [Status Bits for Motion Instructions \(MCLM, MCCM\) when MDCC Is Active](#)
- [Change between master driven and time driven modes for Coordinated Motion instructions](#)
- [Choose a Termination Type](#)
- [Common Action Table for Slave Coordinate System and Master Axis](#)
- [Input and Output Parameters Structure for Coordinate System Motion Instructions](#)
- [Returned Calculated Data Parameter for Coordinated System Motion Instruction](#)
- ▷ [Logical and Move Instructions](#)
- ▷ [Program Control Instructions](#)
- ▷ [Sequencer Instructions](#)
- ▷ [Special Instructions](#)
- ▷ [Timer and Counter Instructions](#)
- ▷ [Trigonometric Instructions](#)
- ▷ [Process Control Instructions](#)
- ▷ [Select/Limit Instructions](#)
- ▷ [Sequential Function Chart \(SFC\) Instructions](#)
- ▷ [Statistical Instructions](#)

Postscan	Same as Rung-condition-in is false.
----------	-------------------------------------

▷ [Safety Instructions](#)

▷ [Studio 5000 Logix Designer Glossary](#)

Structured Text

Condition/State	Action Taken
Prescan	See Prescan in the Ladder Diagram table.
Normal execution	See Rung-condition-in is false, followed by Rung-condition-in is true in the Ladder Diagram table.
Postscan	See Postscan in the Ladder Diagram table.

Inputs

Operand	Data Type	Format	Description
Transform Direction	DINT	Tag	To calculate Cartesian position, select Forward(0). To calculate Robot joint positions, select Inverse(1).
Reference Position	REAL[6]	Tag	If the transform direction is forward, enter an array that has joint angles. If the transform direction is inverse, enter an array that has Cartesian positions.
Robot Configuration	DINT	Tag	If TD is Inverse, create a tag to represent the specified robot Configuration. If the TD is forward, the system returns the robot Configuration defined in the same format. Bit 0 of robot configuration will be a don't care, for both TD=Inverse and TD=Forward. List of bit values: Bit0 – Robot Configuration Change(1)/Same(0) Bit1 – Lefty(1)/Righty(0) Bit2 – Above(1)/Below(0) Bit3 – Flip(1)/No flip(0) Robot Configuration must be zero (0) for Delta robot geometries.
Robot Turns Counters	INT16[4]	Tag	A counter indicating the number of times the 180 degree point is crossed in the +/- direction that the respective axis has turned. If Transform Direction is Forward, the value is computed by the system and returned via this tag. If Transform Direction is Inverse, this value determines final joint positions. The following are array index assignment for the joint axes turns count: Index 0 - J1 axis Index 1 - J4 axis Index 2 - J6 axis Index 3 - Reserved

Outputs

Operand	Data Type	Format	Description
Transform Position	REAL[6]	Tag	Array that stores the calculated position.
Robot Turns Counters	INT16[4]	Tag	If Transform Direction is Forward, this value is computed by the system and returned via this tag.
Robot Configuration	DINT	Tag	<div>If Transform Direction is Forward, then this value is computed by the system and returned via this tag.</div> <div>For Delta Robot geometries, robot configuration output is always zero.</div>

Fault Codes

For the Motion Coordinated Transform with Orientation (MCTO) instruction, an error will occur at runtime if invalid operand values are provided.

Extended Error Codes

Extended Error codes provide additional instruction-specific information for the Error Codes that are generic to other instructions. See Motion Error Codes (ERR) for Motion Instructions. Extended Error Codes meaning depends on the corresponding Error Codes.

Error Code	EX_ERROR Code	Description
61	1	<div>Connection Conflict</div> <div>Transform Motion Group Error</div> <div>Cartesian or Robot coordinate system is ungrouped or associated to a different motion group from the other.</div>
61	2	<div>Connection Conflict</div> <div>Transform Duplicate Systems Error</div> <div>Operand 0 and Operand 1 are the same coordinate system type</div>
61	3	<div>Connection Conflict</div> <div>Transform Source Dimension Error</div> <div>Transform dimension of the Cartesian coordinate system is < 2.</div>
61	4	<div>Connection Conflict</div> <div>Transform Target Dimension Error</div> <div>Robot coordinate system Transform Dimension is equal to zero.</div>
61	9	<div>Connection Conflict</div> <div>Transform Axes Overlap Error</div> <div>An axis is a member of both the Cartesian and Robot systems.</div>
61	12	<div>Connection Conflict</div> <div>Transform Invalid Link Length</div> <div>Link length values for any robot geometry must be ≥0.0 units.</div>

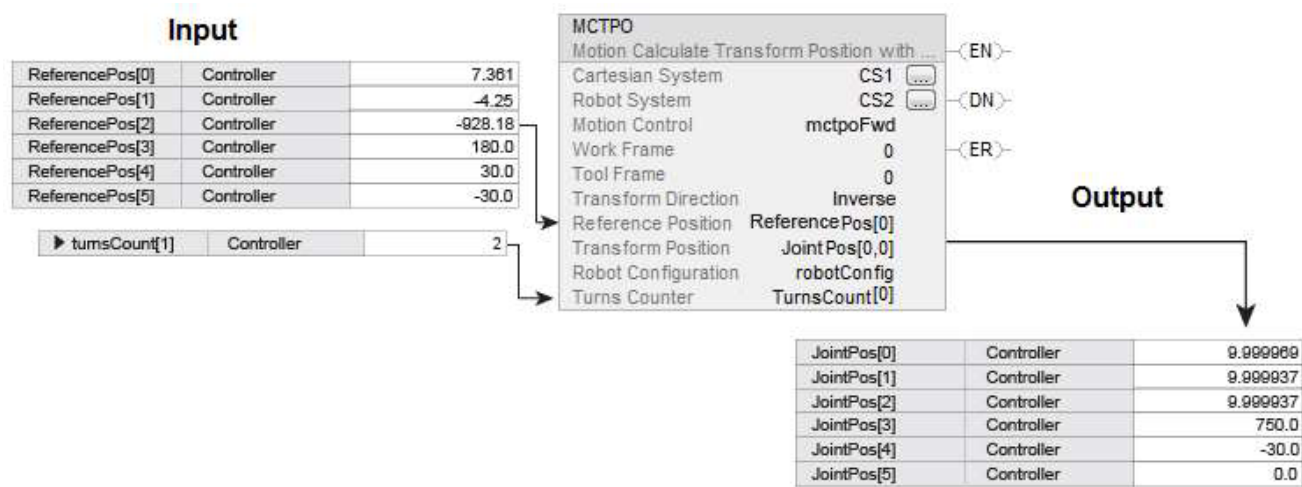
61	15	Connection Conflict Transform Invalid Delta Configuration End Effector Offset Re must not be negative Link length 1 + Rb – Re must be less than link length 2 (Link length 1 + Rb – Re) must be positive or greater than zero
13	3	Value Out Of Range (base angle) Any orientation angle > 360 or < -360.
13	3	Value Out Of Range (base ID) ID equals -1.
13	4	Value Out Of Range (tool angle) Any orientation angle > 360 or < -360.
13	4	Value Out Of Range (tool ID) ID equals -1.
13	5	Parameter Out Of Range Transform Direction.
13	6	Parameter Out Of Range Reference pos array end < 6
13	6	Parameter Out Of Range Input position Rx, Ry, or Rz exists and value >± 180 degrees
13	6	Parameter Out Of Range Operand 6 If robot geometry is Delta J1J2J6 or Delta J1J2J3J6 and if input position Rx exists and is not equal to 180°. If robot geometry is Delta J1J2J6 or Delta J1J2J3J6 and if input position Ry is not equal to 0°. If robot geometry is Delta J1J2J3J4J5 and if input position Rx exists and value is not equal to 0° or 180°.
13	7	Parameter Out Of Range Transform pos array end < 6
13	9	Parameter Out Of Range Any turns counter provided for the inverse transform exceeds 127 or -127 turns.
61	15	Connection Conflict Transform Invalid Delta Configuration Link Length1 must not be equal to LinkLength2 End Effector Offset1 Re must not be negative For Delta J1J2J6 and Delta J1J2J3J6 End Effector Offset3(D3) must not be negative (Link length 1 + Rb - Re) must be less than link length 2 (Link length 1 + Rb – Re) must be positive or greater than zero

67	1	<div>Invalid Transform Position</div> <div>MOP Invalid Rx orientation</div> <div>Invalid Rx orientation value at EOA transformations. For 4 axis geometries (Delta, SCARA, Articulated Dependent) only Rx = 180 deg position is allowed. For all other positions, it will generate error.</div>
67	2	<div>Invalid Transform Position</div> <div>MOP Invalid Ry orientation</div> <div>Invalid Ry orientation value at EOA transformations. For 4 axis geometries, after removing work frame and tool frame, there should not be any Ry orientation values at EOA.</div>
67	3	<div>Invalid Transform Position</div> <div>MOP Invalid Ry orientation</div>
147	3 - 5	<div>Invalid Orientation Scaling Constant</div> <div>Extended error code 3 : Rx</div> <div>Extended error code 4 : Ry</div> <div>Extended error code 5 : Rz</div> <div>Orientation axis</div> <div>has scaling constant $> \text{MAX_K_CONSTANT_FOR_ORIENTATION_AXIS}$. or</div> <div>has scaling constant which is not Integer or</div> <div>has Conversion ratio between Coordination Units and Position Units other than 1:1.</div>
148	3 -5	<div>MCPM Orientation Offset Not Zero</div> <div>Rx orientation offset not valid.</div> <div>Extended error code 3 : Rx orientation offset not valid.</div> <div>Extended error code 4 : Ry orientation offset not valid.</div> <div>Extended error code 5 : Rz orientation offset not valid.</div> <div>If robot Geometry is (MO_CD_J1J2J6 or MO_CD_J1J2J3J6) and:<ul style="list-style-type: none">workframe offset Rx isn't 0 orworkframe offset Ry isn't 0 ortoolframe offset Rx isn't 0 ortoolframe offset Ry isn't 0Or if robot Geometry is MO_CD_J1J2J3J4J5 and:<ul style="list-style-type: none">workframe offset Rx isn't 0 orworkframe offset Ry isn't 0 ortoolframe offset Rx isn't 0 ortoolframe offset Rz isn't 0</div>
151	4	<div>Joint Angle Beyond Limits</div> <div>This indicates the error condition when the Joint 4 in a 5 axis Delta goes beyond turns counter range limit ($45899.99 < J4 < -45900$).</div> <div>Ext Error 4 : Joint J4 Beyond Limit</div>
151	5	<div>Joint Angle Beyond Limits</div> <div>This indicates error condition when the Joint 5 in a 5 axis Delta goes beyond ± 179, $+179 > J5 < -179$</div> <div>Ext Error 5: Joint J5 Beyond Limit</div>

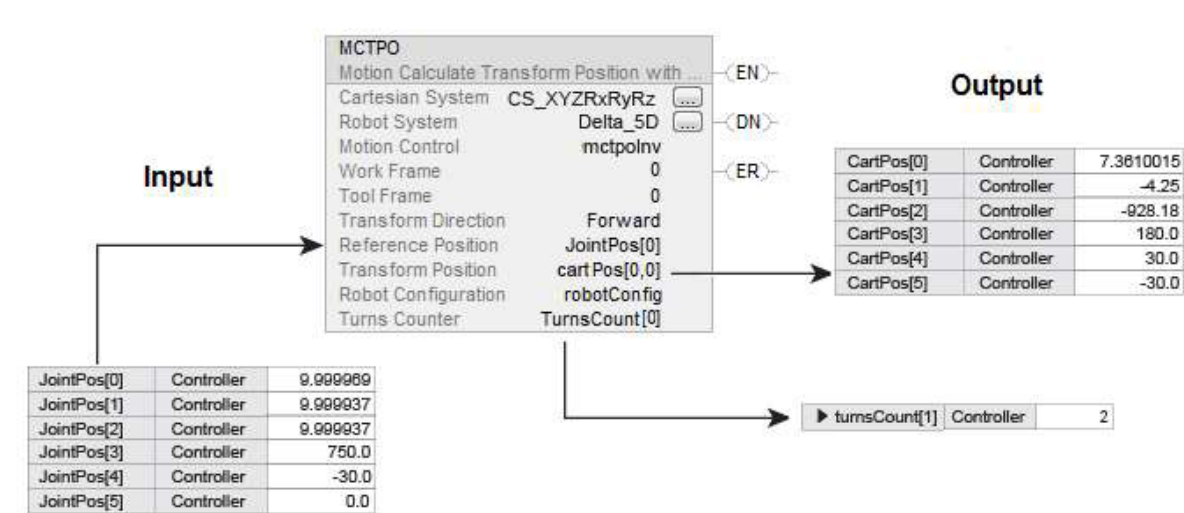
151	6	Joint Angle Beyond Limits This indicates the error condition when the Joint 6 in a 4 axis Delta goes beyond turns counter range limit ($45899.99 < J6 < -45900$) Ext Error 6 : Joint J6 Beyond Limit
153	1	Invalid Translation Position MOP Invalid X Translation Translation on X axis is Invalid
153	2	Invalid Translation Position MOP Invalid Y Translation Translation on Y axis is Invalid
153	3	Invalid Translation Position MOP Invalid Z Translation Translation on Z axis is Invalid

Example Ladder Diagram

This example illustrates an MCTPO instruction with Transform Direction as Inverse, where the user feeds Cartesian positions and turns counter as input. The instruction computes the corresponding target joint angle positions and is written to the Transform Position parameter as the output.



This example illustrates the MCTPO instruction with Transform Direction as Forward. The target positions are guided into the Reference position operand as input. The instruction computes the corresponding Cartesian positions and turns counter as the output.



Structured Text

MCTPO(CS1, CS2, MCTPo1[0], BaseFrame, ToolFrame, Forward, refPos[0], transPos[0], robotConfig[0], TurnsCounter[0]);

Tip: For further information on creating geometries with orientation support, see the [Motion Coordinate System User Manual](#) publication [MOTION-UM002](#).

See also

[Structured Text Syntax](#)

[Index Through Arrays](#)

[Motion Error Codes \(.ERR\)](#)

[Multi-Axis Coordinated Motion Instructions](#)

[Define coordinate system frames](#)