

Manual de aplicaciones MultiMove

Trace back information:
Workspace R15-2 version a14
Checked in 2015-10-07
Skribenta version 4.6.081

Manual de aplicaciones

MultiMove

RobotWare 6.02

ID de documento: 3HAC050961-005

Revisión: B

La información de este manual puede cambiar sin previo aviso y no puede entenderse como un compromiso por parte de ABB. ABB no se hace responsable de ningún error que pueda aparecer en este manual.

Excepto en los casos en que se indica expresamente en este manual, ninguna parte del mismo debe entenderse como una garantía por parte de ABB por las pérdidas, lesiones, daños materiales, idoneidad para un fin determinado ni garantías similares.

ABB no será en ningún caso responsable de los daños accidentales o consecuentes que se produzcan como consecuencia del uso de este manual o de los productos descritos en el mismo.

Se prohíbe la reproducción o la copia de este manual o cualquiera de sus partes si no se cuenta con una autorización escrita de ABB.

Usted puede obtener copias adicionales de este manual a través de ABB.

El idioma original de esta publicación es el inglés. Cualquier otro idioma suministrado ha sido traducido del inglés.

© Copyright 2004-2008, 2015 ABB. Reservados todos los derechos.

ABB AB
Robotics Products
Se-721 68 Västerås
Suecia

Contenido

Descripción general de este manual	7
Documentación del producto, IRC5	9
Seguridad	11
1 Introducción	13
1.1 Acerca de MultiMove	13
1.2 Terminología	15
1.3 Aplicaciones de ejemplo	16
1.3.1 Acerca de las aplicaciones de ejemplo	16
1.3.2 Ejemplo "UnsyncArc"	17
1.3.3 Ejemplo "SyncArc"	18
2 Instalación	19
2.1 Instalación del hardware	19
2.1.1 Acerca de la instalación del hardware	19
2.1.2 Conexiones del Control Module	20
2.1.3 Conexiones del Drive Module	25
2.2 Instalación del software	27
2.2.1 Instalación del software	27
3 Configuración	29
3.1 Descripción general de la configuración	29
3.2 Parámetros del sistema	30
3.2.1 Tema Controller	30
3.2.2 Tema Motion	32
3.2.3 Tema I/O	34
3.3 Ejemplos de configuración	35
3.3.1 Ejemplo de configuración para "UnsyncArc"	35
3.3.2 Ejemplo de configuración para "SyncArc"	37
3.3.3 Ejemplo de configuración de E/S	39
4 Calibración	41
4.1 Descripción general de la calibración	41
4.2 Calibración relativa	42
4.3 Cadenas de calibración	44
4.4 Ejemplos de sistemas de coordenadas	45
4.4.1 Ejemplo "UnsyncArc"	45
4.4.2 Ejemplo "SyncArc"	47
5 Interfaz de usuario específica de MultiMove	49
5.1 FlexPendant para el sistema MultiMove	49
5.2 Indicaciones en la barra de estado	50
5.3 Cómo abrir el Editor de programas	52
5.4 Ventana Producción	53
5.5 Menú de unidades mecánicas	54
5.6 Selección de qué tareas se inician con el botón INICIO	55
6 Programación	59
6.1 Componentes de RAPID	59
6.2 Tareas y técnicas de programación	62
6.3 Objetos de trabajo coordinados	63
6.4 Movimientos independientes	64
6.4.1 Acerca de los movimientos independientes	64
6.4.2 Ejemplo "UnsyncArc" con movimientos independientes	65

6.5	Movimientos semicoordinados	67
6.5.1	Acerca de los movimientos semicoordinados	67
6.5.2	Ejemplo "SyncArc" con movimientos semicoordinados	68
6.5.3	Consideraciones y limitaciones al utilizar movimientos semicordinados	73
6.6	Movimientos coordinados sincronizados	75
6.6.1	Acerca de los movimientos coordinados sincronizados	75
6.6.2	Ejemplo "SyncArc" con movimientos coordinados	76
6.7	Ejecución de programas	79
6.7.1	Zonas de esquina	79
6.7.2	Comportamiento de la sincronización	81
6.7.3	Instrucciones vacías	83
6.7.4	Principios de movimiento	84
6.7.5	Modificación de posiciones	85
6.7.6	Movimiento del puntero de programa	86
6.7.7	Orientación de la herramienta en los movimientos circulares	87
6.7.8	Aplicaciones afectadas por MultiMove	88
6.8	Recomendaciones de programación	89
6.8.1	Recomendaciones de programación	89
7	Recuperación en caso de errores de RAPID	91
7.1	Recuperación en caso de errores para MultiMove	91
7.2	Ejemplo sencillo de recuperación	92
7.3	Errores generados asíncronamente	93
7.4	Ejemplo de creación de un error generado asíncronamente	95
7.5	Ejemplo con movimientos en el gestor de errores	97
8	Ejecución de un subconjunto del sistema MultiMove	99
8.1	Cómo continuar con una o varias unidades de accionamiento inactivas.	99
8.2	Ejecución de un subconjunto en los ejemplos "Unsync Arc"	102
Índice		105

Descripción general de este manual

Acerca de este manual

Este manual contiene información acerca de las opciones de RobotWareMultiMove Independent y MultiMove Coordinated. Esta última incluye determinadas funciones ampliadas. Mientras no se especifique lo contrario, el nombre MultiMove se refiere a ambas opciones.

Utilización

Puede usar este manual como una descripción breve para determinar si MultiMove es la opción adecuada para solucionar un problema determinado o como una descripción para el uso del sistema. Este manual proporciona información sobre los parámetros del sistema y los componentes de RAPID relacionados con MultiMove y contiene muchos ejemplos de cómo usarlos. Los detalles en cuanto a la sintaxis de los componentes de RAPID y elementos similares no se describen en este manual, pero pueden consultarse en sus respectivos manuales de referencia.

¿A quién va destinado este manual?

Este manual está dirigido principalmente a programadores de robots.

Requisitos previos

El lector deberá...

- Estar familiarizado con los robots industriales y su terminología.
- Estar familiarizado con el lenguaje de programación RAPID.
- Estar familiarizado con los parámetros del sistema y cómo configurarlos.
- Estar familiarizado con la opción Multitasking (consulte el manual *Application manual - Engineering tools*).

Referencias

Referencia	ID de documento
<i>Manual de referencia técnica - Descripción general de RAPID</i>	3HAC050947-005
<i>Manual de referencia técnica - Instrucciones, funciones y tipos de datos de RAPID</i>	3HAC050917-005
<i>Technical reference manual - RAPID kernel</i>	3HAC050946-001
<i>Manual del operador - IRC5 con FlexPendant</i>	3HAC050941-005
<i>Manual del operador - RobotStudio</i>	3HAC032104-005
<i>Manual del producto - IRC5</i>	3HAC021313-005
Manual de referencia técnica - Parámetros del sistema	3HAC17076-1
Application manual - Engineering tools	3HAC020434-001
Application manual - Motion functions and events	3HAC18152-1
<i>Application manual - Arc and Arc Sensor</i>	3HAC050988-001
<i>Application manual - Spot options</i>	3HAC050979-001

Continúa en la página siguiente

Revisiones

Revisión	Descripción
-	Publicado con RobotWare 6.0.
A	Publicado con RobotWare 6.01. <ul style="list-style-type: none">• Añadida información acerca del switch de Ethernet; consulte Conexiones Ethernet en la página 21.
B	Publicado con RobotWare 6.02. Sección Creación de un sistema MultiMove en la página 27 actualizada para usar el Administrador de instalación.

Documentación del producto, IRC5

Categorías de documentación de usuario de ABB Robotics

La documentación de usuario de ABB Robotics está dividida en varias categorías. Esta lista se basa en el tipo de información contenida en los documentos, independientemente de si los productos son estándar u opcionales.

Puede pedir a ABB en un DVD todos los documentos enumerados. Los documentos enumerados son válidos para los sistemas de robot IRC5.

Manuales de productos

Los manipuladores, los controladores, el DressPack/SpotPack y la mayoría de demás equipos se entregan con un **Manual del producto** que por lo general contiene:

- Información de seguridad
 - Instalación y puesta en servicio (descripciones de la instalación mecánica o las conexiones eléctricas).
 - Mantenimiento (descripciones de todos los procedimientos de mantenimiento preventivo necesarios, incluidos sus intervalos y la vida útil esperada de los componentes).
 - Reparaciones (descripciones de todos los procedimientos de reparación recomendados, incluidos los repuestos)
 - Calibración.
 - Retirada del servicio.
 - Información de referencia (normas de seguridad, conversiones de unidades, uniones con tornillos, listas de herramientas).
 - Lista de repuestos con vistas ampliadas (o referencias a listas de repuestos separadas).
 - Diagramas de circuitos (o referencias a diagramas de circuitos).
-

Manuales de referencia técnica

Los manuales de referencia técnica describen la información de referencia relativa a los productos de robótica.

- *Technical reference manual - Lubrication in gearboxes*: descripción de los tipos y volúmenes de lubricación de las cajas reductoras del manipulador.
- *Manual de referencia técnica - Descripción general de RAPID*: una descripción general del lenguaje de programación RAPID.
- *Manual de referencia técnica - Instrucciones, funciones y tipos de datos de RAPID*: descripción y sintaxis de todos los tipos de datos, instrucciones y funciones de RAPID.
- *Technical reference manual - RAPID kernel*: una descripción formal del lenguaje de programación RAPID.
- *Manual de referencia técnica - Parámetros del sistema*: una descripción de los parámetros del sistema y los flujos de trabajo de configuración.

Continúa en la página siguiente

Manuales de aplicaciones

Las aplicaciones específicas (por ejemplo opciones de software o hardware) se describen en **Manuales de aplicaciones**. Cada manual de aplicaciones puede describir una o varias aplicaciones.

Generalmente, un manual de aplicaciones contiene información sobre:

- Finalidad de la aplicación (para qué sirve y en qué situaciones resulta útil)
- Contenido (por ejemplo cables, tarjetas de E/S, instrucciones de RAPID, parámetros del sistema, DVD con software para PC)
- Forma de instalar el hardware incluido o necesario.
- Forma de uso de la aplicación.
- Ejemplos sobre cómo usar la aplicación.

Manuales del operador

Los manuales del operador describen el manejo de los productos desde un punto de vista práctico. Estos manuales están orientados a las personas que van a tener contacto de uso directo con el producto, es decir, operadores de células de producción, programadores y técnicos de resolución de problemas.

El grupo de manuales se compone de (entre otros documentos):

- *Manual del operador - Información de seguridad para emergencias*
- *Manual del operador - Información general de seguridad*
- *Manual del operador - Procedimientos iniciales - IRC5 y RobotStudio*
- *Manual del operador - Introducción a RAPID*
- *Manual del operador - IRC5 con FlexPendant*
- *Manual del operador - RobotStudio*
- *Manual del operador - Resolución de problemas del IRC5, para el controlador y el manipulador.*

Seguridad

Seguridad del personal

Los robots son pesados y tienen una fuerza extraordinaria independientemente de su velocidad. Una pausa o una parada larga en un movimiento puede ir seguida de un movimiento rápido y peligroso. Incluso si es posible predecir un patrón de movimientos, una señal externa puede disparar un cambio de funcionamiento y dar lugar a un movimiento inesperado.

Por tanto, es importante respetar toda la normativa de seguridad al entrar en un espacio protegido.

Normativa de seguridad

Antes de empezar a trabajar con el robot, asegúrese de familiarizarse con la normativa de seguridad descrita en el manual *Manual del operador - Información general de seguridad*.

Esta página se ha dejado vacía intencionadamente

1 Introducción

1.1 Acerca de MultiMove

Finalidad

La finalidad de MultiMove es permitir el manejo de varios robots con un solo controlador. De esta forma no sólo se reducen los costes de hardware, sino que permiten conseguir una coordinación avanzada entre varios robots y otras unidades mecánicas.

A continuación aparecen algunos ejemplos de aplicaciones:

- Varios robots trabajan en un mismo objeto de trabajo en movimiento.
 - Un robot mueve un objeto de trabajo mientras otros robots trabajan en él.
 - Varios robots que cooperan para elevar objetos pesados.
-

Funcionalidad incluida

MultiMove permite que hasta 6 tareas sean tareas de movimiento (tareas que contienen instrucciones de movimiento). Dado que no es posible usar más de 4 Drive Modules, cada controlador puede manejar hasta 4 robots. Sin embargo, es posible manejar ejes adicionales mediante tareas separadas, hasta un total de 6 tareas de movimientos.

Las dos opciones de MultiMove permiten implementar lo siguiente:

- Movimientos independientes (consulte [Movimientos independientes en la página 64](#))
- Movimientos semicoordinados (consulte [Movimientos semicoordinados en la página 67](#))

Además de lo ya mencionado, la opción MultiMove Coordinated permite implementar lo siguiente:

- Movimientos coordinados sincronizados (consulte [Movimientos coordinados sincronizados en la página 75](#))
-

Opciones incluidas

Si dispone de MultiMove, tiene un acceso automático a algunas opciones que se necesitan para poder utilizar MultiMove.

MultiMove incluye siempre la opción:

- Multitasking

Además de lo ya mencionado, MultiMove Coordinated incluye la opción:

- Multiple Axis Positioner
-

Enfoque básico

Éste es el enfoque general requerido para la instalación de un sistema MultiMove.

- 1 Instalar el hardware y el software (consulte [Instalación en la página 19](#)).
- 2 Configurar los parámetros del sistema (consulte [Configuración en la página 29](#)).

Continúa en la página siguiente

1 Introducción

1.1 Acerca de MultiMove

Continuación

- 3 Calibrar los sistemas de coordenadas (consulte [Calibración en la página 41](#)).
- 4 Escribir un programa de RAPID para cada tarea (consulte [Programación en la página 59](#)).

1.2 Terminología

Acerca de estos términos

Algunas palabras tienen un significado específico en cuanto a su uso en este manual. Es importante comprender con exactitud el significado de estas palabras. A continuación se enumeran las definiciones de estas palabras con respecto a este manual.

Lista de términos

Término	Explicación
Coordinación	Un robot que está coordinado con un objeto de trabajo sigue los movimientos de dicho objeto.
Sincronización	Movimientos simultáneos entre sí. La sincronización se refiere a una similitud en el tiempo, no en coordenadas espaciales.
Posicionador	Una unidad mecánica sin TCP que sólo puede gestionar movimientos de ejes. Un posicionador es una unidad mecánica dotada de uno o varios ejes y que sostiene y mueve un objeto de trabajo.
Robot	Una unidad mecánica con TCP que puede ser programada con coordenadas cartesianas (x, y, z).
Programa de tarea	Lo mismo que un programa. Es sólo una forma de especificar que se trata de un programa que realiza una tarea específica.

1 Introducción

1.3.1 Acerca de las aplicaciones de ejemplo

1.3 Aplicaciones de ejemplo

1.3.1 Acerca de las aplicaciones de ejemplo

Tres ejemplos uniformes

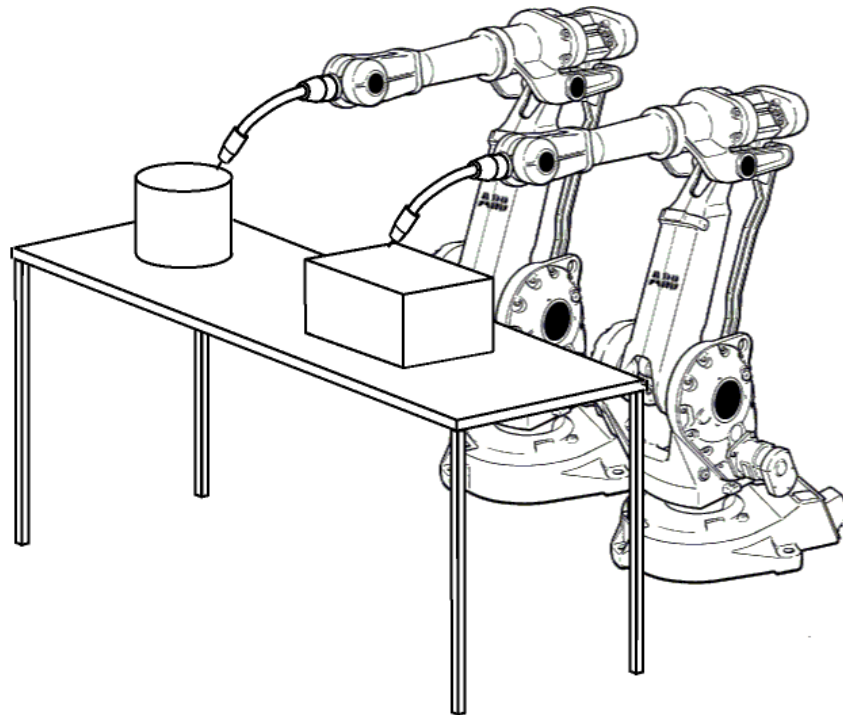
Este manual contiene muchos ejemplos (de configuración, código de RAPID, etc.). Cada uno de los ejemplos ha sido creado para uno de los tres sistemas de robot físicos. Estos ejemplos de configuración de sistema de robot se denominan "UnsyncArc", "SyncArc" y "SyncSpot" y le ayudarán a comprender para qué tipo de sistema de robot se ha hecho cada uno de ellos. Los ejemplos también son uniformes, es decir, que el ejemplo de código de RAPID de "SyncSpot" se ha creado para un sistema de robot configurado con el ejemplo de configuración "SyncSpot".

1.3.2 Ejemplo "UnsyncArc"

Acerca del ejemplo "UnsyncArc"

En este ejemplo, dos robots trabajan de forma independiente en una pieza de trabajo para cada uno de ellos. No cooperan de ninguna forma y no tienen por qué esperar el uno al otro.

Figura



xx0300000590

1 Introducción

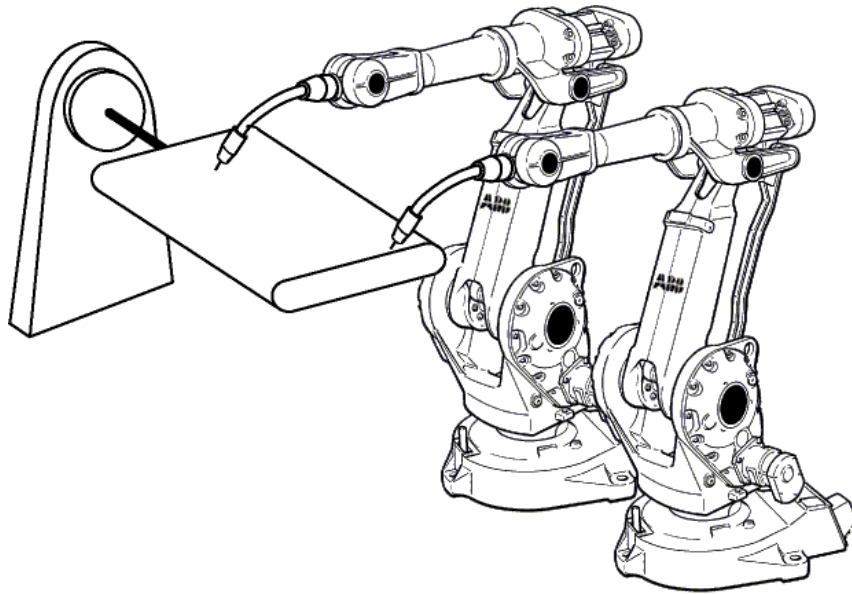
1.3.3 Ejemplo "SyncArc"

1.3.3 Ejemplo "SyncArc"

Acerca del ejemplo "SyncArc"

En este ejemplo, dos robots aplican soldaduras al arco a una misma pieza de trabajo. El objeto de trabajo se gira mediante un posicionador.

Figura



xx0300000594

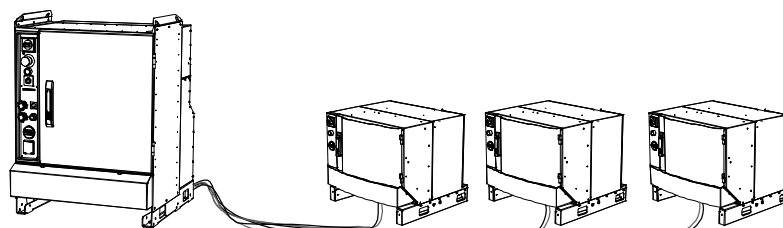
2 Instalación

2.1 Instalación del hardware

2.1.1 Acerca de la instalación del hardware

Descripción general

Los controladores que manejen varios robots requieren Drive Modules adicionales (un Drive Module por cada robot). Es posible utilizar un máximo de cuatro módulos de accionamiento, incluido el que ya se suministra asociado al módulo de control.



xx0400001042

Es necesario conectar al módulo de control un cable Ethernet y un cable de señales de seguridad por cada Drive Module adicional. Los Control Module de MultiMove están equipados con un switch de Ethernet adicional para comunicarse con los Drive Modules adicionales.

En este manual sólo describen los pasos específicos de una instalación MultiMove. Para obtener más información sobre la instalación y puesta en servicio del controlador, consulte el *Manual del producto - IRC5*.

2 Instalación

2.1.2 Conexiones del Control Module

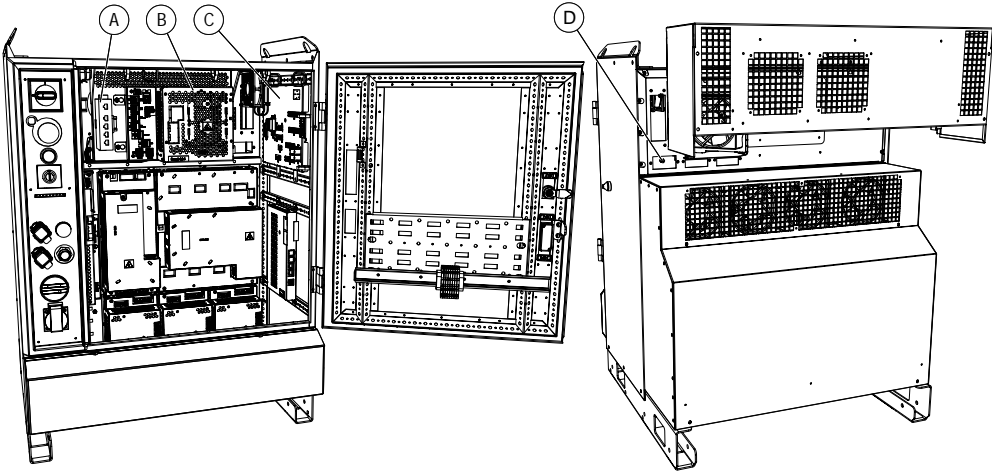
2.1.2 Conexiones del Control Module

Conexión de los Drive Module al Control Module

En el momento de la entrega, tanto el cable Ethernet como el cable de señales de seguridad están conectados al Drive Module. También están conectados a una placa protectora que se adapta a la ranura del Control Module.

Retire la cubierta de una ranura vacía y monte en su lugar la placa de protección de los cables de comunicación. Conecte el cable de Ethernet como se describe en [Conexiones Ethernet en la página 21](#) y el cable de señales de seguridad como se indica en [Conexiones de señales de seguridad en la página 23](#).

DSQC1000

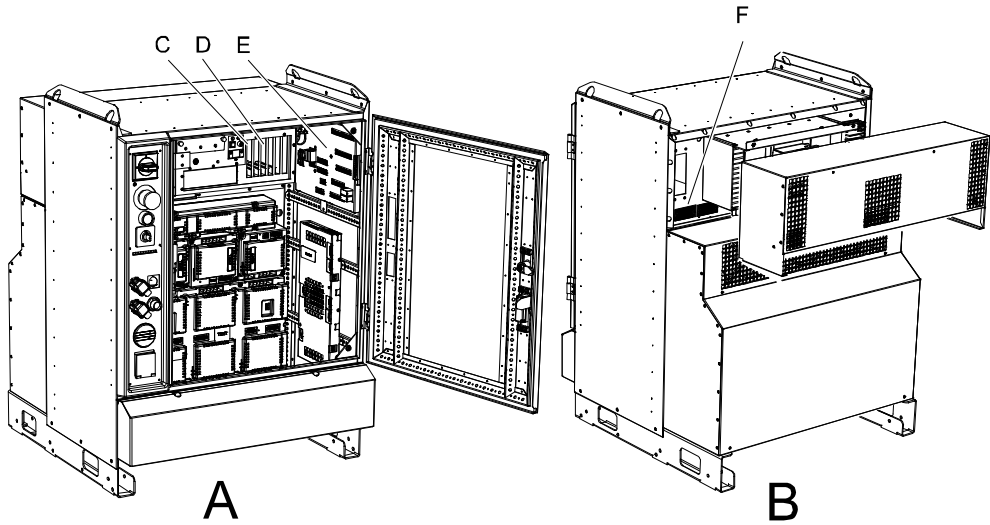


xx1400000408

A	Switch de Ethernet de MultiMove DSQC1007 (3HAC045976-001)
B	Ordenador principal DSQC1000
C	Tarjeta de panel DSQC 643
D	Ranuras para conectar los cables de comunicación al armario de control

Continúa en la página siguiente

DSQC 639




xx0600002780

A	Vista delantera del controlador single cabinet controller
B	Vista posterior del controlador single cabinet controller
C	Tarjeta de comunicación del robot
D	Tarjeta Ethernet (sólo presente si se utiliza más de un módulo de accionamiento)
E	Tarjeta de panel
F	Ranuras para conectar los cables de comunicación al armario de control

Conexiones Ethernet

Conecte los cables Ethernet de la forma mostrada en la figura siguiente:

**Nota**

Es importante conectar el Drive Module correcto a la conexión de Ethernet correspondiente. Si el orden de las conexiones de Ethernet no coincide con las selecciones realizadas en el Administrador de instalación (consulte [Creación de un sistema MultiMove en la página 27](#)), la configuración del robot no se corresponderá con el robot correcto.

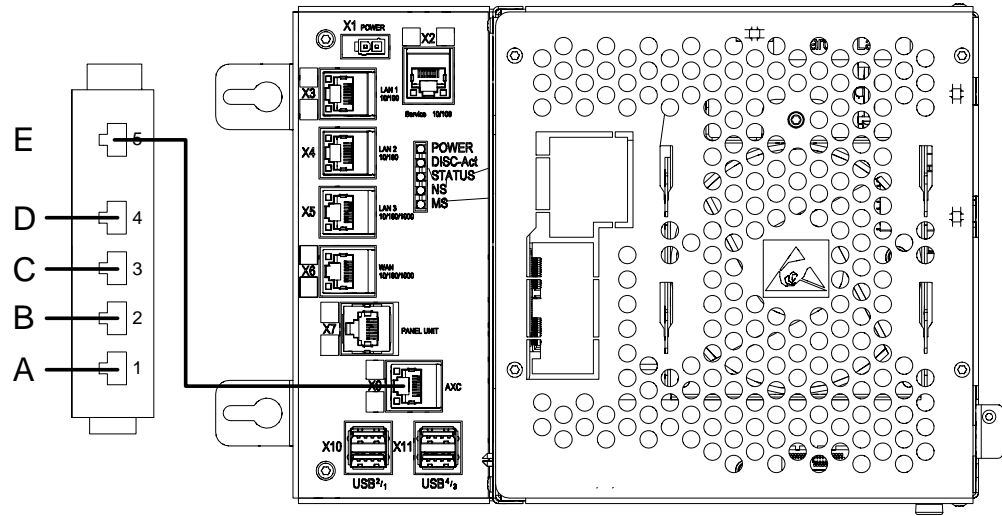
Continúa en la página siguiente

2 Instalación

2.1.2 Conexiones del Control Module

Continuación

DSQC1000



xx1400000409

A	Conexión Ethernet al módulo de accionamiento nº 1 (ya conectada en el momento de la entrega)
B	Conexión Ethernet al módulo de accionamiento nº 2
C	Conexión Ethernet al Drive Module 3
D	Conexión Ethernet al Drive Module 4
E	Conexión de Ethernet entre el switch de MultiMove y el ordenador principal (ya conectado en el momento de la entrega)

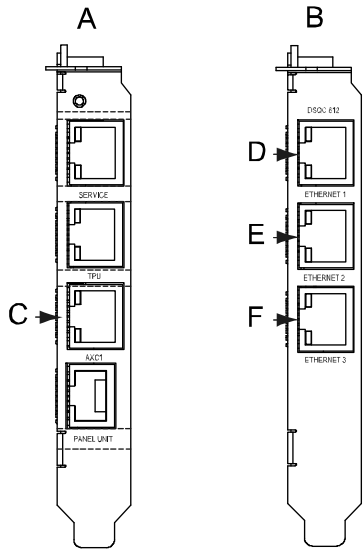


Nota

El switch de Ethernet es obligatorio para MultiMove y también al ejecutar MultiMove con un único ordenador de ejes en el sistema. Por ejemplo, al ejecutar MultiMove con un único robot junto con un posicionador o ejes adicionales.

Continúa en la página siguiente

DSQC 639

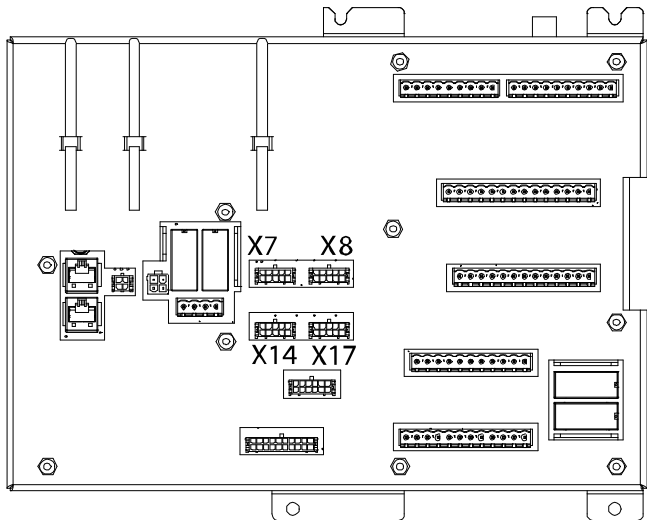


xx0400001141

A	Tarjeta de comunicación del robot
B	Tarjeta Ethernet
C	Conexión Ethernet al módulo de accionamiento nº 1 (ya conectada en el momento de la entrega)
D	Conexión Ethernet al módulo de accionamiento nº 2
E	Conexión Ethernet al Drive Module 3
F	Conexión Ethernet al Drive Module 4

Conexiones de señales de seguridad

El cable de señales de seguridad de una unidad de accionamiento se conecta a la tarjeta de panel de la forma mostrada en la figura siguiente:



xx0400001295

Continúa en la página siguiente

2 Instalación

2.1.2 Conexiones del Control Module

Continuación

X7	Conector para el cable de señales de seguridad del módulo de accionamiento nº 1 (ya conectada en el momento de la entrega)
X8	Conector para el cable de señales de seguridad del módulo de accionamiento nº 2
X14	Conector para el cable de señales de seguridad del módulo de accionamiento nº 3
X17	Conector para el cable de señales de seguridad del módulo de accionamiento nº 4

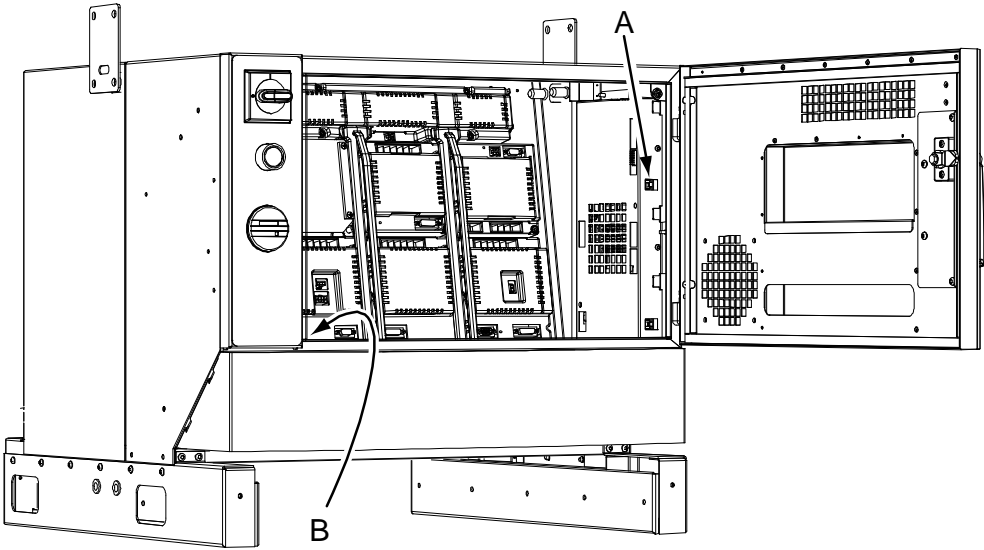
Retire el conector de puente y sustitúyalo con el cable de señales de seguridad.

2.1.3 Conexiones del Drive Module

Ya conectada en el momento de la entrega

Cuando ABB suministra un sistema MultiMove, el cable de Ethernet y el cable de señales de seguridad ya están conectados a la unidad de Drive Module. Solo necesita saber cómo están conectados estos cables si prevé cambiar la configuración del hardware o sustituir piezas.

Conecte los cables del Control Module



xx0600002787

A	Conexión a Ethernet
B	Placa de interfaz de contactor

Conecte el cable de Ethernet a la conexión Ethernet marcada como Computer module link. Conecte el cable de señales de seguridad de acuerdo con [Conexión de señales de seguridad en la página 26](#).

Continúa en la página siguiente

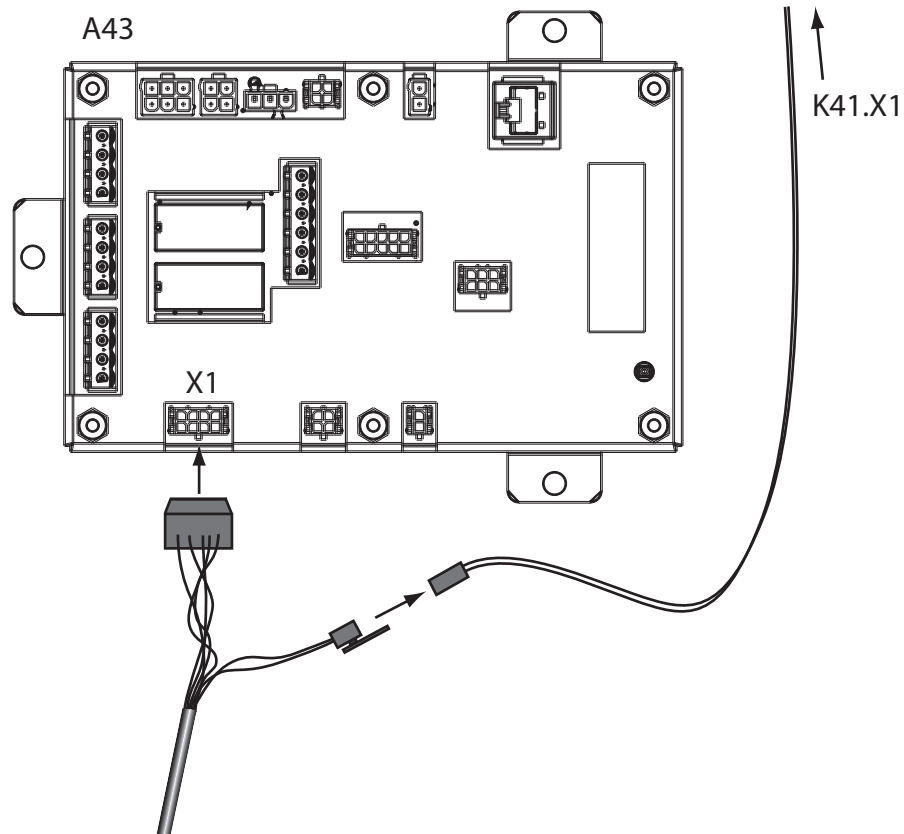
2 Instalación

2.1.3 Conexiones del Drive Module

Continuación

Conexión de señales de seguridad

El cable de señales de seguridad se conecta a la tarjeta de interfaz de contactor (A43), conector X1. También existe un conector que debe ser conectado a un cable con un conector marcado como K41.X1.



xx0600002786

2.2 Instalación del software

2.2.1 Instalación del software

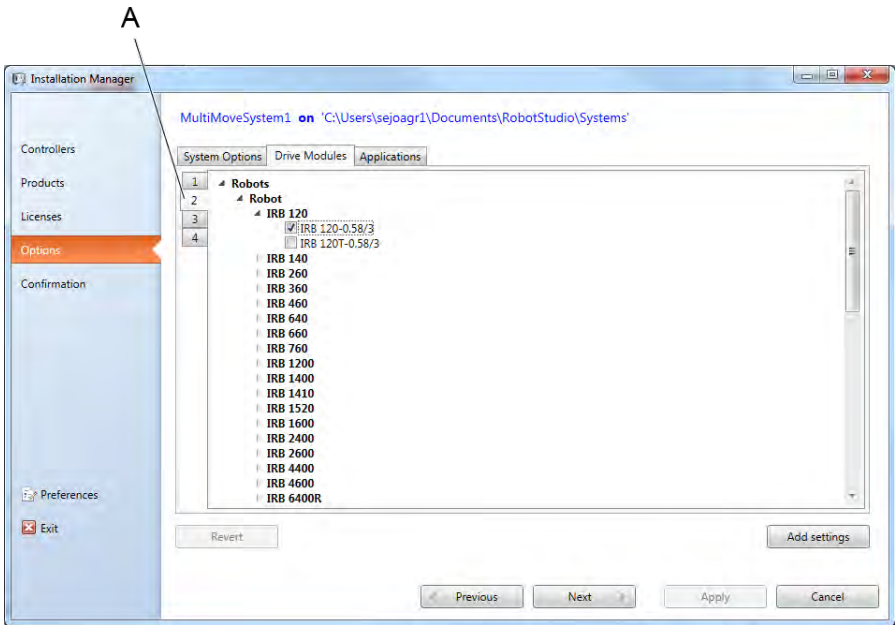
Instalación de RobotStudio y RobotWare

La instalación de RobotStudio y RobotWare en un PC se describe en el *Manual del operador - Procedimientos iniciales - IRC5 y RobotStudio*.

Creación de un sistema MultiMove

La creación de un nuevo sistema se describe en *Manual del operador - RobotStudio*. Seleccione la opción MultiMove en **Opciones del sistema**.

Una característica específica de los sistemas MultiMove es que se debe seleccionar un robot en la pestaña **Drive Modules** para cada Drive Module.



xx1500000865

A	Una pestaña para cada Drive Module
---	------------------------------------

Configuraciones automáticas en el momento de la instalación

Al crear un sistema, algunos ajustes se configuran automáticamente basándose en la información de su licencia. Para cada robot se crean los parámetros siguientes:

- Task
- Mechanical Unit Group
- Mechanical Unit
- Motion Planner

Para obtener más información acerca de estos tipos de parámetros del sistema, consulte [Parámetros del sistema en la página 30](#).

Continúa en la página siguiente

2 Instalación

2.2.1 Instalación del software

Continuación



¡CUIDADO!

El planificador de movimientos (tipo Motion Planner) creado por el proceso de instalación está configurado para optimizar los movimientos de su robot en concreto. Si la configuración predeterminada se ajusta de forma que el robot utiliza un planificador de movimientos incorrecto, el movimiento del robot se verá afectado.

3 Configuración

3.1 Descripción general de la configuración

Acerca de los parámetros del sistema

Este capítulo contiene una descripción breve de los distintos parámetros de sistema específicos de MultiMove. En él no se mencionan los parámetros que se utilizan de la misma forma que los de un sistema de robot independiente.

Para obtener más información acerca de los parámetros de sistema, consulte el *Manual de referencia técnica - Parámetros del sistema*.

Acerca de los ejemplos

Los tres primeros ejemplos tratan los temas *Controller* y *Motion*, dado que están relacionados con la constelación física del sistema de robot. El último ejemplo corresponde al tema *I/O System*, que se gestiona de una forma similar independientemente del sistema de robot.

3 Configuración

3.2.1 Tema Controller

3.2 Parámetros del sistema

3.2.1 Tema Controller

Task

Estos parámetros corresponden al tipo *Task* del tema *Controller*:

Parámetro	Descripción
Task	El nombre de la tarea. Recuerde que el nombre de cada tarea debe ser exclusivo. Esto significa que no puede tener el mismo nombre que la unidad mecánica y ninguna variable del programa de RAPID puede tener el mismo nombre.
Type	Controla el inicio y la detención del sistema y su comportamiento al reiniciarlo: <ul style="list-style-type: none">NORMAL: El programa de la tarea se inicia y detiene manualmente (por ejemplo desde el FlexPendant). La tarea se detiene en caso de un paro de emergencia.STATIC: En caso de reinicio, el programa de la tarea prosigue desde el punto en el que se encuentre. El programa de la tarea no puede ser detenido desde el FlexPendant ni con un paro de emergencia.SEMISTATIC: En caso de reinicio, el programa de la tarea se inicia desde el principio. El programa de la tarea no puede ser detenido desde el FlexPendant ni con un paro de emergencia. Las tareas que controlan una unidad mecánica deben ser del tipo NORMAL.
MotionTask	Indica si el programa de la tarea puede controlar una unidad mecánica con instrucciones de movimiento de RAPID.
Use Mechanical Unit Group	Define qué grupo de unidades mecánicas se utiliza en esta tarea. <i>Use Mechanical Unit Group</i> corresponde al parámetro <i>Name</i> del tipo <i>Mechanical Unit Group</i> . Una tarea de movimiento (<i>MotionTask</i> con el valor Yes) controla las unidades mecánicas del grupo de unidades mecánicas. Una tarea sin movimientos (<i>MotionTask</i> con el valor No) puede leer los valores (por ejemplo la posición del TCP) de las unidades mecánicas activas del grupo de unidades mecánicas. Recuerde que el parámetro <i>Use Mechanical Unit Group</i> debe estar definido en todas las tareas, incluso si la tarea no controla ninguna unidad mecánica.

Mechanical Unit Group

Cada grupo de unidades mecánicas debe contener al menos una unidad mecánica, un robot u otro tipo de unidad mecánica (es decir, no es posible dejar vacíos *Robot* y *Mech Unit 1* simultáneamente).

Estos parámetros corresponden al tipo *Mechanical Unit Group* del tema *Controller*:

Parámetro	Descripción
Name	El nombre del grupo de unidades mecánicas.
Robot	Especifica el robot (con TCP), si lo hay, del grupo de unidades mecánicas. <i>Robot</i> corresponde al parámetro <i>Name</i> del tipo <i>Mechanical Unit</i> del tema <i>Motion</i> .

Continúa en la página siguiente

Parámetro	Descripción
Mech Unit 1	Especifica una unidad mecánica sin TCP, si lo hay, del grupo de unidades mecánicas. <i>Mech Unit 1</i> corresponde al parámetro <i>Name</i> del tipo <i>Mechanical Unit</i> del tema <i>Motion</i> .
Mech Unit 2	Especifica la segunda unidad mecánica sin TCP, si hay más de una, del grupo de unidades mecánicas. <i>Mech Unit 2</i> corresponde al parámetro <i>Name</i> del tipo <i>Mechanical Unit</i> del tema <i>Motion</i> .
Mech Unit 3	Especifica la tercera unidad mecánica sin TCP, si hay más de dos, del grupo de unidades mecánicas. <i>Mech Unit 3</i> corresponde al parámetro <i>Name</i> del tipo <i>Mechanical Unit</i> del tema <i>Motion</i> .
Mech Unit 4	Especifica la cuarta unidad mecánica sin TCP, si hay más de tres, del grupo de unidades mecánicas. <i>Mech Unit 4</i> corresponde al parámetro <i>Name</i> del tipo <i>Mechanical Unit</i> del tema <i>Motion</i> .
Mech Unit 5	Especifica la quinta unidad mecánica sin TCP, si hay más de cuatro, del grupo de unidades mecánicas. <i>Mech Unit 5</i> corresponde al parámetro <i>Name</i> del tipo <i>Mechanical Unit</i> del tema <i>Motion</i> .
Mech Unit 6	Especifica la sexta unidad mecánica sin TCP, si hay más de cinco, del grupo de unidades mecánicas. <i>Mech Unit 6</i> corresponde al parámetro <i>Name</i> del tipo <i>Mechanical Unit</i> del tema <i>Motion</i> .
Use Motion Planner	Define qué planificador de movimientos se utiliza para calcular los movimientos de este grupo de unidades mecánicas. <i>Use Motion Planner</i> corresponde al parámetro <i>Name</i> del tipo <i>Motion Planner</i> del tema <i>Motion</i> .

3 Configuración

3.2.2 Tema Motion

3.2.2 Tema Motion

Drive Module User Data

Si es necesario desconectar un Drive Module sin perturbar a los robots y ejes adicionales conectados a los otros Drive Module del sistema de robots, utilice la función Drive Module Disconnect.

Este parámetro pertenece al tipo *Drive Module User Data* del tema *Motion*.

Parámetro	Descripción
Allow Drive Module Disconnect	Cambie <i>Allow Drive Module Disconnect</i> a TRUE para desconectar el Drive Module.

Mechanical Unit

Ninguno de los parámetros de tipo *Mechanical Unit* pueden editarse para un robot. Sólo pueden editarse para los ejes adicionales.

Estos parámetros corresponden al tipo *Mechanical Unit* del tema *Motion*:

Parámetro	Descripción
Name	El nombre de la unidad mecánica.
Allow move of user frame	Indica si se permite que la unidad mecánica traslade las bases de coordenadas del usuario.
Activate at Start Up	Indica si la unidad mecánica debe activarse al iniciarse el controlador. En un sistema con un solo robot, el robot está siempre activado. En un sistema MultiMove, las distintas mecánicas (incluidos los robots) pueden estar inactivas en el momento del inicio y pueden ser activadas más adelante.
Deactivation Forbidden	Indica si se permite desactivar la unidad mecánica. En un sistema con un solo robot, no es posible desactivar el robot. En un sistema MultiMove, un robot puede estar desactivado mientras otro sigue activo.

Motion Planner

Los planificadores de movimientos calculan los movimientos de un grupo de unidades mecánicas. Cuando se utilizan varias tareas en un modo de movimiento sincronizado, utilizan el mismo planificador de movimientos (el primero de los planificadores de movimientos implicados). Consulte las imágenes de los ejemplos siguientes.

En el momento de la instalación, se configura un *Motion Planner* para cada robot. El *Motion Planner* está configurado para optimizar los movimientos de cada robot en concreto. No cambie la conexión entre el robot y el *Motion Planner*.

Estos parámetros corresponden al tipo *Motion Planner* del tema *Motion*:

Parámetro	Descripción
Name	El nombre del planificador de movimientos.

Continúa en la página siguiente

Parámetro	Descripción
Speed Control Warning	<p>En el modo de movimiento sincronizado, la velocidad de un robot puede ser menor que la velocidad programada. Esto se debe a que otro robot puede limitar la velocidad (por ejemplo, si el otro robot tiene una trayectoria más larga). Si <i>Speed Control Warning</i> tiene el valor Yes, aparece una advertencia cuando el robot se mueve a una velocidad inferior a la velocidad programada, respecto del objeto de trabajo.</p> <p><i>Speed Control Warning</i> sólo se utiliza para supervisar la velocidad del TCP, es decir, que la velocidad de un eje adicional no se supervisa.</p>
Speed Control Percent	<p>Si <i>Speed Control Warning</i> tiene el valor Yes, se genera un aviso cuando la velocidad real sea menor que este porcentaje de la velocidad programada.</p>

3 Configuración

3.2.3 Tema I/O

3.2.3 Tema I/O

Sistemas con varios robots

La configuración del tema *I/O* para un sistema con varios robots no suele ser distinta de la de un sistema con un solo robot. Sin embargo, en algunas entradas y salidas del sistema existe la necesidad de especificar a qué tarea o qué robot se refieren. A continuación encontrará una descripción de los parámetros del sistema utilizados para especificar para qué tarea es válida una entrada de sistema o para qué robot es válida una salida de sistema. También se explica en qué situaciones es necesario utilizarlas.

Para obtener más información, consulte el *Manual de referencia técnica - Parámetros del sistema*.

System Input

Estos parámetros corresponden al tipo *System Input* del tema *I/O*.

Parámetro	Descripción
Argument 2	Especifica a qué tarea debe afectar esta entrada del sistema. Si el parámetro <i>Action</i> tiene el valor <i>Interrupt</i> o <i>Load and Start</i> , <i>Argument 2</i> debe especificar una tarea. Todos los demás valores de <i>Action</i> dan lugar a una entrada de sistema que es válida para todas las tareas y hacen que <i>Argument 2</i> no sea necesario. <i>Argument 2</i> corresponde al parámetro <i>Task</i> del tipo <i>Task</i> .

System Output

Estos parámetros corresponden al tipo *System Output* del tema *I/O*.

Parámetro	Descripción
Argument	Especifica a qué unidad mecánica corresponde la salida del sistema. Si el parámetro <i>Status</i> tiene el valor <i>TCP Speed</i> , <i>TCP Speed Reference</i> o <i>Mechanical Unit Active</i> , <i>Argument</i> debe especificar una unidad mecánica. Con cualquier otro valor en <i>Status</i> , la salida del sistema no se refiere a un solo robot y <i>Argument</i> no es necesario. <i>Argument</i> corresponde al parámetro <i>Name</i> del tipo <i>Mechanical Unit</i> del tema <i>Motion</i> .

3.3 Ejemplos de configuración

3.3.1 Ejemplo de configuración para "UnsyncArc"

Acerca de este ejemplo

Éste es un ejemplo de cómo configurar el ejemplo "UnsyncArc", con dos robots independientes. Cada robot se maneja con su propia tarea.

Task

Task	Type	MotionTask	Use Mechanical Unit Group
T_ROB1	NORMAL	Yes	rob1
T_ROB2	NORMAL	Yes	rob2

Mechanical Unit Group

Name	Robot	Mech Unit 1	Use Motion Planner
rob1	ROB_1		motion_planner_1
rob2	ROB_2		motion_planner_2

Motion Planner

Name	Speed Control Warning
motion_planner_1	No
motion_planner_2	No

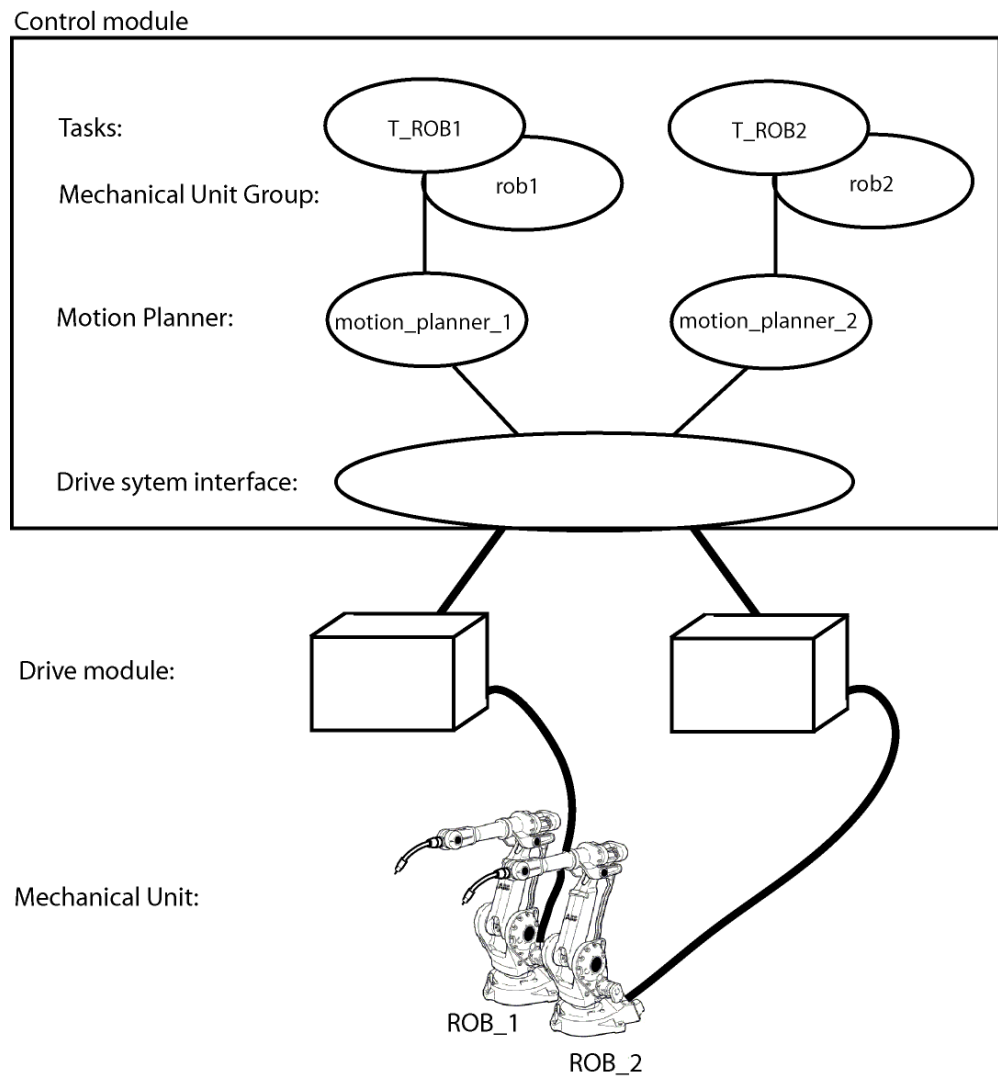
Continúa en la página siguiente

3 Configuración

3.3.1 Ejemplo de configuración para "UnsyncArc"

Continuación

Figura



en0400000773

3.3.2 Ejemplo de configuración para "SyncArc"

Acerca de este ejemplo

Éste es un ejemplo de cómo configurar el ejemplo "SyncArc", con dos robots y un posicionador. Cada una de estas tres unidades mecánicas se maneja con su propia tarea.

Task

Task	Type	MotionTask	Use Mechanical Unit Group
T_ROB1	NORMAL	Yes	rob1
T_ROB2	NORMAL	Yes	rob2
T_STN1	NORMAL	Yes	stn1

Mechanical Unit Group

Name	Robot	Mech Unit 1	Use Motion Planner
rob1	ROB_1		motion_planner_1
rob2	ROB_2		motion_planner_2
stn1		STN_1	motion_planner_3

Motion Planner

Name	Speed Control Warning	Speed Control Percent
motion_planner_1	Yes	90
motion_planner_2	Yes	90
motion_planner_3	No	

Mechanical Unit

Name	Allow move of user frame	Activate at Start Up	Deactivation Forbidden
ROB_1	Yes	Yes	Yes
ROB_2	Yes	Yes	Yes
STN_1	Yes	Yes	No

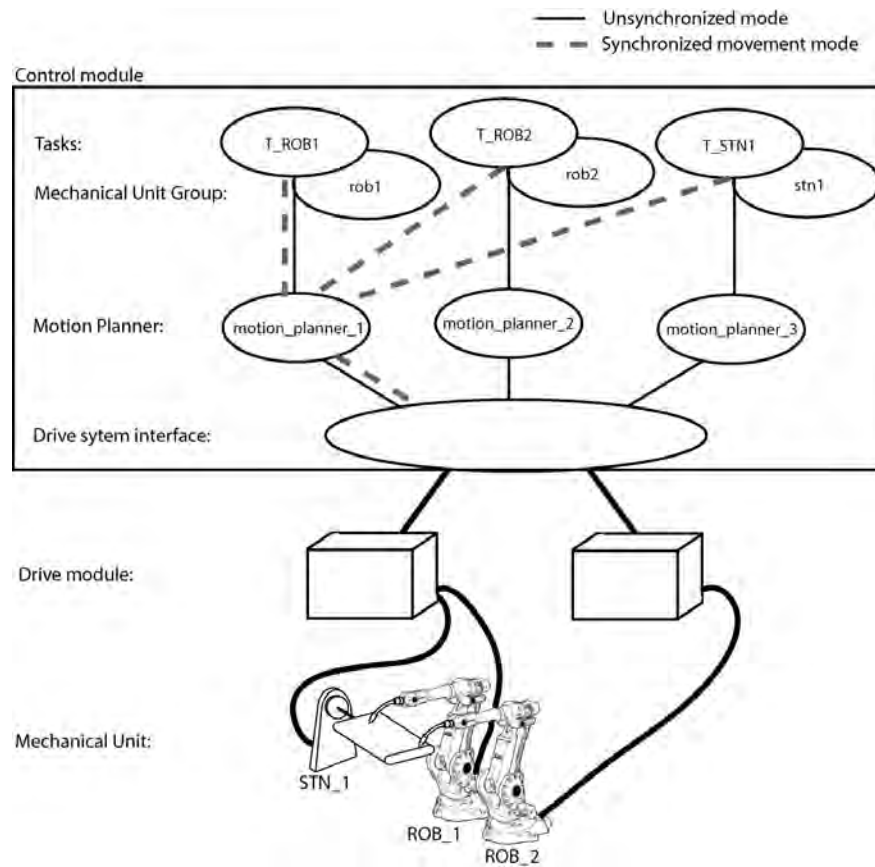
Continúa en la página siguiente

3 Configuración

3.3.2 Ejemplo de configuración para "SyncArc"

Continuación

Figura



en0400000774

3.3.3 Ejemplo de configuración de E/S

Acerca de este ejemplo

Éste es un ejemplo de cómo configurar algunas señales de E/S que requieren un argumento de tarea o robot. Este ejemplo se basa en el ejemplo "SyncArc".

La señal de entrada `di_position` se ha configurado para interrumpir la ejecución del programa y llamar a la rutina `SetStartPosition` de la tarea `T_STN1`. La señal de salida `ao_speed1` está configurada para indicar la velocidad del robot 1, mientras que la salida `ao_speed2` se utiliza para indicar la velocidad del robot 2.

System Input

Signal Name	Action	Argument	Argument 2
<code>di_position</code>	Interrupt	<code>SetStartPosition</code>	<code>T_STN1</code>

System Output

Status	Signal Name	Argument
TCP Speed	<code>ao_speed1</code>	<code>ROB_1</code>
TCP Speed	<code>ao_speed2</code>	<code>ROB_2</code>

Esta página se ha dejado vacía intencionadamente

4 Calibración

4.1 Descripción general de la calibración

Dos tipos de calibración

Existen dos tipos de calibración que deben realizarse para un sistema de robot:

- 1 La calibración de ejes, que garantiza que todos los ejes se encuentren en las posiciones correctas. Normalmente, esto se realiza antes de la entrega de un nuevo robot y sólo requiere una recalibración después de reparar el robot. Para obtener más información, consulte el manual del producto del robot correspondiente.
- 2 La calibración de sistemas de coordenadas debe realizarse tras la instalación del robot. A continuación aparece una breve descripción de qué sistemas de coordenadas se deben calibrar y en qué orden.

Calibración de sistemas de coordenadas

En primer lugar es necesario decidir qué sistemas de coordenadas deben utilizarse y cómo establecer sus orígenes y direcciones. Para ver algunos ejemplos de sistemas de coordenadas adecuados, consulte [Ejemplos de sistemas de coordenadas en la página 45](#).

A continuación, los sistemas de coordenadas se calibran en el orden siguiente:

	Acción
1	Calibre la herramienta. Este proceso incluye la calibración del TCP y de los datos de carga. Para obtener una descripción de cómo calibrar la herramienta, consulte el <i>Manual del operador - IRC5 con FlexPendant</i> .
2	Para todos los robots, calibre el sistema de coordenadas de la base respecto del sistema de coordenadas mundo. Para obtener una descripción de cómo calibrar el sistema de coordenadas de la base de un robot, consulte el <i>Manual del operador - IRC5 con FlexPendant</i> . Si un robot determinado ya tiene calibrado su sistema de coordenadas de la base, el sistema de coordenadas de la base de otro robot puede calibrarse haciendo que los TCPs de los dos robots coincidan en varios puntos. Este método se describe en Calibración relativa en la página 42 .
3	En el caso de los posicionadores, calibre los sistemas de coordenadas de la base respecto del sistema de coordenadas mundo. Para obtener una descripción de cómo calibrar el sistema de coordenadas de la base de un posicionador, consulte el <i>Application manual - Additional axes and stand alone controller</i> .
4	Calibre los sistemas de coordenadas de usuario respecto del sistema de coordenadas mundo. Para obtener una descripción de cómo calibrar un sistema de coordenadas de usuario, consulte el <i>Manual del operador - IRC5 con FlexPendant</i> .
5	Calibre los sistemas de coordenadas de objeto respecto del sistema de coordenadas de usuario. Para obtener una descripción de cómo calibrar un sistema de coordenadas de objeto, consulte el <i>Manual del operador - IRC5 con FlexPendant</i> .

4 Calibración

4.2 Calibración relativa

4.2 Calibración relativa

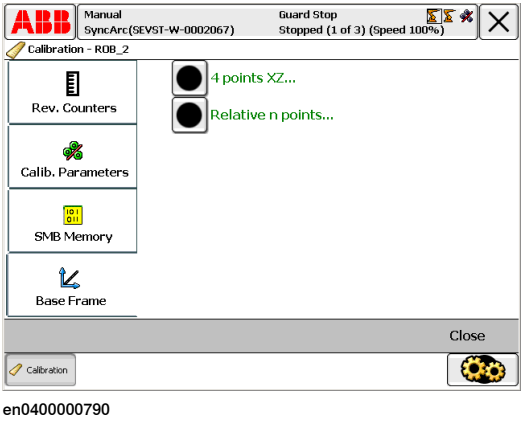
En qué consiste la calibración relativa

La calibración relativa se utiliza para calibrar el sistema de coordenadas de la base de un robot, utilizando un robot previamente calibrado. Este método de calibración sólo puede usarse en los sistemas MultiMove en los que dos robots estén a una distancia lo suficientemente cercana como para que tengan en común una parte de sus áreas de trabajo.

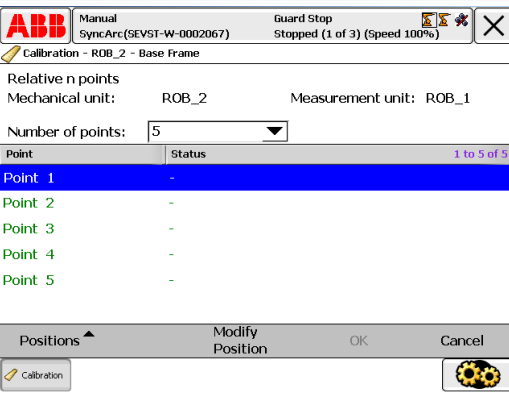
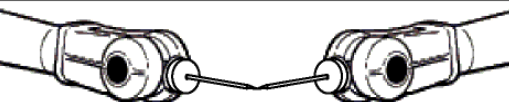
Si un robot tiene un sistema de coordenadas de la base que es idéntico al sistema de coordenadas mundo, puede usarse como una referencia para otro robot. Si ninguno de los robots tiene un sistema de coordenadas de la base idéntico al sistema de coordenadas mundo, es necesario calibrar previamente el sistema de coordenadas de la base de un robot. Para obtener más información acerca de otros métodos de calibración, consulte el *Manual del operador - IRC5 con FlexPendant*.

Cómo realizar la calibración relativa

Las herramientas de los dos robots deben estar calibradas correctamente antes de usar la calibración relativa y las herramientas deben estar activas durante la calibración.

	Acción	Información/figura
1	En el menúABB, seleccione Calibration (Calibración).	
2	Toque en la pantalla el robot que desee calibrar.	
3	Toque Base de coordenadas de la base .	
4	Toque n puntos relativos .	
5	Si tiene más de dos robots, debe seleccionar qué robot desea utilizar como referencia. Si sólo tiene dos robots, este paso se omite.	

Continúa en la página siguiente

	Acción	Información/figura
6	<p>La calibración puede realizarse con un número de puntos de entre 3 y 10. Utilice el campo Number of Points (Número de puntos) para indicar el número de puntos que desea utilizar. Para obtener una exactitud adecuada, se recomienda utilizar al menos 5 puntos.</p>	 <p>en0400000791</p>
7	<p>Seleccione Point 1 (Punto 1).</p>	
8	<p>Mueva el robot que desee calibrar y el robot de referencia de forma que los dos TCPs se encuentren en el mismo punto.</p>	 <p>xx0400000785</p>
9	<p>Toque Modify Position (Modificar posición).</p>	
10	<p>Repita los pasos del 7 al 9 con todos los puntos. Asegúrese de que los puntos aparezcan repartidos por las coordenadas, x, y, z. Por ejemplo, si todos los puntos se encuentran a una misma altura, la calibración de la coordenada Z no será muy exacta.</p>	
11	<p>Toque OK.</p>	<p>Se muestra el resultado de la calibración.</p>
12	<p>Toque OK para aceptar la calibración.</p>	
13	<p>Reinicie el controlador.</p>	

4 Calibración

4.3 Cadenas de calibración

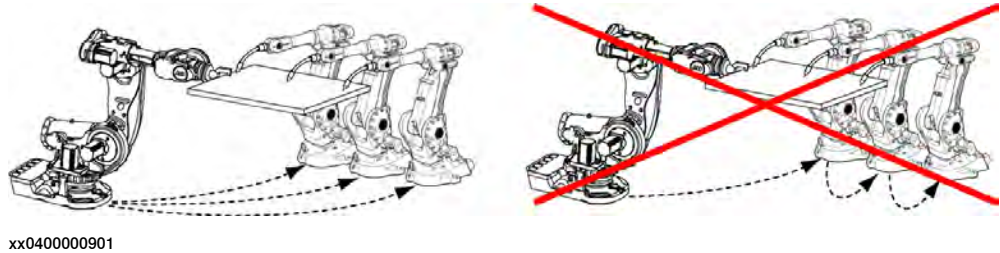
4.3 Cadenas de calibración

Evite utilizar cadenas largas de calibración

Si un robot calibrado con una calibración relativa hace las veces de referencia de la siguiente calibración, los problemas de exactitud de las calibraciones se van sumando hasta el último robot.

Ejemplo

Tiene cuatro robots, de los cuales el robot 1 sostiene una pieza de trabajo en la que trabajan los robots 2, 3 y 4.



Calibre los robots 2, 3 y 4 respecto del robot 1.

Por ejemplo, si utiliza el robot 1 como referencia del robot 2, el robot 2 como referencia del robot 3 y el robot 3 como referencia del robot 4, la exactitud del robot 4 sería insuficiente.

4.4 Ejemplos de sistemas de coordenadas

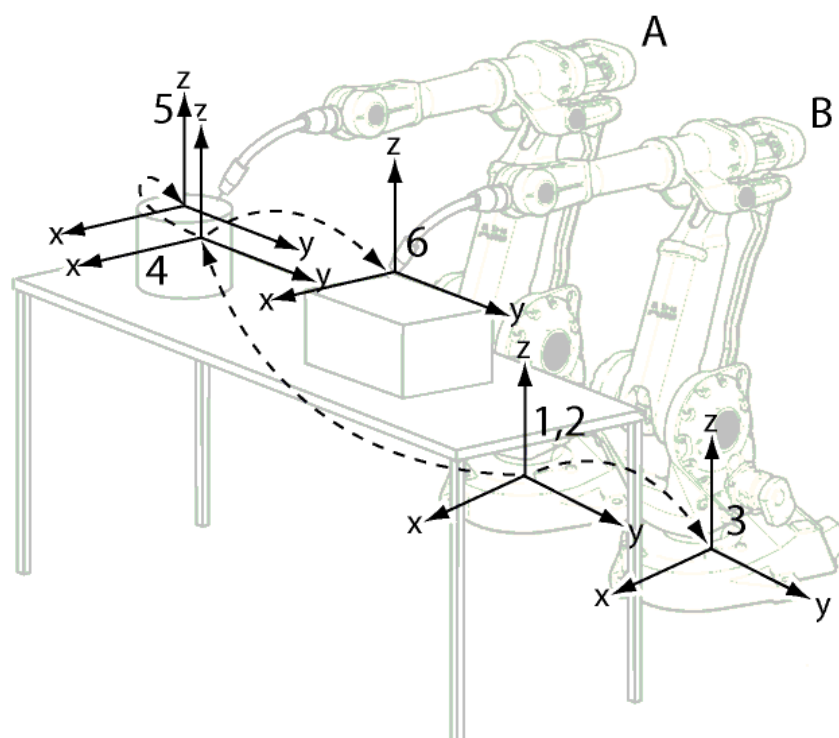
4.4.1 Ejemplo "UnsyncArc"

Acerca de este ejemplo

En este ejemplo, el sistema de coordenadas mundo y el sistema de coordenadas de la base del robot 1 (A) son idénticos.

Se define el sistema de coordenadas de la base del robot 2 (B). Los dos robots tienen un sistema de coordenadas de usuario con el origen situado en la esquina de un tablero. Se define un sistema de coordenadas de objeto para cada objeto de trabajo de robot.

Figura



xx0300000591

Sistemas de coordenadas

Elemento	Descripción
A	Robot 1
B	Robot 2
1	Sistema de coordenadas mundo
2	Sistema de coordenadas de la base del robot 1
3	Sistema de coordenadas de la base del robot 2
4	Sistema de coordenadas de usuario de los dos robots
5	Sistema de coordenadas del objeto del robot 1

Continúa en la página siguiente

4 Calibración

4.4.1 Ejemplo "UnsyncArc"

Continuación

Elemento	Descripción
6	Sistema de coordenadas de objeto del robot 2

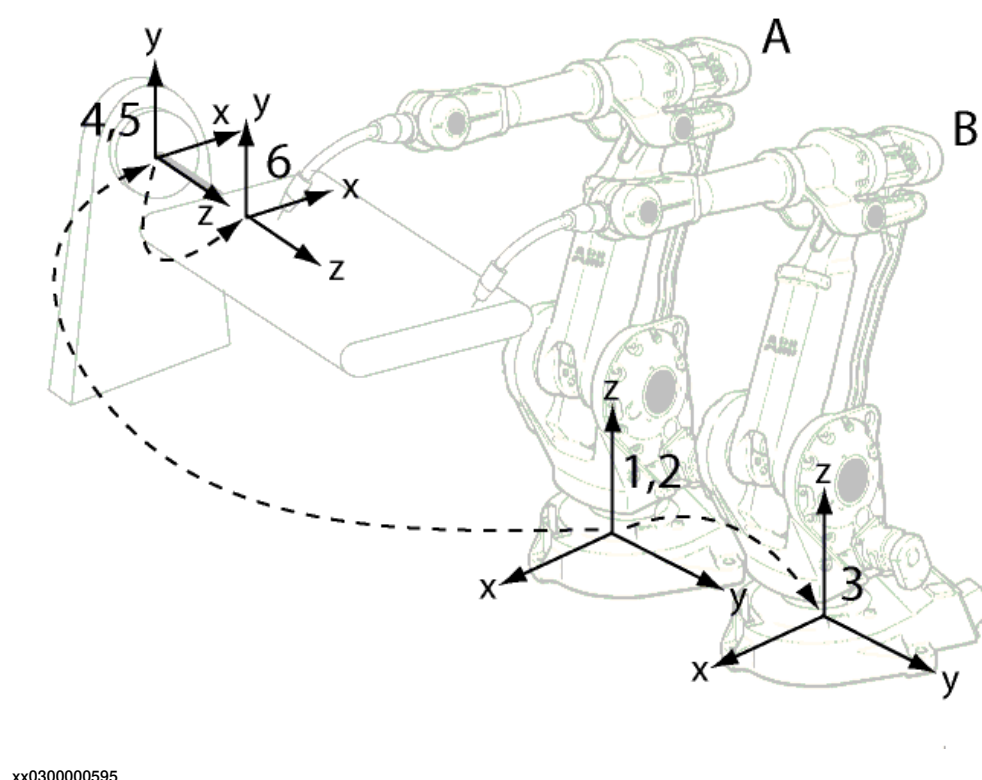
4.4.2 Ejemplo "SyncArc"

Acerca de este ejemplo

En este ejemplo, el sistema de coordenadas mundo y el sistema de coordenadas de la base del robot 1 (A) son idénticos.

Se define el sistema de coordenadas de la base del robot 2 (B). Se define un sistema de coordenadas de usuario para conectarlo al eje giratorio del posicionador. Se define un sistema de coordenadas de objeto que será fijo respecto del objeto de trabajo sostenido por el posicionador.

Figura



xx0300000595

Sistemas de coordenadas

Elemento	Descripción
A	Robot 1
B	Robot 2
1	Sistema de coordenadas mundo
2	Sistema de coordenadas de la base del robot 1
3	Sistema de coordenadas de la base del robot 2
4	Sistema de coordenadas de la base del posicionador
5	Sistema de coordenadas de usuario de los dos robots
6	Sistema de coordenadas de objeto de los dos robots

Esta página se ha dejado vacía intencionadamente

5 Interfaz de usuario específica de MultiMove

5.1 FlexPendant para el sistema MultiMove

Acerca del FlexPendant del sistema MultiMove

El trabajo con el FlexPendant en un sistema MultiMove no es muy distinto de cuando se usa un sistema con un solo robot. Este capítulo explica algunos detalles que son específicos de los sistemas MultiMove. Para obtener información general acerca del FlexPendant, consulte el *Manual del operador - IRC5 con FlexPendant*.

¿Qué elementos son específicos de MultiMove?

Algunas de las cosas que son específicas de MultiMove son las siguientes:

- La barra de estado muestra cuáles de los robots (y de los ejes adicionales) están coordinados. Consulte [Indicaciones en la barra de estado en la página 50](#).
- Al abrir el editor de programas, debe seleccionar una tarea. Consulte [Cómo abrir el Editor de programas en la página 52](#).
- La ventana Producción contiene pestañas para las distintas tareas. Consulte [Ventana Producción en la página 53](#).
- El menú de unidades mecánicas puede contener varios robots. Consulte [Menú de unidades mecánicas en la página 54](#).
- Puede seleccionar qué tareas desea ejecutar en el momento del inicio. Consulte [Selección de qué tareas se inician con el botón INICIO en la página 55](#).
- Existe un método adicional para calibrar una base de coordenadas de robot, la calibración relativa. Consulte [Calibración relativa en la página 42](#).







5 Interfaz de usuario específica de MultiMove

5.2 Indicaciones en la barra de estado

5.2 Indicaciones en la barra de estado

Descripciones de los símbolos

El lado derecho de la barra de estado, situada en la parte superior del FlexPendant, contiene símbolos para las unidades mecánicas presentes en el sistema.

Símbolo	Descripción
 xx0400001165	Un robot que no es la unidad mecánica seleccionada ni está coordinado con la unidad mecánica seleccionada. Las operaciones de movimiento no afectarán a este robot.
 xx0400001166	Un robot que es la unidad mecánica seleccionada o está coordinado con la unidad mecánica seleccionada. Las operaciones de movimiento afectarán a este robot (junto con las demás unidades mecánicas coordinadas).
 xx0400001167	Un robot que pertenece a una tarea inactiva. Para saber cómo se desactivan las tareas, consulte Selección de qué tareas se inician con el botón INICIO en la página 55 .
 xx0400001168	Un eje adicional que no es la unidad mecánica seleccionada ni está coordinado con la unidad mecánica seleccionada. Las operaciones de movimiento no afectarán a este eje adicional.
 xx0400001169	Un eje adicional que es la unidad mecánica seleccionada o está coordinado con la unidad mecánica seleccionada. Las operaciones de movimiento afectarán a este eje adicional (junto con las demás unidades mecánicas coordinadas).
 xx0400001170	Un eje adicional que no está activo. O bien la unidad mecánica está desactivada, o la tarea está inactiva.

Ejemplo



en0400001158

A continuación se ofrece un ejemplo con un sistema MultiMove compuesto por 4 robots y 2 ejes adicionales y en el que...

- El robot 1 pertenece a una tarea que está inactiva.
- El robot 2 no está seleccionado ni coordinado con la unidad seleccionada (no se ve afectado por las operaciones de movimiento).

Continúa en la página siguiente

- El eje adicional 1 es la unidad mecánica seleccionada y los robots 3 y 4 están coordinados con el eje adicional 1. Cualquier operación de movimiento en este punto moverá estas tres unidades mecánicas.
- El eje adicional 2 está desactivado o su tarea está inactiva.

5.3 Cómo abrir el Editor de programas

Selección de la tarea

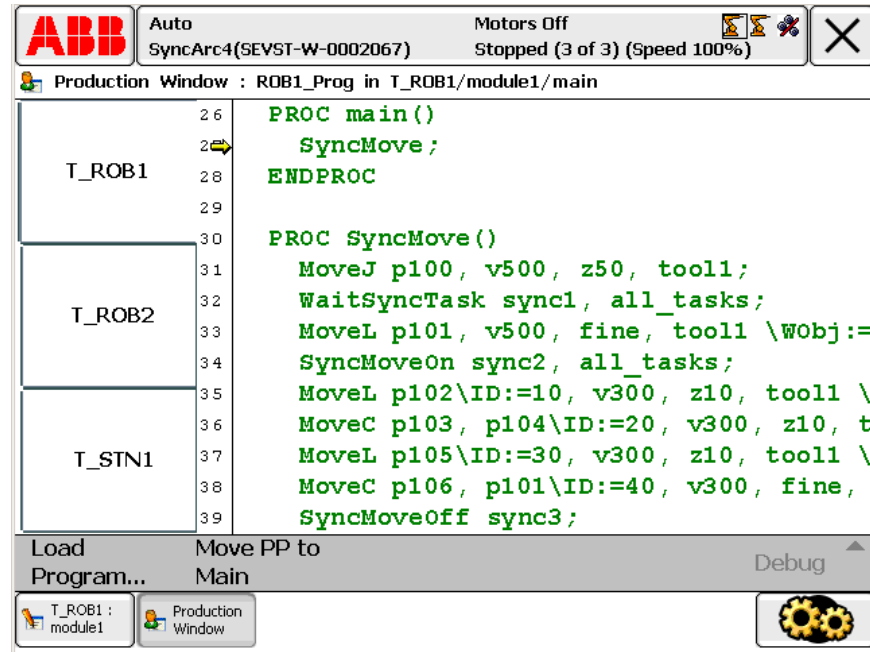
Al abrir el Editor de programas de un sistema que cuenta con más de una tarea, aparece una lista con todas las tareas. Al tocar la tarea que desee, se muestra el código de programa de dicha tarea.

En el caso de los sistemas con una única tarea, esta lista no se muestra en ningún caso. Se muestra directamente el código del programa.

5.4 Ventana Producción

La pantalla gráfica

En un sistema que cuenta con más de una tarea de movimiento, aparece una pestaña para cada tarea de movimiento. Al tocar una pestaña, puede ver el código del programa de la tarea correspondiente y dónde se encuentran el puntero de programa y el puntero de movimiento dentro de la tarea.



en0400000796

Movimiento del puntero de programa

Si toca **Mover PP a Main**, el puntero de programa se mueve a Main en todos los programas de tarea de movimiento.

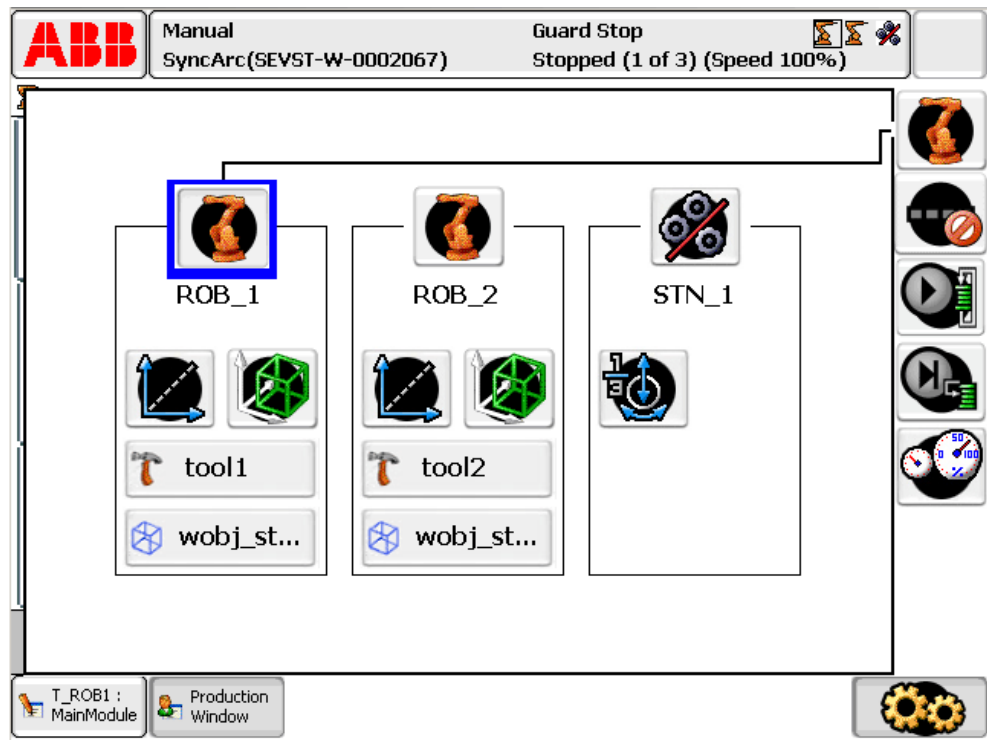
5 Interfaz de usuario específica de MultiMove

5.5 Menú de unidades mecánicas

5.5 Menú de unidades mecánicas

La pantalla gráfica

En el menú QuickSet, toque el botón del menú de unidades mecánicas. Se muestran todas las unidades mecánicas.



en0400000789

La unidad mecánica seleccionada aparece resaltada por un marco que la rodea. Cualquier unidad mecánica que esté coordinada con la unidad seleccionada se indicará con un marco parpadeante y el texto "Coord".

Movimientos coordinados o no coordinados

El movimiento de una unidad mecánica hace que se muevan automáticamente todas las unidades que estén coordinadas con ella.

En el ejemplo anterior, el movimiento de STN_1 mueve también ROB_1 y ROB_2, dado que están coordinados con STN_1 (el objeto de trabajo wobj_stn1 es movido por STN_1). Para poder mover STN_1 sin mover los robots, cambie el sistema de coordenadas de los robots a coordenadas mundo o cambie el objeto de trabajo de los robots a wobj0. Para obtener más información sobre la coordinación de los objetos de trabajo con una unidad mecánica, consulte [Objetos de trabajo coordinados en la página 63](#).

5.6 Selección de qué tareas se inician con el botón INICIO

Segundo plano

El comportamiento predeterminado es que los programas de todas las tareas NORMAL se inicien simultáneamente al presionar el botón INICIO. Sin embargo, no es necesario ejecutar todos los programas de tipo NORMAL al mismo tiempo. Es posible seleccionar cuál de los programas de tarea NORMAL debe iniciarse al presionar el botón INICIO.

Si se ha seleccionado **Todas las tareas** en la **Configuración de panel de tareas**, los programas de todas las tareas STATIC y SEMISTATIC que tengan en *TrustLevel* el valor NoSafety pueden seleccionarse para iniciarlas con el botón INICIO, ejecutarlas paso a paso hacia delante con el botón Adelant., ejecutarlas paso a paso hacia atrás con el botón Atrás y detenerlas con el botón PARAR.

Si **Configuración de panel de tareas** tiene el valor **Sólo tareas normales**, todas las demás tareas de tipo STATIC y SEMISTATIC se atenúan y no pueden seleccionarse en el panel de tareas, menú de configuración rápida (consulte el *Manual del operador - IRC5 con FlexPendant*, sección *Menú de configuración rápida*). Todas las tareas STATIC y SEMISTATIC se iniciarán si se pulsa el botón de inicio.

Si **Configuración de panel de tareas** tiene el valor **Todas las tareas**, las tareas STATIC y SEMISTATIC que tengan el *TrustLevel/NoSafety* pueden seleccionarse en el panel de tareas. Todas las tareas STATIC y SEMISTATIC pueden ser paradas, ejecutadas paso a paso e iniciadas. .

Las tareas STATIC o SEMISTATIC no seleccionadas en el panel de tareas pueden seguir siendo ejecutadas. Esto no es posible con las tareas de tipo NORMAL.

El modo de ejecución es siempre continuo en las tareas de tipo STATIC y SEMISTATIC. El valor de Modo de ejecución en el menú de configuración rápida sólo se aplica a las tareas de tipo NORMAL (consulte el *Manual del operador - IRC5 con FlexPendant*, sección *Menú de configuración rápida*).

Esto sólo funciona en el modo manual. Ninguna tarea de tipo STATIC o SEMISTATIC puede ser iniciada, ejecutada paso a paso o detenida en el modo automático.

Configuración de panel de tareas

Para ir a la **Configuración de panel de tareas**, toque el menú ABB, seguido de **Panel de control**, **FlexPendant** y **Configuración de panel de tareas**.

Selección de tareas

Utilice este procedimiento para seleccionar qué tareas deben iniciarse al presionar el botón INICIO.

	Acción
1	Cambie el controlador al modo manual.

Continúa en la página siguiente

5 Interfaz de usuario específica de MultiMove

5.6 Selección de qué tareas se inician con el botón INICIO

Continuación

	Acción
2	En el FlexPendant, toque el botón Configuración rápida y a continuación el botón de panel de tareas para mostrar todas las tareas. Si Configuración de panel de tareas tiene el valor Sólo tareas normales , todas las demás tareas de tipo STATIC y SEMISTATIC se atenúan y no pueden seleccionarse. Si Configuración de panel de tareas tiene el valor Todas las tareas , es posible seleccionar las tareas de tipo STATIC y SEMISTATIC que tienen el <i>TrustLevel/NoSafety</i> , mientras que las tareas de tipo STATIC y SEMISTATIC que tienen otros valores en <i>TrustLevel</i> aparecen atenuadas y no pueden ser seleccionadas.
3	Active las casillas de verificación de las tareas cuyos programas deban iniciarse al presionar el botón INICIO.

Restablecimiento de la configuración de depuración en el modo manual

Utilice este procedimiento para reanudar la ejecución normal en el modo manual.

	Acción
1	Seleccione Sólo tareas normales en Configuración de panel de tareas .
2	Pulse el botón INICIAR. Todas las tareas de tipo STATIC y SEMISTATIC se ejecutarán continuamente y no se detendrán con el botón PARAR ni por el paro de emergencia.

Cambio al modo automático

Al cambiar al modo automático, todas las tareas de tipo STATIC y SEMISTATIC se deseleccionarán en el panel de tareas. Las tareas de tipo STATIC y SEMISTATIC detenidas se iniciarán la próxima vez que se presione el botón INICIO, Adelant. o Atrás. En este caso, las tareas se ejecutarán continuamente hacia delante y no se detendrán con el botón PARAR ni por el paro de emergencia.

Lo que ocurre con las tareas de tipo NORMAL que han sido deseleccionadas en el panel de tareas depende del parámetro del sistema *Reset* del tipo *Auto Condition Reset* del tema *Controller*. Si *Reset* tiene el valor Yes, todas las tareas de tipo NORMAL se seleccionarán en el panel de tareas y se iniciarán con el botón INICIO. Si *Reset* tiene el valor No, sólo las tareas de tipo NORMAL seleccionadas en el panel de tareas se iniciarán con el botón INICIO.



Nota

Recuerde que el cambio del valor del parámetro del sistema *Reset* afectará a los restablecimientos de depuración (p.ej. redefinición de velocidad y E/S simulada). Para obtener más información, consulte *Manual de referencia técnica - Parámetros del sistema*, sección *Auto Condition Reset*.

Reinicio del controlador

Si se reinicia el controlador, todas las tareas de tipo NORMAL conservan su estado, mientras que las tareas de tipo STATIC y SEMISTATIC se deseleccionan del panel de tareas. Cuando el controlador se pone en marcha, todas las tareas de tipo STATIC y SEMISTATIC se inician y funcionan continuamente a partir de ese momento.

Continúa en la página siguiente

Tarea deseleccionada en el modo sincronizado

Si la tarea se encuentra en un modo sincronizado, es decir, con el puntero de programa entre `SyncMoveOn` y `SyncMoveOff`, la tarea puede ser deseleccionada pero no seleccionada de nuevo. La tarea no puede ser seleccionada hasta que finalice la sincronización. Si la ejecución continúa, la sincronización llegará a finalizarse para las demás tareas, pero no para la tarea deseleccionada. La sincronización puede ser finalizada para esta tarea moviendo el puntero de programa a `Main` o a una rutina.

Si se cambia el valor del parámetro de sistema *Reset* a *Yes*, cualquier intento de cambiar al modo automático no tendrá éxito mientras la tarea deseleccionada esté en el modo sincronizado. El cambio al modo automático debe hacer que todas las tareas *NORMAL* se seleccionen, y cuando esto no es posible, tampoco es posible cambiar al modo automático.

Esta página se ha dejado vacía intencionadamente

6 Programación

6.1 Componentes de RAPID

Tipos de datos

A continuación aparece una descripción breve de los distintos tipos de datos de MultiMove. Para obtener más información, consulte el tipo de dato correspondiente en el *Manual de referencia técnica - Instrucciones, funciones y tipos de datos de RAPID*.

Tipo de dato	Descripción
syncident	Se utiliza una variable del tipo <code>syncident</code> para identificar qué instrucciones <code>WaitSyncTask</code> , <code>SyncMoveOn</code> o <code>SyncMoveOff</code> de los distintos programas de tarea deben estar sincronizadas entre sí. El nombre de la variable <code>syncident</code> debe ser igual en todos los programas de tarea. Declare las variables <code>syncident</code> de forma global en cada tarea. No reutilice una misma variable <code>syncident</code> (cada instrucción <code>WaitSyncTask</code> , <code>SyncMoveOn</code> y <code>SyncMoveOff</code> de un programa de tarea debe tener su variable <code>syncident</code> exclusiva).
tasks	La variable persistente del tipo de dato <code>tasks</code> contiene los nombres de las tareas que se sincronizarán con <code>WaitSyncTask</code> o <code>SyncMoveOn</code> . Las variables <code>tasks</code> deben estar declaradas como una variable (persistente) global del sistema, con el mismo nombre y el mismo contenido en todos los programas de tarea.
identno	Los valores numéricos o variables del tipo <code>identno</code> se utilizan en el argumento <code>ID</code> de cualquier instrucción de movimiento ejecutada entre las instrucciones <code>SyncMoveOn</code> y <code>SyncMoveOff</code> .

Dato de sistema

Los datos de sistema son los datos internos predefinidos en el robot. Es posible leer los datos de sistema, pero no cambiarlos, desde un programa de RAPID. Para obtener más información, consulte el *Manual de referencia técnica - Instrucciones, funciones y tipos de datos de RAPID*.

Dato de sistema	Descripción
ROB_ID	Referencia al robot (si lo hay) que se está controlando desde la tarea. Si se utiliza en una tarea que no controla a ningún robot, se producirá un error. Utilice siempre <code>TaskRunRob()</code> para comprobar su valor antes de usar <code>ROB_ID</code> .

Continúa en la página siguiente

6 Programación

6.1 Componentes de RAPID

Continuación

Instrucciones

A continuación aparece una descripción breve de las distintas instrucciones de MultiMove. Para obtener más información, consulte la instrucción correspondiente en el *Manual de referencia técnica - Instrucciones, funciones y tipos de datos de RAPID*.

Instrucción	Descripción
WaitSyncTask	<p>WaitSyncTask se utiliza para sincronizar varios programas de tarea en un punto especial del programa.</p> <p>La instrucción WaitSyncTask se utiliza para esperar a los otros programas de tarea. Cuando todos los programas de tarea han alcanzado la instrucción WaitSyncTask, prosigue su ejecución.</p>
SyncMoveOn	<p>SyncMoveOn se utiliza para iniciar el modo de movimiento sincronizado.</p> <p>La instrucción SyncMoveOn se utiliza para esperar a los otros programas de tarea. Cuando todos los programas de tarea han alcanzado la instrucción SyncMoveOn, prosigue su ejecución en el modo de movimiento sincronizado. Las instrucciones de movimiento de los distintos programas de tarea se ejecutan simultáneamente hasta que se ejecuta la instrucción SyncMoveOff.</p> <p>Es necesario programar un punto de paro antes de la instrucción SyncMoveOn.</p>
SyncMoveOff	<p>SyncMoveOff se utiliza para finalizar el modo de movimiento sincronizado.</p> <p>La instrucción SyncMoveOff se utiliza para esperar a los otros programas de tarea. Cuando todos los programas de tarea han alcanzado la instrucción SyncMoveOff, prosigue su ejecución en el modo no sincronizado.</p> <p>Es necesario programar un punto de paro antes de la instrucción SyncMoveOff.</p>
SyncMoveUndo	<p>SyncMoveUndo se utiliza para desactivar el movimiento sincronizado incluso si no todos los demás programas de tarea ejecutan la instrucción SyncMoveUndo.</p> <p>SyncMoveUndo se ha diseñado para gestores UNDO. Cuando se mueve el puntero de programa a un lugar distinto del procedimiento, SyncMoveUndo se utiliza para desactivar la sincronización.</p>
MoveExtJ	<p>MoveExtJ (del inglés "Move External Joints", mover ejes externos) mueve una o varias unidades mecánicas sin TCP.</p> <p>MoveExtJ se utiliza para mover ejes adicionales en tareas que no tienen ningún robot.</p>

Funciones

A continuación aparece una descripción breve de las distintas funciones de MultiMove. Para obtener más información, consulte la función correspondiente en el *Manual de referencia técnica - Instrucciones, funciones y tipos de datos de RAPID*.

Función	Descripción
IsSyncMoveOn	<p>IsSyncMoveOn se utiliza para determinar si el grupo de unidades mecánicas se encuentra en el modo de movimiento sincronizado.</p> <p>Las tareas que no controlen ninguna unidad mecánica pueden determinar si las unidades mecánicas definidas en el parámetro <i>Use Mechanical Unit Group</i> se encuentran en el modo de movimiento sincronizado.</p>

Continúa en la página siguiente

Función	Descripción
RobName	RobName se utiliza para obtener el nombre del robot controlado por la tarea. Devuelve el nombre de la unidad mecánica en forma de una cadena. Si se llama a esta función desde una tarea que no controla ningún robot, devuelve una cadena vacía.

Argumento de sincronización

A continuación aparece una descripción breve de los argumentos utilizados por las instrucciones de movimiento para facilitar la sincronización entre tareas. Para obtener más información, consulte cualquier instrucción de movimiento en el *Manual de referencia técnica - Instrucciones, funciones y tipos de datos de RAPID*.

Argumento	Descripción
ID	<p>Todas las instrucciones de movimiento ejecutadas entre las instrucciones SyncMoveOn y SyncMoveOff deben tener especificado el argumento ID. El argumento ID debe ser igual en todas las instrucciones de movimiento (de cada programa de tarea) que se desee ejecutar simultáneamente.</p> <p>El argumento ID puede ser un valor numérico o una variable syncident.</p> <p>La finalidad de ID es ofrecer una ayuda al operador, haciendo más fácil ver qué instrucciones de movimiento están sincronizadas entre sí. Asegúrese de no utilizar un valor de ID en más de una instrucción de movimiento situada entre las mismas instrucciones SyncMoveOn y SyncMoveOff. También resulta útil al operador si los valores de ID van ascendiendo con las instrucciones de movimiento consecutivas (por ejemplo 10, 20, 30, ...).</p> <p>Las instrucciones de movimiento que no se encuentren entre las instrucciones SyncMoveOn y SyncMoveOff no deben tener especificado el argumento ID.</p>

6.2 Tareas y técnicas de programación

Distintas tareas

Cada programa de tarea puede gestionar los movimientos de un robot y un máximo de 6 ejes adicionales. Pueden usarse varias tareas, cada una de ellas con un programa muy similar al programa de la tarea principal de una aplicación con un solo robot. Para obtener más información sobre las tareas, consulte la sección Multitasking del manual *Application manual - Controller software IRC5*.

Un programa de tarea por robot

Cada programa de tarea sólo puede gestionar un TCP. Esto significa que debe tener una tarea para cada robot.

Ejes adicionales en tareas separadas

Los ejes adicionales que mueven un objeto de trabajo pueden ser gestionados por el mismo programa de tarea que uno de los robots. Sin embargo, si es necesario que los ejes adicionales puedan moverse de forma independiente de los robots, la mejor opción es utilizar un programa de tarea separado para gestionar los ejes adicionales.

6.3 Objetos de trabajo coordinados

Acerca de los objetos de trabajo

Esta sección sólo describe cómo coordinar un objeto de trabajo con una unidad mecánica. Para obtener una descripción detallada de los objetos de trabajo, consulte *wobjdata - Datos de objeto de trabajo* en el *Manual de referencia técnica - Instrucciones, funciones y tipos de datos de RAPID*.

¿Qué determina la coordinación?

A la hora de declarar un objeto de trabajo, el segundo atributo (`ufprog`) y el tercero (`ufmec`) determinan si el objeto de trabajo está coordinado con cualquier unidad mecánica.

robhold

`robhold` define si el objeto de trabajo está sostenido por el robot en esta tarea. `robhold` tiene normalmente el valor `FALSE`. La tarea del robot que sostiene el objeto de trabajo (donde `robhold` tendría el valor `TRUE`) no tiene que declararlo a no ser que se use una herramienta estacionaria.

ufprog

Si el objeto de trabajo es fijo, `ufprog` tiene el valor `TRUE`.

Si es posible mover el objeto de trabajo con cualquier unidad mecánica, `ufprog` tiene el valor `FALSE`.

ufmec

`ufmec` contiene el nombre de la unidad mecánica que mueve el objeto de trabajo. Si `ufprog` tiene el valor `TRUE`, `ufmec` puede dejarse como una cadena vacía (lo que indica que ninguna unidad mecánica puede mover el objeto de trabajo).

Ejemplo 1

A continuación se ofrece un ejemplo de un objeto de trabajo que puede ser movido por una unidad mecánica que tiene el nombre `STN_1`:

```
PERS wobjdata wobj_stn1 := [FALSE, FALSE, "STN_1",
                             [[0,0,0],[1,0,0,0]], [[0,0,250],[1,0,0,0]]];
```

Ejemplo 2

El robot `ROB_1` está soldando una pieza sujeta por el robot `ROB_2`. El objeto de trabajo es movido por el robot `ROB_2`.

Al declarar el objeto de trabajo en el `ROB_1`, el argumento `robhold` debe cambiarse a `FALSE`, dado que `robhold TRUE` sólo se usa para herramientas fijas. En el caso de `ROB_2`, puede estar activo cualquier objeto de trabajo, ya que sólo los ángulos de los ejes de `ROB_2` se coordinan con el objeto de trabajo de `ROB_1`.

```
PERS wobjdata wobj_rob1 := [FALSE, FALSE, "ROB_2",
                             [[0,0,0],[1,0,0,0]], [[0,0,250],[1,0,0,0]]];
```

6.4 Movimientos independientes

6.4.1 Acerca de los movimientos independientes

En qué consisten los movimientos independientes

Si los distintos programas de tarea (y sus robots) trabajan independientemente, no se necesita ninguna sincronización o coordinación. En este caso, cada programa de tarea se escribe como si se tratara del programa de un sistema con un solo robot.

Otras dependencias distintas del movimiento

En ocasiones, incluso si no es necesario coordinar los movimientos, los programas de tarea presentan dependencias. Por ejemplo, si un robot deposita un objeto que será recogido por otro robot, el primer robot debe terminar con el objeto antes de que el segundo robot pueda utilizarlo.

Estas interacciones pueden resolverse con ayuda de:


- La instrucción `WaitSyncTask`
- Señales de E/S
- Variables persistentes junto con `WaitUntil`

Consulte la sección Multitasking del manual *Application manual - Controller software IRC5*.

6.4.2 Ejemplo "UnsyncArc" con movimientos independientes

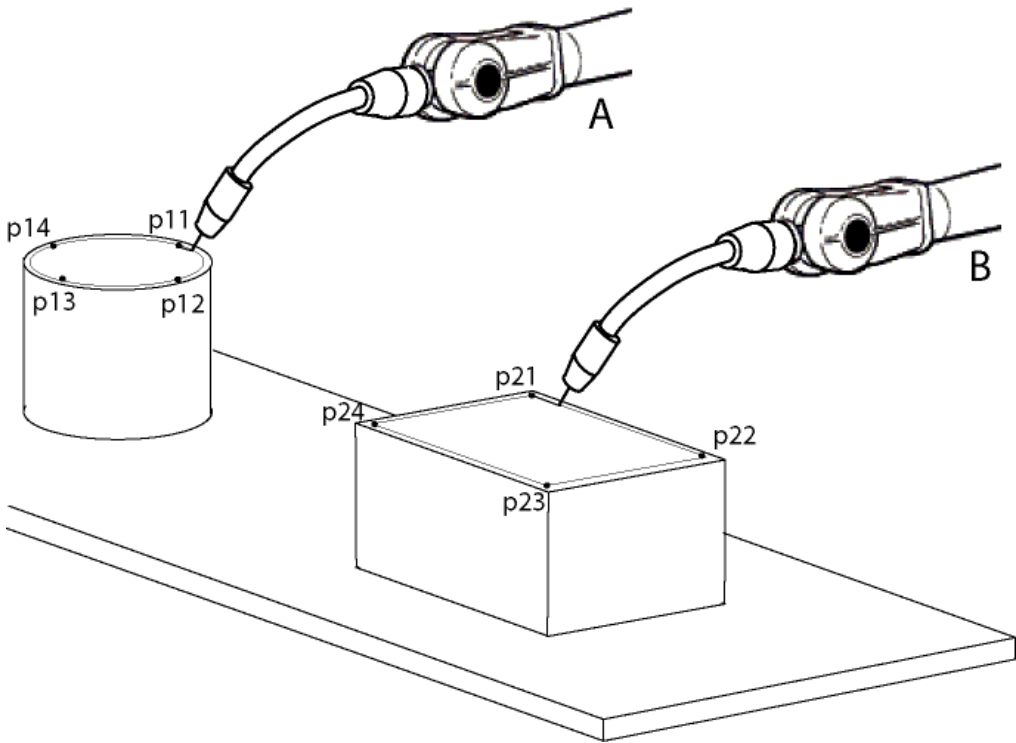
Descripción del programa

En este ejemplo, un robot suelda un círculo sobre un objeto mientras el otro robot suelda un cuadrado sobre otro objeto.

 **Nota**

Para que este ejemplo resulte sencillo y general, se utilizan instrucciones de movimiento normales (por ejemplo, `MoveL`) en lugar de usar instrucciones de soldadura (por ejemplo, `ArcL`). Para obtener más información acerca de la soldadura al arco, consulte *Application manual - Arc and Arc Sensor*.

Figura



xx0300000603

A	Robot 1
B	Robot 2

Programa de tarea T_ROB1

```
MODULE module1
  TASK PERS wobjdata wobj1 :=
    [ FALSE, TRUE, "",
      [ [500, -200, 1000], [1, 0, 0, 0] ],
      [ [100, 200, 100], [1, 0, 0, 0] ] ];
  TASK PERS tooldata tool1 := ...
  CONST robtarget p11 := ...
```

Continúa en la página siguiente

6 Programación

6.4.2 Ejemplo "UnsyncArc" con movimientos independientes

Continuación

```
...
CONST robtarget p14 := ...

PROC main()
...
IndependentMove;
...
ENDPROC

PROC IndependentMove()
MoveL p11, v500, fine, tool1\WObj:=wobj1;
MoveC p12, p13, v500, z10, tool1\WObj:=wobj1;
MoveC p14, p11, v500, fine, tool1\WObj:=wobj1;
ENDPROC
ENDMODULE
```

Tarea de programa T_ROB2

```
MODULE module2
TASK PERS wobjdata wobj2 :=
[ FALSE, TRUE, "",
[ [500, -200, 1000], [1, 0, 0, 0] ],
[ [100, 1200, 100], [1, 0, 0, 0] ] ];
TASK PERS tooldata tool2 := ...
CONST robtarget p21 := ...
...
CONST robtarget p24 := ...

PROC main()
...
IndependentMove;
...
ENDPROC

PROC IndependentMove()
MoveL p21, v500, fine, tool2\WObj:=wobj2;
MoveL p22, v500, z10, tool2\WObj:=wobj2;
MoveL p23, v500, z10, tool2\WObj:=wobj2;
MoveL p24, v500, z10, tool2\WObj:=wobj2;
MoveL p21, v500, fine, tool2\WObj:=wobj2;
ENDPROC
ENDMODULE
```

6.5 Movimientos semicoordinados

6.5.1 Acerca de los movimientos semicoordinados

En qué consisten los movimientos semicoordinados

Varios robots pueden trabajar con el mismo objeto de trabajo sin movimientos sincronizados siempre y cuando el objeto de trabajo no esté en movimiento.

Un posicionador puede mover el objeto de trabajo cuando los robots no están coordinados con él y los robots pueden ser coordinados con el objeto de trabajo cuando éste no está en movimiento. El cambio entre el movimiento del objeto y la coordinación de los robots se conoce como movimientos semicoordinados.

Implementación

Los movimientos semicoordinados requieren la sincronización entre los programas de tarea (por ejemplo, una instrucción *WaitSyncTask*). El posicionador debe saber cuándo es posible mover el objeto de trabajo y los robots deben saber cuándo pueden trabajar en el objeto de trabajo. Sin embargo, no es obligatorio que todas las instrucciones de movimiento estén sincronizadas.

Ventajas

La ventaja es que cada robot puede trabajar en el objeto de trabajo de forma independiente. Si hay robots diferentes que realizan asignaciones muy diferentes, esta posibilidad puede suponer un ahorro en el tiempo de ciclo en comparación con la coordinación de todos los movimientos de los robots.

6.5.2 Ejemplo "SyncArc" con movimientos semicoordinados

Descripción del programa

En este ejemplo se desea realizar la soldadura de un pequeño cuadrado y una línea larga en un lado del objeto. En el otro lado del objeto, se desea realizar un cuadrado y un círculo.

El posicionador pondrá en primer lugar el objeto de trabajo con el primer lado hacia arriba, mientras los robots permanecen en espera. A continuación, el robot 1 soldará una línea al mismo tiempo que el robot 2 suelda un cuadrado.

Cuando los robots han terminado las primeras operaciones de soldadura, permanecen en espera mientras el posicionador gira el objeto de trabajo de forma que el segundo lado quede orientado hacia arriba. A continuación, el robot 1 soldará un círculo al mismo tiempo que el robot 2 suelda un cuadrado.



¡AVISO!

Si el movimiento del objeto de trabajo y del robot no están separados con `WaitSyncTask` y puntos de paro, puede ocurrir lo siguiente:

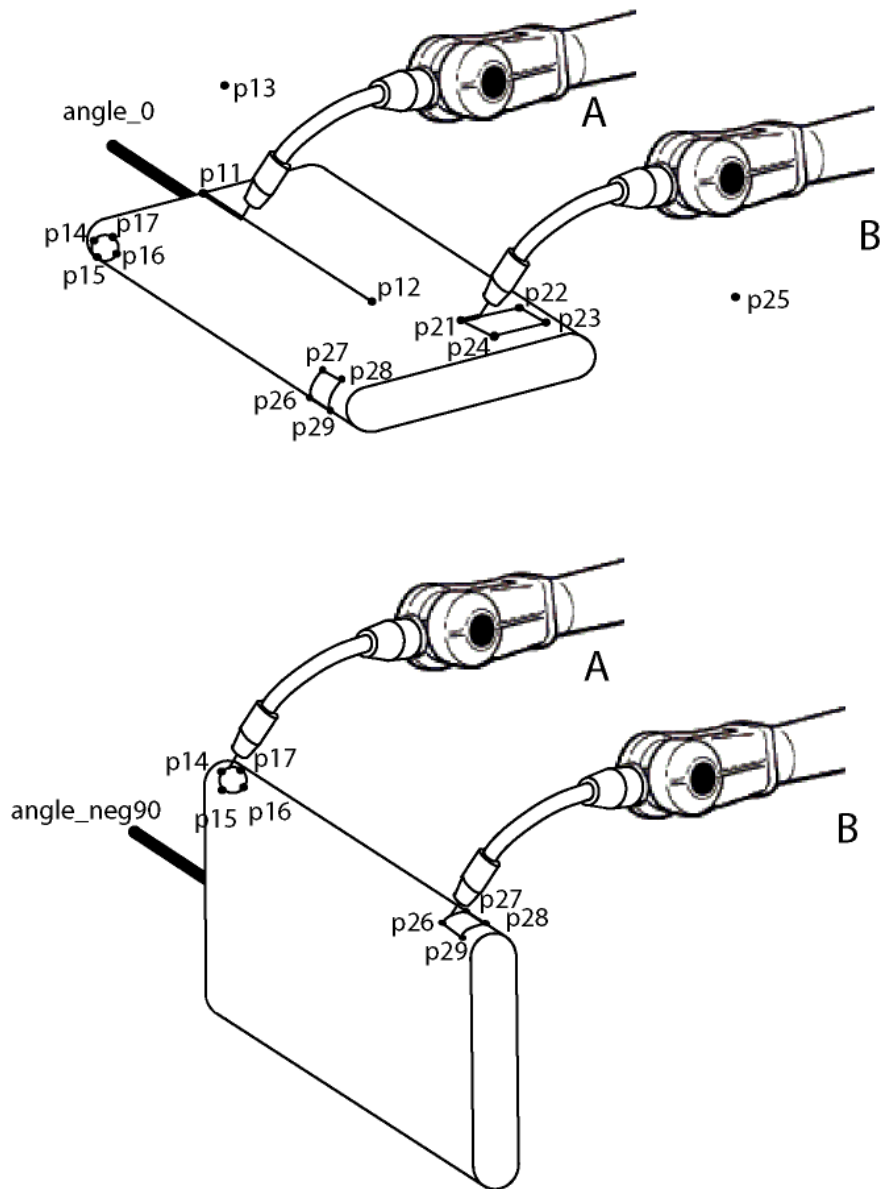
- Las unidades mecánicas controladas por las distintas tareas pueden colisionar.
- El robot retrocede paso a paso en el sentido incorrecto.
- El movimiento o la instrucción de reinicio pueden bloquearse.



Nota

Para que este ejemplo resulte sencillo y general, se utilizan instrucciones de movimiento normales (por ejemplo, `MoveL`) en lugar de usar instrucciones de soldadura (por ejemplo, `ArcL`). Para obtener más información acerca de la soldadura al arco, consulte *Application manual - Arc and Arc Sensor*.

Figura



xx0300000596

A	Robot 1
B	Robot 2

Programa de tarea T_ROB1

```
MODULE module1
  VAR syncident sync1;
  VAR syncident sync2;
  VAR syncident sync3;
  VAR syncident sync4;
  PERS tasks all_tasks{3} := [[ "T_ROB1", "T_ROB2", "T_STN1" ]];
  PERS wobjdata wobj_stn1 := [ FALSE, FALSE, "STN_1", [ [ 0, 0, 0 ],
    [ 1, 0, 0, 0 ] ], [ [ 0, 0, 250 ], [ 1, 0, 0, 0 ] ] ];
  TASK PERS tooldata tool1 := ...
```

Continúa en la página siguiente

6 Programación

6.5.2 Ejemplo "SyncArc" con movimientos semicoordinados

Continuación

```
CONST robtarget p11 := ...
...
CONST robtarget p17 := ...

PROC main()
...
SemiSyncMove;
...
ENDPROC

PROC SemiSyncMove()
! Wait for the positioner
WaitSyncTask sync1, all_tasks;
MoveL p11, v1000, fine, tool1 \WObj:=wobj_stn1;
MoveL p12, v300, fine, tool1 \WObj:=wobj_stn1;
! Move away from the object
MoveL p13, v1000, fine, tool1;
! Sync to let positioner move
WaitSyncTask sync2, all_tasks;
! Wait for the positioner
WaitSyncTask sync3, all_tasks;
MoveL p14, v1000, fine, tool1 \WObj:=wobj_stn1;
MoveC p15, p16, v300, z10, tool1 \WObj:=wobj_stn1;
MoveC p17, p14, v300, fine, tool1 \WObj:=wobj_stn1;
WaitSyncTask sync4, all_tasks;
MoveL p13, v1000, fine, tool1;
ENDPROC
ENDMODULE
```

Tarea de programa T_ROB2

```
MODULE module2
VAR syncident sync1;
VAR syncident sync2;
VAR syncident sync3;
VAR syncident sync4;
PERS tasks all_tasks{3} := [{"T_ROB1"}, {"T_ROB2"}, {"T_STN1"}];
PERS wobjdata wobj_stn1 := [ FALSE, FALSE, "STN_1", [ [0, 0, 0],
[1, 0, 0, 0] ], [ [0, 0, 250], [1, 0, 0, 0] ] ];
TASK PERS tooldata tool2 := ...
CONST robtarget p21 := ...
...
CONST robtarget p29 := ...

PROC main()
...
SemiSyncMove;
...
ENDPROC

PROC SemiSyncMove()
! Wait for the positioner
```

Continúa en la página siguiente

```

WaitSyncTask sync1, all_tasks;
MoveL p21, v1000, fine, tool2 \WObj:=wobj_stn1;
MoveL p22, v300, z10, tool2 \WObj:=wobj_stn1;
MoveL p23, v300, z10, tool2 \WObj:=wobj_stn1;
MoveL p24, v300, z10, tool2 \WObj:=wobj_stn1;
MoveL p21, v300, fine, tool2 \WObj:=wobj_stn1;
! Move away from the object
MoveL p25, v1000, fine, tool2;
! Sync to let positioner move
WaitSyncTask sync2, all_tasks;
! Wait for the positioner
WaitSyncTask sync3, all_tasks;
MoveL p26, v1000, fine, tool2 \WObj:=wobj_stn1;
MoveL p27, v300, z10, tool2 \WObj:=wobj_stn1;
MoveL p28, v300, z10, tool2 \WObj:=wobj_stn1;
MoveL p29, v300, z10, tool2 \WObj:=wobj_stn1;
MoveL p26, v300, fine, tool2 \WObj:=wobj_stn1;
WaitSyncTask sync4, all_tasks;
MoveL p25, v1000, fine, tool2;
ENDPROC
ENDMODULE

```

Programa de tarea T_STN1

```

MODULE module3
VAR syncident sync1;
VAR syncident sync2;
VAR syncident sync3;
VAR syncident sync4;
PERS tasks all_tasks{3} := [{"T_ROB1"}, {"T_ROB2"}, {"T_STN1"}];
CONST jointtarget angle_0 := [ [ 9E9, 9E9, 9E9, 9E9, 9E9, 9E9 ],
[ 0, 9E9, 9E9, 9E9, 9E9, 9E9 ] ];
CONST jointtarget angle_neg90 := [ [ 9E9, 9E9, 9E9, 9E9, 9E9,
9E9 ], [ -90, 9E9, 9E9, 9E9, 9E9, 9E9 ] ];

PROC main()
...
SemiSyncMove;
...
ENDPROC

PROC SemiSyncMove()
! Move to the wanted frame position. A movement of the
positioner is always required before the first semi
coordinated movement.
MoveExtJ angle_0, vrot50, fine;
! Sync to let the robots move
WaitSyncTask sync1, all_tasks;
! Wait for the robots
WaitSyncTask sync2, all_tasks;
MoveExtJ angle_neg90, vrot50, fine;
WaitSyncTask sync3, all_tasks;

```

Continúa en la página siguiente

6 Programación

6.5.2 Ejemplo "SyncArc" con movimientos semicoordinados

Continuación

```
        WaitSyncTask sync4, all_tasks;  
    ENDPROC  
ENDMODULE
```


6.5.3 Consideraciones y limitaciones al utilizar movimientos semicordinados

Reposo en una posición conocida

La unidad que controla el bastidor debe estar detenida en una posición conocida. Para obtener una posición conocida, solicite un movimiento a un punto fino.

Activar tarea

La unidad que controla el bastidor debe estar activada en el panel de selección de tareas de FlexPendant (consulte [Selección de tareas en la página 55](#)).

Puntos finos y WaitSyncTask antes y después del movimiento semicordinado

El movimiento semicordinado debe separarse en puntos finos e instrucciones WaitSyncTask antes y después del movimiento.

Funcionamiento con una ruta borrada

Cuando se utiliza alguna de las instrucciones enumeradas abajo, la ruta se elimina y la posición no puede ser leída por otras tareas.

- ActUnit
- DeactUnit
- ClearPath
- SyncMoveOn
- SyncMoveoff
- SyncMoveSuspend
- SyncMoveResume

Después de cualquiera de estas instrucciones, solicite un movimiento a una posición deseada para la unidad que controla el bastidor e inserte una instrucción WaitSyncTask antes del movimiento semicordinado.

Antes de cambiar el movimiento sincronizado con SyncMoveOn o SyncMoveResume, el movimiento semicordinado debe finalizarse con un punto fino y un WaitSyncTask.

Ejemplo de movimiento semicordinado y movimiento coordinado

```
!Example with semicordinated and synchronized movement
!Program example in task T_ROB1
PERS tasks task_list{2} := [ ["T_ROB1"], ["T_ROB2"] ];
PERS wobjdata rob2_obj:= [FALSE,FALSE,"ROB_2",
  [[0,0,0],[1,0,0,0]],[[155.241,-51.5938,57.6297],
  [0.493981,0.506191,-0.501597,0.49815]]];
VAR syncident sync0;
VAR syncident sync1;
VAR syncident sync2;
VAR syncident sync3;
VAR syncident sync4;

PROC main()
...
WaitSyncTask sync0, task_list;
```

Continúa en la página siguiente

```
MoveL p1_90, v100, fine, tcp1 \WObj:= rob2_obj;
WaitSyncTask sync1, task_list;
SyncMoveOn sync2, task_list;
MoveL p1_100 \ID:=10, v100, fine, tcp1 \WObj:= rob2_obj;
SyncMoveOff sync3;
!Wait until the movement has been finished in T_ROB2
WaitSyncTask sync3, task_list;
!Now a semicoordinated movement can be performed
MoveL p1_120, v100, z10, tcp1 \WObj:= rob2_obj;
MoveL p1_130, v100, fine, tcp1 \WObj:= rob2_obj;
WaitSyncTask sync4, task_list;
...
ENDPROC

!Program example in task T_ROB2
PERS tasks task_list{2} := [ ["T_ROB1"], ["T_ROB2"] ];
VAR syncident sync0;
VAR syncident sync1;
VAR syncident sync2;
VAR syncident sync3;
VAR syncident sync4;

PROC main()
...
MoveL p_fine, v1000, fine, tcp2;
WaitSyncTask sync0, task_list;
!Wait until the movement in T_ROB1 task is finished
WaitSyncTask sync1, task_list;
SyncMoveOn sync2, task_list;
MoveL p2_100 \ID:=10, v100, fine, tcp2;
SyncMoveOff sync3;
!The path has been removed at SyncMoveOff
!Perform a movement to wanted position for the object to make
the position available for other tasks
MoveL p2_100, v100, fine, tcp2;
WaitSyncTask sync3, task_list;
WaitSyncTask sync4, task_list;
MoveL p2_110, v100, z10, tcp2;
...
ENDPROC
```

Al cambiar entre el movimiento semicordinado y el movimiento sincronizado, se requiere una instrucción `WaitSyncTask` (si se utiliza la identidad `sync1`).

Al cambiar entre el movimiento sincronizado y el movimiento semicordinado, la tarea que mueve el objeto de trabajo (`rob2_obj`) debe moverse hasta la posición deseada. A continuación se requiere `WaitSyncTask` (identidad `sync3`) para poder realizar el movimiento semicordinado.

6.6 Movimientos coordinados sincronizados

6.6.1 Acerca de los movimientos coordinados sincronizados

En qué consisten los movimientos coordinados sincronizados

Varios robots pueden trabajar en un mismo objeto de trabajo en movimiento.

El posicionador o robot que sostiene el objeto de trabajo y los robots que trabajan con el objeto de trabajo deben tener movimientos sincronizados. Esto significa que los programas de tarea de RAPID, que gestionan una unidad mecánica cada uno, ejecutan simultáneamente sus instrucciones de movimiento.

Implementación

El modo de movimiento sincronizado se inicia ejecutando una instrucción `SyncMoveOn` en cada programa de tarea. El modo de movimiento sincronizado se finaliza ejecutando una instrucción `SyncMoveOff` en cada programa de tarea. El número de instrucciones de movimiento ejecutadas entre `SyncMoveOn` y `SyncMoveOff` debe ser el mismo en todos los programas de tarea.

Ventajas

Los movimientos coordinados sincronizados suelen reducir los tiempos de ciclo dado que los robots no tienen que esperar mientras se mueve el objeto de trabajo. También permite a los robots cooperar de formas que de otra manera serían difíciles o imposibles de conseguir.

Limitaciones


Sólo puede utilizar los movimientos coordinados sincronizados si cuenta con la opción MultiMove Coordinated de RobotWare.

6.6.2 Ejemplo "SyncArc" con movimientos coordinados

Descripción del programa

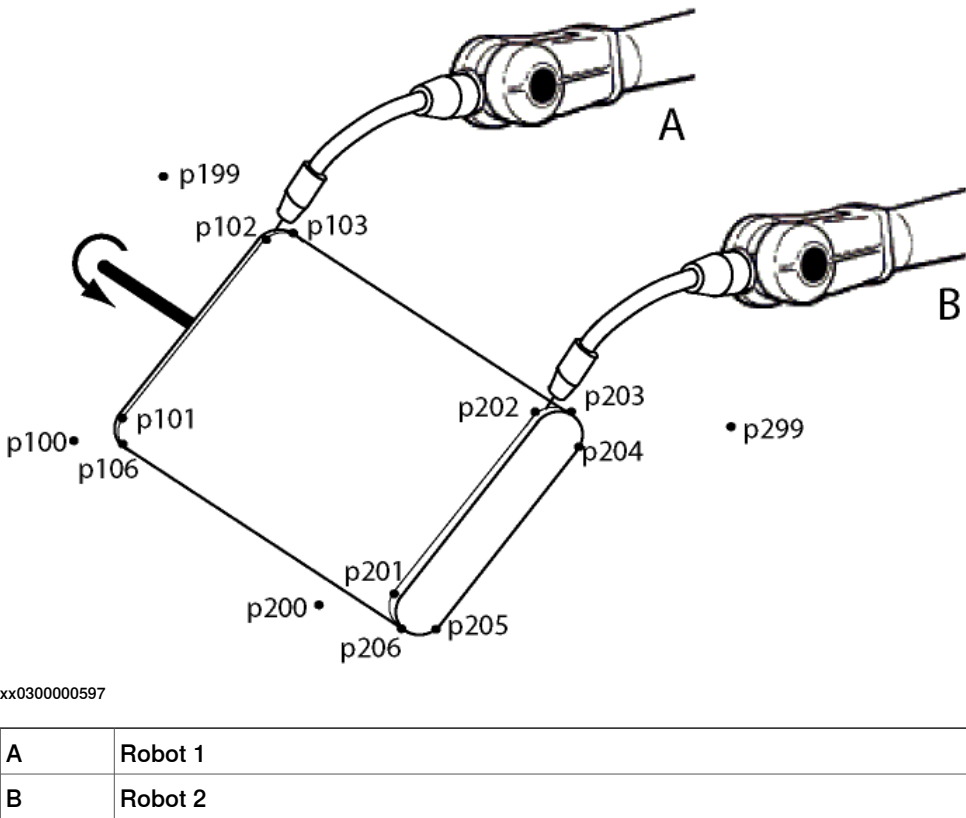
En este ejemplo, el objetivo es que los dos robots suelden una línea continua alrededor de todo el objeto.

Los TCPs de los robots están programados para describir una trayectoria circular respecto del objeto de trabajo. Sin embargo, dado que el objeto de trabajo está girando, los robots permanecerán casi inmóviles mientras el objeto de trabajo gira.

 **Nota**

Para que este ejemplo resulte sencillo y general, se utilizan instrucciones de movimiento normales (por ejemplo, MoveL) en lugar de usar instrucciones de soldadura (por ejemplo, ArcL). Para obtener más información acerca de la soldadura al arco, consulte *Application manual - Arc and Arc Sensor*.

Figura



Programa de tarea T_ROB1

```
MODULE module1
  VAR syncident sync1;
  VAR syncident sync2;
  VAR syncident sync3;
  PERS tasks all_tasks{3} := [{"T_ROB1"}, {"T_ROB2"}, {"T_STN1"}];
```

Continúa en la página siguiente

6.6.2 Ejemplo "SyncArc" con movimientos coordinados

Continuación

```

PERS wobjdata wobj_stn1 := [ FALSE, FALSE, "STN_1", [ [0, 0, 0],
    [1, 0, 0, 0] ], [ [0, 0, 250], [1, 0, 0, 0] ] ];
TASK PERS tooldata tool1 := ...
CONST robtarget p100 := ...
...
CONST robtarget p199 := ...

PROC main()
    ...
    SyncMove;
    ...
ENDPROC

PROC SyncMove()
    MoveJ p100, v1000, z50, tool1;
    WaitSyncTask sync1, all_tasks;
    MoveL p101, v500, fine, tool1 \WObj:=wobj_stn1;
    SyncMoveOn sync2, all_tasks;
    MoveL p102\ID:=10, v300, z10, tool1 \WObj:=wobj_stn1;
    MoveC p103, p104\ID:=20, v300, z10, tool1 \WObj:=wobj_stn1;
    MoveL p105\ID:=30, v300, z10, tool1 \WObj:=wobj_stn1;
    MoveC p106, p101\ID:=40, v300, fine, tool1 \WObj:=wobj_stn1;
    SyncMoveOff sync3;
    MoveL p199, v1000, fine, tool1;
UNDO
    SyncMoveUndo;
ENDPROC
ENDMODULE

```

Tarea de programa T_ROB2

```

MODULE module2
    VAR syncident sync1;
    VAR syncident sync2;
    VAR syncident sync3;
    PERS tasks all_tasks{3} := [{"T_ROB1"}, {"T_ROB2"}, {"T_STN1"}];
    PERS wobjdata wobj_stn1 := [ FALSE, FALSE, "STN_1", [ [0, 0, 0],
        [1, 0, 0, 0] ], [ [0, 0, 250], [1, 0, 0, 0] ] ];
    TASK PERS tooldata tool2 := ...
    CONST robtarget p200 := ...
    ...
    CONST robtarget p299 := ...

    PROC main()
        ...
        SyncMove;
        ...
    ENDPROC

    PROC SyncMove()
        MoveJ p200, v1000, z50, tool2;
        WaitSyncTask sync1, all_tasks;

```

Continúa en la página siguiente

6 Programación

6.6.2 Ejemplo "SyncArc" con movimientos coordinados

Continuación

```
MoveL p201, v500, fine, tool2 \WObj:=wobj_stn1;
SyncMoveOn sync2, all_tasks;
MoveL p202\ID:=10, v300, z10, tool2 \WObj:=wobj_stn1;
MoveC p203, p204\ID:=20, v300, z10, tool2 \WObj:=wobj_stn1;
MoveL p205\ID:=30, v300, z10, tool2 \WObj:=wobj_stn1;
MoveC p206, p201\ID:=40, v300, fine, tool2 \WObj:=wobj_stn1;
SyncMoveOff sync3;
MoveL p299, v1000, fine, tool2;
UNDO
SyncMoveUndo;
ENDPROC
ENDMODULE
```

Programa de tarea T_STN1

```
MODULE module3
VAR syncident sync1;
VAR syncident sync2;
VAR syncident sync3;
PERS tasks all_tasks{3} := [{"T_ROB1"}, {"T_ROB2"}, {"T_STN1"}];
CONST jointtarget angle_neg20 := [ [ 9E9, 9E9, 9E9, 9E9, 9E9,
    9E9], [ -20, 9E9, 9E9, 9E9, 9E9, 9E9] ];
...
CONST jointtarget angle_340 := [ [ 9E9, 9E9, 9E9, 9E9, 9E9, 9E9],
    [ 340, 9E9, 9E9, 9E9, 9E9, 9E9] ];

PROC main()
...
SyncMove;
...
ENDPROC

PROC SyncMove()
MoveExtJ angle_neg20, vrot50, fine;
WaitSyncTask sync1, all_tasks;
! Wait for the robots
SyncMoveOn sync2, all_tasks;
MoveExtJ angle_20\ID:=10, vrot100, z10;
MoveExtJ angle_160\ID:=20, vrot100, z10;
MoveExtJ angle_200\ID:=30, vrot100, z10;
MoveExtJ angle_340\ID:=40, vrot100, fine;
SyncMoveOff sync3;
UNDO
SyncMoveUndo;
ENDPROC
ENDMODULE
```

6.7 Ejecución de programas

6.7.1 Zonas de esquina

Zonas de esquina y WaitSyncTask

Es posible utilizar zonas de esquina al sincronizar varios programas de tarea con `WaitSyncTask`.

Zonas de esquina y movimientos sincronizados

Es necesario utilizar puntos de paro tanto al iniciar los movimientos sincronizados con `SyncMoveOn` como antes de finalizarlos con `SyncMoveOff`. Por otro lado, todas las demás instrucciones de movimiento ejecutadas entre `SyncMoveOn` y `SyncMoveOff` pueden utilizar zonas de esquina.

Dependencias entre instrucciones sincronizadas

En el modo de movimientos sincronizados, todas las instrucciones de movimiento simultáneas deben programarse con o sin zonas de esquina. Esto significa que todas las instrucciones de movimiento que tengan la misma `ID` deben tener zonas de esquina o que todas deben tener puntos de paro. Si una instrucción de movimiento con zona de esquina y una instrucción de movimiento con punto de paro se ejecutan al mismo tiempo en sus programas de tarea respectivos, se producirá un error.

Las instrucciones de movimiento ejecutadas a la vez pueden tener zonas de esquina de tamaños diferentes (por ejemplo, una puede utilizar `z10` y otra puede usar `z50`).

Zonas de esquina convertidas en puntos de paro

Una zona de esquina se convierte en un punto de paro si el programa de tarea tiene que esperar a otro programa de tarea. Esto puede producirse si la instrucción `WaitSyncTask` se ejecuta en una zona de esquina pero un programa de tarea alcanza esta instrucción más tarde que los demás.

Ejemplo con zonas de esquina

Partiendo del código de RAPID que aparece a continuación, ocurrirá lo siguiente:

- Si robot1 alcanza p11 aproximadamente al mismo tiempo que robot2 alcanza p21, los dos robots se sincronizarán en las zonas de esquina (p11 y p21).
- Si robot1 alcanza el punto p11 antes de que robot2 alcance p21, p11 se convertirá en un punto de paro.
- Si robot2 alcanza el punto p21 antes de que robot1 alcance p11, p21 se convertirá en un punto de paro.

Observe que tanto las instrucciones de movimiento con zonas de esquina como las instrucciones de movimiento con puntos de paro pueden usarse en cada tarea. Sólo es necesario asegurarse de que las instrucciones que tengan la misma `ID`

Continúa en la página siguiente

6 Programación

6.7.1 Zonas de esquina

Continuación

en ambos programas de tarea sean del mismo tipo. Las instrucciones que precedan a **SyncMoveOn** y **SyncMoveOff** deben tener puntos de paro.

Parte del programa de tarea T_ROB1:

```
MoveL p11, v500, z50, tool1;  
WaitSyncTask sync1, all_tasks;  
MoveL p12, v500, fine, tool1;  
SyncMoveOn sync2, all_tasks;  
MoveL p13\ID:=10, v500, z50, tool1 \WObj:=wobj_stn1;  
MoveL p14\ID:=20, v500, fine, tool1 \WObj:=wobj_stn1;  
SyncMoveOff sync3;  
MoveL p15, v500, fine, tool1;
```

Parte del programa de tarea T_ROB2:

```
MoveL p21, v500, z50, tool2;  
WaitSyncTask sync1, all_tasks;  
MoveL p22, v500, fine, tool2;  
SyncMoveOn sync2, all_tasks;  
MoveL p23\ID:=10, v500, z10, tool2 \WObj:=wobj_stn1;  
MoveL p24\ID:=20, v500, fine, tool2 \WObj:=wobj_stn1;  
SyncMoveOff sync3;  
MoveL p25, v500, fine, tool2;
```


6.7.2 Comportamiento de la sincronización

Punto de sincronización

Cuando un programa de tarea alcanza un punto de sincronización, esperará hasta que todos los programas de tarea hayan alcanzado el mismo punto de sincronización.

Son puntos de sincronización:

- Todas las instrucciones `WaitSyncTask`
- Todas las instrucciones `SyncMoveOn`
- Todas las instrucciones `SyncMoveOff`
- Todas las instrucciones de movimiento situadas entre `SyncMoveOn` y `SyncMoveOff`

Cuando un programa de tarea alcanza una instrucción `WaitSyncTask`, `SyncMoveOn` o `SyncMoveOff`, esperará hasta que todos los programas de tarea hayan alcanzado la instrucción que tenga la misma variable `syncident`.

Todas las instrucciones de movimiento ejecutadas entre las instrucciones `SyncMoveOn` y `SyncMoveOff` deben usar el argumento `ID`. Cuando un programa de tarea llega a una instrucción de movimiento de este tipo, esperará a que todos los programas hayan alcanzado la instrucción de movimiento cuyo argumento `ID` tenga el mismo valor.

Otras instrucciones distintas de las de movimiento

Todas las tareas de programa deben ejecutar el mismo número de instrucciones de movimiento entre las instrucciones `SyncMoveOn` y `SyncMoveOff`. Esto no afecta a las funciones u otras instrucciones distintas de las instrucciones de movimiento. Es posible utilizar cualquier número de funciones e instrucciones distintas de las instrucciones de movimiento.

Ejemplo

En este ejemplo, los dos programas de tarea ejecutan dos instrucciones de movimiento, pero una de ellas ejecuta otras instrucciones y funciones.

El robot 2 esperará y no avanzará hasta p21 hasta que robot 1 empiece a moverse hacia p11.

Dado que `SyncMoveOff` es un punto de sincronización, ambas tareas esperarán hasta que `dil` cambie a 1 antes de ejecutar `SyncMoveOff`.

Parte del programa de tarea T_ROB2:

```
SyncMoveOn sync1, all_tasks;
time := CTime();
Write log, "Synchronization started "\NoNewLine;
Write log, time;
MoveL p11\ID:=10, v500, fine, tool1 \WObj:=wobj_stn1;
Set dol;
MoveC p12, p13\ID:=20, v500, fine, tool1 \WObj:=wobj_stn1;
WaitDI dil, 1;
SyncMoveOff sync2;
```

Continúa en la página siguiente

6 Programación

6.7.2 Comportamiento de la sincronización

Continuación

Parte del programa de tarea T_ROB2:

```
SyncMoveOn sync1, all_tasks;  
MoveJ p21\ID:=10, v500, fine, tool2 \WObj:=wobj_stn1;  
MoveL p22\ID:=20, v500, fine, tool2 \WObj:=wobj_stn1;  
SyncMoveOff sync2;
```

6.7.3 Instrucciones vacías

Acerca de las instrucciones vacías

Es necesario ejecutar el mismo número de instrucciones de movimiento entre `SyncMoveOn` y `SyncMoveOff` en todos los programas de tarea. Si una instrucción de movimiento sólo se ejecuta en determinadas circunstancias, el número de instrucciones de movimiento puede ser distinto del de otros programas de tarea. Esto puede resolverse añadiendo una instrucción de movimiento hacia el punto en el que ya se encuentra el robot (una instrucción vacía), para el caso de que la instrucción de movimiento original no se ejecute.

Ejemplo con instrucciones de movimiento vacías

En este ejemplo, el programa de tarea necesita ejecutar dos instrucciones de movimiento si `di1` tiene el valor 1. Si `di1` tiene el valor 0, se ejecutan dos instrucciones de movimiento para mover el robot hacia la posición en la que ya se encuentra (instrucciones vacías).

Parte de un programa de tarea

```
SyncMoveOn sync1, all_tasks;
MoveL p1\ID:=10, v500, fine, tool1 \WObj:=wobj_stn1;
IF di1=1 THEN
    ! Instructions executed under certain conditions
    MoveL p2\ID:=20, v500, fine, tool1 \WObj:=wobj_stn1;
    MoveL p1\ID:=30, v500, fine, tool1 \WObj:=wobj_stn1;
ELSE
    ! Add dummy move instructions
    MoveL p1\ID:=20, v500, fine, tool1 \WObj:=wobj_stn1;
    MoveL p1\ID:=30, v500, fine, tool1 \WObj:=wobj_stn1;
ENDIF
SyncMoveOff sync2;
```

6.7.4 Principios de movimiento

Velocidades de robot

Cuando los movimientos de varios robots están sincronizados, todos los robots ajustan sus velocidades para terminar sus movimientos al mismo tiempo. Esto significa que el movimiento del robot que requiera más tiempo determina la velocidad de los demás robots.

Ejemplo de velocidades de robot

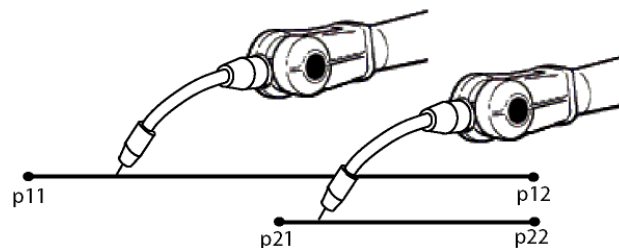
En este ejemplo, la distancia existente entre p11 y p12 es de 1.000 mm y la que separa a p21 de p22 es de 500 mm. Al ejecutar el código siguiente, robot1 se desplaza 1.000 mm a una velocidad de 100 mm/s. Dado que este recorrido requerirá 10 segundos, robot2 se moverá 500 mm en 10 segundos. La velocidad de robot2 será de 50 mm/s (y no de 500 mm/s como se ha programado).

Parte del programa de tarea T_ROB1:

```
MoveJ p11, v1000, fine, tool1;  
SyncMoveOn sync1, all_tasks;  
MoveL p12\ID:=10, v100, fine, tool1;
```

Parte del programa de tarea T_ROB2:

```
MoveJ p21, v1000, fine, tool2;  
SyncMoveOn sync1, all_tasks;  
MoveL p22\ID:=10, v500, fine, tool2;
```



xx0400000907

6.7.5 Modificación de posiciones

Acerca de la modificación de posiciones

Es posible modificar con ayuda del FlexPendant una posición programada. Consulte [Ventana Producción en la página 53](#).

Modificación de posiciones en el modo no sincronizado

Cuando los movimientos de las distintas tareas no están sincronizados, la posición de cada unidad mecánica se modifica de forma individual.

Modificación de posiciones en el modo de movimiento sincronizado

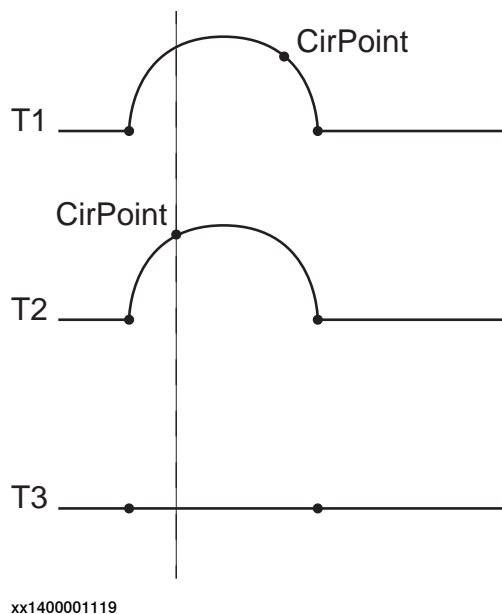
La modificación de posiciones en el modo de movimiento sincronizado (cuando la ejecución se encuentra entre una instrucción `SyncMoveOn` y una instrucción `SyncMoveOff`) se comporta de una forma distinta en función de si se realiza desde la ventana Producción o con el Editor de programas.

En la ventana Producción, la posición se modifica en todas las tareas pertenecientes al modo de movimiento sincronizado. Los puntos de círculo no pueden ser modificados desde la ventana Producción durante el modo de movimiento sincronizado. Por tanto, si el punto marcado es un punto de círculo, la función de modificación de posiciones de la ventana Producción no estará habilitada. En la ventana Producción, la posición sólo puede modificarse para la instrucción de movimiento actual (en la que se encuentra el puntero de movimiento).

En el Editor de programas, la posición se modifica sólo en el programa de tarea que esté abierto en ese momento en la ventana del editor.

Consulte también el ejemplo sobre el movimiento circular en la descripción de la modificación de posiciones en el *Manual del operador - IRC5 con FlexPendant*.

Modificación de posiciones circulares en el modo de movimiento sincronizado



6.7.6 Movimiento del puntero de programa

Movimiento del puntero de programa en el modo no sincronizado

Cuando ninguna de las tareas está en el modo de movimiento sincronizado, un puntero de programa en una tarea puede moverse sin que otras tareas se vean afectadas.




Movimiento del puntero de programa en el modo de movimiento sincronizado

Si se mueve el puntero de programa de una tarea, se pierden los punteros de programa de todas las tareas que se encuentren en el modo de movimiento sincronizado. Esto ocurre incluso si la tarea cuyo puntero de programa se mueve no está en el modo de movimiento sincronizado. Incluso si una tarea está inactiva, mover su puntero de programa afectará a los punteros de programa de todas las tareas en el modo de movimiento sincronizado.

Ejemplo

En este ejemplo existen tres tareas. Las tareas Task2 y Task3 están en el modo de movimiento sincronizado, mientras que Task1 funciona independientemente. En esta situación, el usuario toca **Move PP a Main** para la tarea Task1.

Los punteros de programa de Task2 y Task3 se pierden.

Task1:	Task2:	Task3:
MoveL p11 ...	MoveL p21 ...	MoveL p31 ...
MoveL p12 ...	SyncMoveOn sync1 ...	SyncMoveOn sync1 ...
MoveJ p13 ...	 MoveL p22 ...	 MoveL p32 ...
MoveL p14 ...	MoveL p23 ...	MoveL p33 ...
 MoveL p15 ...	SyncMoveOff sync2;	SyncMoveOff sync2;

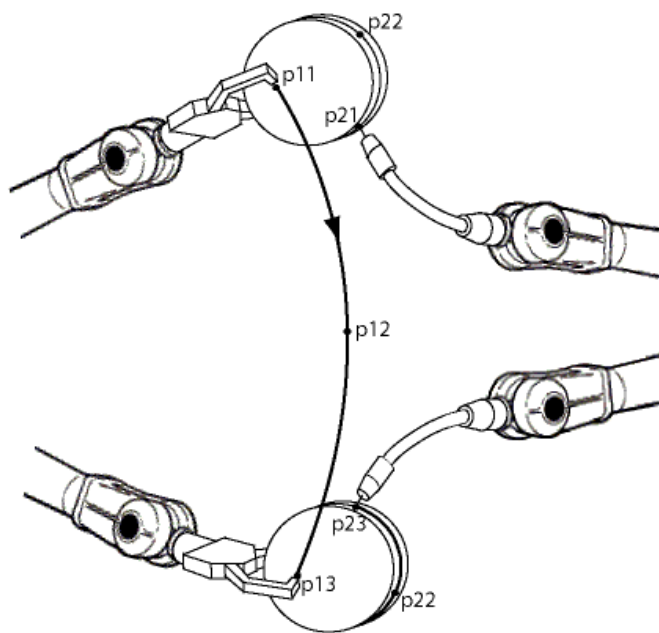
xx0500001444

6.7.7 Orientación de la herramienta en los movimientos circulares

Instrucciones de movimiento circular coordinado

Existe el riesgo de que la orientación de la herramienta sea incorrecta si dos programas de tarea coordinados ejecutan instrucciones de movimiento circular sincronizado a la vez. Si uno de los robots sostiene un objeto de trabajo en el que trabaja otro robot, la interpolación de círculo afecta a los dos robots. El punto del círculo debe alcanzarse al mismo tiempo en las dos trayectorias circulares para evitar la orientación incorrecta de la herramienta.

Ejemplo



xx0400000717

Si p12 se encontrara al comienzo de su trayectoria circular (más cerca de p11 que de p13) y p22 se encontrara al final de su trayectoria circular (más cerca de p23 que de p21), la orientación de la herramienta podría llegar a ser incorrecta. Si los puntos p12 y p22 se encuentran en una misma posición relativa en la trayectoria (en cuanto al porcentaje de la longitud de la trayectoria), la orientación de la herramienta seguirá siendo correcta.



Recomendación

Al modificar la posición del punto de círculo de los dos robots al mismo tiempo, se garantiza que la orientación de la herramienta sea la correcta. Esto significa que, en este ejemplo, debe ejecutar el programa paso a paso y, a continuación, modificar los puntos p12 y p22 al mismo tiempo.

6.7.8 Aplicaciones afectadas por MultiMove

Detección de colisiones

Si se detecta una colisión para un robot, todos los robots se detienen. Incluso si los robots funcionan individualmente, todos los robots se comportan como si hubieran sufrido la colisión.

Un motivo de este comportamiento es que cuando se detecta una colisión existe un riesgo elevado de que sean dos robots los que han colisionado. Otro motivo es que si uno de los robots se detiene pero otro sigue en funcionamiento, puede producirse una colisión adicional.

Zonas mundo

Una zona mundo declarada en un programa de tarea sólo es válida para las unidades mecánicas que pertenecen a la tarea. Para que una zona mundo afecte a todas las unidades mecánicas, es necesario declararla en todos los programas de tarea.

6.8 Recomendaciones de programación

6.8.1 Recomendaciones de programación

Declare syncident como global en la tarea

Mediante la declaración del tipo de dato `syncident` de forma global en el programa de tarea, se elimina el riesgo de que haya dos variables `syncident` con el mismo nombre en un mismo programa de tarea.

No reutilice syncident

Las variables `syncident` se utilizan como argumento de todas las instrucciones `WaitSyncTask`, `SyncMoveOn` y `SyncMoveOff`, de forma que el operador pueda distinguir qué instrucciones se ejecutan simultáneamente en los distintos programas de tarea. Si se utilizara una variable `syncident` como argumento de más de una instrucción por tarea, no sería posible identificar de forma exclusiva la instrucción. Para asegurarse de que su código de programa es comprensible, no reutilice nunca una misma variable `syncident`.

Declaración de herramientas, objetos de trabajo y cargas útiles

La declaración de una variable como `TASK PERS` hará que sea persistente en el programa de tarea, pero no compartida entre tareas. Al declarar las herramientas, los objetos de trabajo y las cargas útiles como persistentes de la tarea, no es necesario hacer un seguimiento de si el nombre de la variable se utiliza en otras tareas. Si las herramientas, los objetos de trabajo y las cargas útiles se declaran como `TASK PERS`, no es necesario cambiar sus nombres en caso de que se copie el programa o se reutilice en otra tarea.

En el caso de los objetos de trabajo que se utilizan en varios programas de tarea, es preferible declararlos como `PERS`. Las herramientas pueden ser declaradas como `PERS` si una tarea en segundo plano necesita leer la posición del robot.

Cambio del valor de una variable PERS

Una variable `PERS` declarada globalmente conserva su valor incluso si se carga una nueva declaración de la misma variable `PERS`. El valor de la variable `PERS` cargada en primer lugar se conserva, siempre y cuando haya cualquier referencia a dicha variable `PERS`.

Si desea reemplazar todos los programas de tarea con nuevos programas en los que los valores de la variable `PERS` es distinto, elimine primero todos los programas de tarea y cargue a continuación todos los nuevos programas de tarea. De esta forma, el valor anterior de la variable `PERS` se pierde, ya que se eliminan todas sus declaraciones.

El cambio del valor de una variable `PERS` desde la vista **Data Variable** (Variable de datos) del FlexPendant y el guardado del programa actualizan la variable `PERS` de la forma correcta.

Continúa en la página siguiente

6 Programación

6.8.1 Recomendaciones de programación

Continuación

Utilice SyncMoveUndo

Utilice siempre un gestor UNDO con una instrucción SyncMoveUndo en los procedimientos que tengan movimientos sincronizados (es decir, que tengan una instrucción SyncMoveOn).

Después de una instrucción SyncMoveOn, los movimientos del programa de tarea se sincronizan con los movimientos de los otros programas de tarea. Si a continuación se mueve manualmente el puntero de programa antes de que se ejecute la instrucción SyncMoveOff, los movimientos se seguirán sincronizando. Esto puede evitarse utilizando un gestor UNDO que incluya una instrucción SyncMoveUndo.

Si se mueve manualmente el puntero de programa a un punto que está fuera de un procedimiento, se llama al gestor UNDO que corresponde al procedimiento. La instrucción SyncMoveUndo finalizará la sincronización si los movimientos están sincronizados en ese momento. Si los movimientos no están sincronizados al mover el puntero de programa, SyncMoveUndo no hará nada. En otras palabras, el uso de SyncMoveUndo no supone ninguna desventaja, pero resulta muy útil si se mueve el puntero de programa.

Para obtener más información acerca de los gestores UNDO, consulte el *Manual de referencia técnica - Descripción general de RAPID*.

Coordinación respecto de un objeto de trabajo

La coordinación respecto de un objeto de trabajo movido por una unidad mecánica de otra tarea puede hacerse de dos formas:

- Todas las instrucciones de movimiento coordinadas con el objeto de trabajo deben ejecutarse cuando el objeto de trabajo se encuentra en reposo. Consulte [Movimientos semicoordinados en la página 67](#).
- El robot que está coordinado con el objeto de trabajo y la unidad mecánica que mueve el objeto de trabajo deben estar sincronizados en el modo de movimiento. Consulte [Movimientos coordinados sincronizados en la página 75](#).

No es posible realizar la coordinación respecto de un objeto de trabajo en movimiento, controlado desde otra tarea sin utilizar el modo de movimiento sincronizado.

Área de trabajo común

Si dos robots utilizan un área de trabajo común y no se encuentran en el modo de movimiento sincronizado, es necesario tener precauciones para evitar colisiones. Asegúrese de que sólo uno de los robots se encuentre en el área común en un momento determinado. Para ello, utilice lo siguiente:

- WaitSyncTask
- World Zones
- Señal de E/S

7 Recuperación en caso de errores de RAPID

7.1 Recuperación en caso de errores para MultiMove

Error en el modo no sincronizado

Si se produce un error durante el modo no sincronizado, se gestiona exactamente igual que en un sistema con un solo robot. Ningún otro programa de tarea se ve afectado por el error.

Error en el modo de movimiento sincronizado

Si se produce un error durante el modo de movimiento sincronizado, el programa de tarea que contiene el error se detendrá y generará un código de error, al igual que en los sistemas con un solo robot. Debido a la sincronización, las demás tareas no seguirán moviéndose. Una vez que el error ha sido resuelto, el movimiento puede continuar en todos los programas de tarea.

7 Recuperación en caso de errores de RAPID

7.2 Ejemplo sencillo de recuperación

7.2 Ejemplo sencillo de recuperación

Acerca de este ejemplo

En este ejemplo, una división entre cero provoca un error durante el modo de movimiento sincronizado. Dado que el gestor de errores puede resolver el error sin ninguna instrucción de movimiento, no es necesario que el gestor de errores tenga en cuenta la sincronización. El modo de movimiento sincronizado está activado todo el tiempo y la segunda instrucción de movimiento se inicia en los dos robots tan pronto como el gestor de errores ha terminado de ejecutarse. Si no es posible que se produzca ningún otro error, el programa de tarea T_HANDLEROB no necesita tener ningún gestor de errores.

Programa de tarea T_PROCROB

```
...
SyncMoveOn, sync1, motion_tasks;
MoveL p101\ID:=10, v100, z10, gun2 \WObj:=wobj_handlerob;
a:=3;
b:=0;
c:=a/b;
MoveL p102\ID:=20, v100, fine, gun2 \WObj:=wobj_handlerob;
SyncMoveOff sync2;
...
ERROR
  IF ERRNO = ERR_DIVZERO THEN
    b:=1;
    RETRY;
  ENDIF
```

Programa de tarea T_HANDLEROB

```
...
SyncMoveOn, sync1, motion_tasks;
MoveL p201\ID:=10, v100, z10, gripl;
MoveL p202\ID:=20, v100, fine, gripl;
SyncMoveOff sync2;
...
```

7.3 Errores generados asincrónicamente

En qué consiste un error generado asincrónicamente

Los errores generados asincrónicamente pueden ser generados por una instrucción distinta de la instrucción en la que se encuentra el puntero de programa. Esto significa que un error asíncrono puede ser generado mientras el robot se encuentra en medio de un movimiento por una trayectoria. Para obtener más información acerca de los errores generados asincrónicamente, consulte el manual *Technical reference manual - RAPID kernel*.

La técnica basada en los errores generados asincrónicamente permite que una instrucción fallida de un programa de tarea genere un error en todos los demás programas de tarea que tengan sincronizados sus movimientos.

Cómo generar un error asíncrono

La instrucción `ProcerrRecovery` genera el error `ERR_PATH_STOP` y detiene el movimiento de todos los programas de tarea que tengan sincronizados sus movimientos.

Los errores asíncronos también pueden ser generados por instrucciones de proceso (por ejemplo, `ArcL`). Éstas pueden generar un código de error (que describe la causa del error) en el programa de tarea en el que se produjo el error, además de generar el error `ERR_PATH_STOP` en los otros programas de tarea que tengan sincronizados sus movimientos.

Programas de tarea sin errores

Si dos programas de tarea ejecutan instrucciones de movimiento sincronizadas y uno de ellos genera un error asíncrono, los movimientos se detienen en las dos tareas. El programa de tarea en el que todo funcionaba correctamente recibirá el error `ERR_PATH_STOP`. Es necesario gestionar este error con un gestor de errores. El gestor de errores puede gestionar el error `ERR_PATH_STOP` con sólo esperar a que la otra tarea resuelva sus problemas, reanudando a continuación sus movimientos.

Al utilizar la instrucción `StartMoveRetry`, la ejecución continuará cuando todas las tareas alcancen esta instrucción.

Movimientos independientes en el gestor de errores

Si el gestor de errores de un programa de tarea necesita ejecutar una instrucción de movimiento, es necesario suspender en primer lugar la sincronización.

La sincronización se suspende automáticamente cuando se usa la instrucción `StorePath`. Todas las tareas cuyos movimientos estén sincronizados deben ejecutar una instrucción `StorePath` antes de desactivar la sincronización y que pueda continuar la ejecución.

La instrucción `RestoPath` devuelve la sincronización al modo que tenía antes de ejecutar `StorePath`. Todos los programas de tarea cuyos movimientos estén sincronizados deben ejecutar la instrucción `RestoPath` en sus gestores de errores antes de reanudar la sincronización y poder continuar la ejecución.

Continúa en la página siguiente

7 Recuperación en caso de errores de RAPID

7.3 Errores generados asíncronamente

Continuación

Entre las instrucciones `StorePath` y `RestoPath`, el programa de tarea que presenta el fallo puede moverse independientemente para resolver su problema. Dado que `RestoPath` hace las veces de un punto de sincronización, los demás programas de tarea esperarán en este punto hasta que se haya resuelto el problema.

Si el programa de tarea no se encuentra en el modo de movimiento sincronizado, `StorePath` y `RestoPath` funcionan del mismo modo que en un sistema con un solo robot. Esto significa que el mismo código de gestor de errores puede gestionar los errores que se produzcan tanto en el modo de movimiento sincronizado como en el modo no sincronizado.

`StorePath` y `RestoPath` requieren la opción `Path Recovery`. Para obtener más información acerca de `StorePath` y `RestoPath`, consulte el manual *Application manual - Controller software IRC5*.

7.4 Ejemplo de creación de un error generado asincrónicamente

Acerca de este ejemplo

En este ejemplo, se inicia un proceso cambiando `do_myproc` a 1. A continuación, se supervisa el proceso, y la señal `di_proc_sup` cambia a 1 si el proceso falla.

Si se produce un fallo en el proceso durante un movimiento del robot, una interrupción llama a una rutina TRAP. La instrucción `ProcerrRecovery` detiene el movimiento y genera el error `ERR_PATH_STOP` en todos los programas de tarea que tengan sus movimientos sincronizados.

El programa de tarea `T_HANDLEROB` debe tener un gestor de errores que reinicie el movimiento cuando el error haya sido resuelto en la tarea de programa `T_PROCROB`. Para ello se requiere solo una instrucción, `StartMoveRetry`.

Programa de tarea `T_PROCROB`

```
VAR intnum proc_sup_int;

PROC main()
...
SyncMoveOn, sync1, motion_tasks;
my_proc_on;
MoveL p101\ID:=10, v100, z10, gun1 \WObj:=wobj_handlerob;
MoveL p102\ID:=20, v100, fine, gun1 \WObj:=wobj_handlerob;
my_proc_off;
SyncMoveOff sync2;
...

ERROR
IF ERRNO = ERR_PATH_STOP THEN
    my_proc_on;
    StartMoveRetry;
ENDIF
ENDPROC

TRAP iprocfail
my_proc_off;
ProcerrRecovery \SyncLastMoveInst;
RETURN;
ENDTRAP

PROC my_proc_on()
SetDO do_myproc, 1;
CONNECT proc_sup_int WITH iprocfail;
ISignalDI di_proc_sup, 1, proc_sup_int;
ENDPROC

PROC my_proc_off()
SetDO do_myproc, 0;
IDelete proc_sup_int;
ENDPROC
```

Continúa en la página siguiente

7 Recuperación en caso de errores de RAPID

7.4 Ejemplo de creación de un error generado asincrónicamente

Continuación

Programa de tarea T_HANDLEROB

```
PROC main()  
  ...  
  SyncMoveOn, sync1, motion_tasks;  
  MoveL p201\ID:=10, v100, z10, gripl;  
  MoveL p202\ID:=20, v100, fine, gripl;  
  SyncMoveOff sync2;  
  ...  
  
  ERROR  
  IF ERRNO = ERR_PATH_STOP THEN  
    StartMoveRetry;  
  ENDIF  
ENDPROC
```


7.5 Ejemplo con movimientos en el gestor de errores

Acerca de este ejemplo

En este ejemplo, puede producirse un error asíncrono que hace que el robot se mueva a otra posición para resolver el error. La sincronización se suspende con `StorePath` en todas las tareas cuyos movimientos estén sincronizados y se restablece con `RestoPath`.

En este ejemplo se utiliza la instrucción `ArcL`. Esta instrucción gestiona el proceso de soldadura al arco, a la vez que actúa como una instrucción de movimiento. Para comprender este ejemplo, todo lo que necesita saber es que se trata de una instrucción de movimiento (parecida a `MoveL`) que puede dar lugar a errores asíncronos de proceso. Para obtener más información acerca de `ArcL`, consulte *Application manual - Arc and Arc Sensor* y *Manual de referencia técnica - Instrucciones, funciones y tipos de datos de RAPID*.



Nota

Observe que el programa de tarea `T_STN1` debe contener las instrucciones `StorePath` y `RestoPath`, incluso si no hay ningún código entre estas instrucciones. Ningún programa de tarea seguirá ejecutando su gestor de errores hasta que todos los programas de tarea ejecuten la instrucción `StorePath`.

Programa de tarea `T_ROB1`

```
...
SyncMoveOn, sync1, all_tasks;
ArcL p101\ID:=10, v100, seam1, weld1, weave1, z10, gun1
  \WObj:=wobj_stn1;
...
ERROR
  IF ERRNO=AW_WELD_ERR OR ERRNO=ERR_PATH_STOP THEN
    StorePath;
    IF ERRNO=AW_WELD_ERR THEN
      gun_cleaning;
    ENDIF
    RestoPath;
    StartMoveRetry;
  ENDIF
...
PROC gun_cleaning()
  VAR robtarget p199;
  p199 := CRobT(\Tool:=gun1 \WObj:=wobj0);
  MoveL pclean, v100, fine, gun1;
  ...
  MoveL p199, v100, fine, gun1;
ENDPROC
```

Tarea de programa `T_ROB2`

```
...
SyncMoveOn, sync1, all_tasks;
```

Continúa en la página siguiente

7 Recuperación en caso de errores de RAPID

7.5 Ejemplo con movimientos en el gestor de errores

Continuación

```
ArcL p201\ID:=10, v100, seam2, weld2, weave2, z10, gun2
  \WObj:=wobj_stn1;
...
ERROR
  IF ERRNO=AW_WELD_ERR OR ERRNO=ERR_PATH_STOP THEN
    StorePath;
    IF ERRNO=AW_WELD_ERR THEN
      gun_cleaning;
    ENDIF
    RestoPath;
    StartMoveRetry;
  ENDIF
...
PROC gun_cleaning()
  VAR robtarg p299;
  p299 := CRobT(\Tool:=gun2 \WObj:=wobj0);
  MoveL pclean, v100, fine, gun2;
  ...
  MoveL p299, v100, fine, gun2;
ENDPROC
```

Programa de tarea T_STN1

```
...
SyncMoveOn, sync1, all_tasks;
MoveExtJ angle_20\ID:=10, vrot50, z10;
...
ERROR
  IF ERRNO=ERR_PATH_STOP THEN
    StorePath;
    RestoPath;
    StartMoveRetry;
  ENDIF
...
```

8 Ejecución de un subconjunto del sistema MultiMove

8.1 Cómo continuar con una o varias unidades de accionamiento inactivas.

Descripción general

Es posible desconectar un Drive Module y seguir trabajando, por ejemplo:

- Si un Drive Module es necesario en otro emplazamiento debido a una avería o una situación similar.
- Para realizar ajustes durante la puesta en servicio, por ejemplo la programación de un robot cada vez mientras el resto están apagados temporalmente.

Asegúrese de que la aplicación permita que el trabajo continúe sin este Drive Module, usando el procedimiento [Continuación con la función Drive Module Disconnect en la página 99](#).

Sin embargo, si se cumple alguna de las siguientes condiciones, use el procedimiento [Continuación con una configuración alternativa en la página 100](#).

- La primera alternativa no funciona.
- Se utilizan interruptores de límite en el robot.
- Es necesario trasladar el Drive Module (por ejemplo para su reparación o su instalación en otra célula).



Recomendación

En ocasiones es necesario cambiar el programa y/o la configuración de forma que la aplicación funcione con un Drive Module menos.

Continuación con la función Drive Module Disconnect

Este procedimiento muestra cómo hacer que los robots que funcionan sigan con sus aplicaciones sin cambiar la configuración. Para ello se requiere que los robots que funcionan no tengan dependencias con el robot desconectado ni con ejes adicionales conectados al mismo Drive Module.

A continuación aparece una breve descripción de cómo desconectar el Drive Module. Para obtener una información más detallada, consulte *Manual del producto - IRC5*, sección *Conexiones - Conexión del Drive Module Disconnect*.

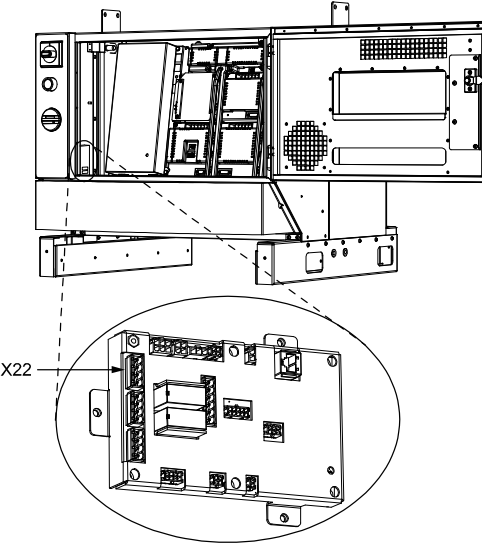
	Acción	Información/figura
1	Asegúrese de que el parámetro del sistema <code>Allow_Drive_Module_Disconnect</code> tiene el valor <code>true</code> .	
2	Cambie al modo manual.	
3	Asegúrese de que el controlador se encuentre en el estado <code>Motors Off</code> .	

Continúa en la página siguiente

8 Ejecución de un subconjunto del sistema MultiMove

8.1 Cómo continuar con una o varias unidades de accionamiento inactivas.

Continuación

	Acción	Información/figura
4	Desconecte el conector de X22 del Drive Module. Cuando se desconecta el conector, aparece el siguiente mensaje en el FlexPendant: Mensaje de evento 50320, Drive Module desconectado.	 <p>xx0500001599</p>
5	El sistema actúa ahora como si ninguna de las unidades mecánicas conectadas a este módulo de accionamiento existiera.	



Nota

En función del tipo de fallo, este método podría no funcionar. Por ejemplo, si se produce un error en el ordenador de ejes. En este caso, continúe con la configuración alternativa.

Continuación con una configuración alternativa

En este procedimiento se muestra cómo dejar que los robots en funcionamiento continúen con sus aplicaciones pero realizando cambios en la configuración estándar.

	Acción	Información/figura
1	Reinicie el controlador mediante el modo de reinicio Iniciar Boot Application .	
2	Apague el controlador.	
3	Localice el cable de conexión de Ethernet del Drive Module que desee desconectar. Retírelo de la tarjeta de comunicación de robot del Control Module. Recuerde que los cables de Ethernet del Drive Module deben estar conectados en el orden siguiente. Es necesario que no haya ningún hueco libre en esta secuencia: <ul style="list-style-type: none">• AXC1 en la tarjeta de comunicación del robot• ETHERNET 1 en la tarjeta Ethernet• ETHERNET 2 en la tarjeta Ethernet• ETHERNET 3 en la tarjeta Ethernet	Consulte Conexiones Ethernet en la página 21 .

Continúa en la página siguiente

8 Ejecución de un subconjunto del sistema MultiMove

8.1 Cómo continuar con una o varias unidades de accionamiento inactivas.

Continuación

	Acción	Información/figura
4	Localice el cable de conexión de señales de seguridad del Drive Module que desee desconectar. Desconéctelo de la tarjeta de panel del Control Module y sustitúyalo con un conector de puente. Traslade los cables de conexión de señales de seguridad para que no quede ningún hueco en el siguiente orden: X7, X8, X14 y X17.	Consulte Conexiones Ethernet en la página 21 .
5	Encienda la alimentación del controlador.	
6	Seleccione un nuevo sistema de robot que esté configurado sin contar con la unidad mecánica ahora desconectada. Recuerde que la configuración tiene que estar en concordancia con las conexiones del paso 3.	

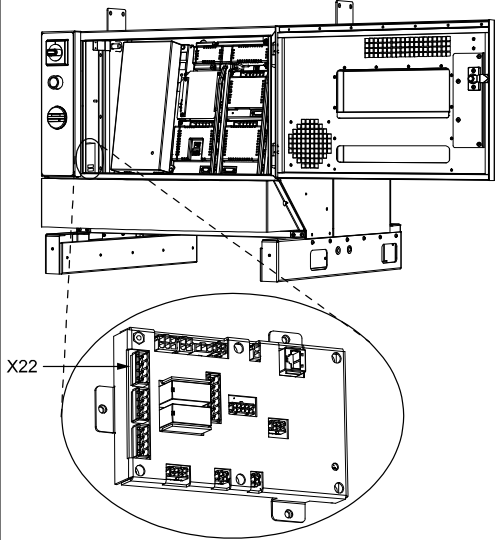
8 Ejecución de un subconjunto del sistema MultiMove

8.2 Ejecución de un subconjunto en los ejemplos "Unsync Arc"

8.2 Ejecución de un subconjunto en los ejemplos "Unsync Arc"

Ejemplo con Drive Module Disconnect

En este ejemplo, la configuración corresponde a "UnsyncArc" y se produce un error en los equipos de proceso del robot 1. La función Drive Module Disconnect está configurada y no hay interruptores de límite en los robots. El robot 2 debe continuar su trabajo.

	Acción	Información/figura
1	Asegúrese de que el parámetro del sistema <code>Allow_Drive_Module_Disconnect</code> tiene el valor <code>true</code> .	
2	Cambie al modo manual.	
3	Asegúrese de que el controlador se encuentre en el estado <code>Motors Off</code> .	
4	Desconecte el conector de X22 en el Drive Module 1.	 xx0500001599
5	Ahora el sistema actúa como si el robot 1 no existiera. Ahora puede continuar trabajando de forma segura con el robot 2.	

Ejemplo sin Drive Module Disconnect

En este ejemplo se utilizan interruptores de límite en los robots. La configuración está de acuerdo con "UnsyncArc" y se produce un error en el robot 1. El robot 2 debe continuar con su trabajo.

	Acción	Información/figura
1	Reinicie el controlador mediante el modo de reinicio Iniciar Boot Application .	
2	Apague el controlador.	

Continúa en la página siguiente

8 Ejecución de un subconjunto del sistema MultiMove

8.2 Ejecución de un subconjunto en los ejemplos "Unsync Arc"

Continuación

	Acción	Información/figura
3	Desconecte la conexión de Ethernet del Drive Module 1 de la tarjeta de comunicación del robot del Control Module y sustitúyala con un conector de puente. Traslade la conexión de Ethernet del Drive Module 2, de la tarjeta de Ethernet a la tarjeta de comunicación del robot.	Consulte Conexiones Ethernet en la página 21 .
4	Desconecte la conexión de señales de seguridad del Drive Module 1, del conector X7 de la tarjeta de panel del Control Module. Traslade la conexión de señales de seguridad del Control Module 2, de X8 a X7. A continuación, presione el conector de puente para introducirlo en el conector X8.	Consulte Conexiones Ethernet en la página 21 .
5	Encienda la alimentación del controlador.	
6	Seleccione un nuevo sistema de robot que esté configurado sin el robot 1. Recuerde que la configuración tiene que estar de acuerdo con las conexiones del paso 3. Es decir, el robot restante será conocido como robot 1.	



Recomendación

Estas mismas acciones pueden tomarse si el ordenador de ejes falla o si el fallo no puede resolverse con la función Drive Module Disconnect.

Esta página se ha dejado vacía intencionadamente

Índice

A

Activate at Start Up, 32, 37
Allow Drive Module Disconnect, 32
Allow move of user frame, 32, 37
aplicaciones de ejemplo, 16
Argument, 34, 39
Argument 2, 34, 39
argumento de sincronización, 61
avería de hardware, 99

B

barra de estado, 50

C

calibración, 41
calibración relativa, 42
conexiones de las señales de seguridad, 23, 26
Conexiones Ethernet, 21
configuración, 29
controlador, 19
Controller, 30
Controller, tema, 30
Control Module, 20
coordinación, 15
coordinado, 54
crear un sistema MultiMove, 27

D

Deactivation Forbidden, 32, 37
desactivar tareas, 55
Detección de colisiones, 88
Drive Module, 19, 25
Drive Module Disconnect, 99
Drive Module User Data, 32

E

ejemplo de programa, 65, 68, 76
ejemplo de RAPID, 65, 68, 76
ERR_PATH_STOP, 93
errores generados asincrónicamente, 93

F

FlexPendant, 49
funciones, 60

H

hardware, 19
hardware, avería, 99

I

I/O, 34
I/O, tema, 34
ID, 61
identno, 59
instalación, 19
instrucciones, 60
instrucciones de movimiento, 83
instrucciones vacías, 83
interfaz de usuario, 49
IsSyncMoveOn, 60

M

Mechanical Unit, 32, 37
Mechanical Unit Group, 30, 35, 37
menú de unidades mecánicas, 54

modificar posiciones, 85
modo automático, 57
Motion, 32
Motion, tema, 32
Motion Planner, 32, 35, 37
MotionTask, 30, 35, 37
MoveExtJ, 60
Mover PP a Main, 53
movimiento, 50, 54
movimientos circulares, 87
movimientos coordinados sincronizados, 75
movimientos independientes, 64
movimientos semicoordinados, 67

N

NORMAL, 30
n puntos relativos, 42

O

objeto de trabajo, 63
objeto de trabajo coordinado, 63
opciones, 7, 13
orientación de la herramienta, 87

P

parámetros del sistema, 30
PERS, 89
posicionador, 15
ProcerrRecovery, 93, 95
Producción, ventana, 53
programa de tarea, 15
programar, 59
puntos de paro, 79

R

RAPID, 59
recuperación en caso de error, 91
RestoPath, 93, 97
robot, 15

S

seguridad, 11
seleccionar tareas, 55
SEMISTATIC, 30
sincronización, 15, 67, 81
sistema de coordenadas de la base, 45, 47
sistema de coordenadas del objeto, 45
sistema de coordenadas del usuario, 45, 47
sistema de coordenadas de objeto, 47
Sistema de coordenadas mundo, 45, 47
sistemas de coordenadas, 41, 45
Speed Control Percent, 33, 37
Speed Control Warning, 33, 35, 37
STATIC, 30
StorePath, 93, 97
SyncArc, 18, 37, 47, 68, 76
syncident, 59, 89
SyncMoveOff, 60, 75
SyncMoveOn, 60, 75
SyncMoveUndo, 60, 90
System Input, 34, 39
System Output, 34, 39

T

tareas de movimiento, 13
Task, 30, 35, 37
TASK PERS, 89

tasks, 55, 59, 62
tipos de datos, 59
Type, 30, 35, 37

U

UNDO, 90
UnsyncArc, 17, 35, 45, 65, 102
Use Mechanical Unit Group, 30, 35, 37
Use Motion Planner, 31, 35, 37

V

velocidad, 84
velocidad del robot, 84

W

WaitSyncTask, 60, 67

Z

zonas de esquina, 79
Zonas mundo, 88

Contact us

ABB AB

**Discrete Automation and Motion
Robotics**

S-721 68 VÄSTERÅS, Sweden

Telephone +46 (0) 21 344 400

ABB AS, Robotics

Discrete Automation and Motion

Nordlysvegen 7, N-4340 BRYNE, Norway

Box 265, N-4349 BRYNE, Norway

Telephone: +47 51489000

ABB Engineering (Shanghai) Ltd.

No. 4528 Kangxin Hingway

PuDong District

SHANGHAI 201319, China

Telephone: +86 21 6105 6666

www.abb.com/robotics