Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería Mecánica Eléctrica

DISEÑO DE CONTROL PARA TROQUELADORA DE LÁMINA DE MATRIZ POSICIONADA POR MEDIO DE SERVOMOTORES

Carlos Humberto López Valdez
Asesorado por la Inga. Ingrid Salomé Rodríguez García de Loukota

Guatemala, abril de 2011

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

DISEÑO DE CONTROL PARA TROQUELADORA DE LÁMINA DE MATRIZ POSICIONADA POR MEDIO DE SERVOMOTORES

TRABAJO DE GRADUACIÓN

PRESENTADO A JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

CARLOS HUMBERTO LÓPEZ VALDEZ
ASESORADO POR LA INGA. INGRID SALOMÉ RODRÍGUEZ GARCÍA DE LOUKOTA

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN ELECTRÓNICA

GUATEMALA, ABRIL DE 2011

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Murphy Olympo Paiz Recinos
VOCAL I	Ing. Alfredo Enrique Beber Aceituno
VOCAL II	Ing. Pedro Antonio Aguilar Polanco
VOCAL III	Ing. Miguel Angel Dávila Calderón
VOCAL IV	Br. Luis Pedro Ortiz de León
VOCAL V	P. A. José Alfredo Ortiz Herincx
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Murphy Olympo Paiz Recinos
EXAMINADOR	Ing. Carlos Eduardo Guzmán Salazar
EXAMINADOR	Ing. Julio César Solares Peñate
EXAMINADOR	Ing. Romeo Neftalí López Orozco
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

Guatemala 9 de septiembre del 2010

Ingeniero
Carlos Eduardo Guzmán Salazar
Coordinador del Área de Electrónica
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería, USAC.

Estimado Ingeniero Guzmán.

Me permito dar aprobación al trabajo de graduación titulado: **"DISEÑO DE CONTROL PARA TROQUELADORA DE LÁMINA DE MATRIZ POSICIONADA POR MEDIO DE SERVOMOTORES"**, del señor Carlos Humberto López Valdez, por considerar que cumple con los requisitos establecidos.

Por tanto, el autor de este trabajo de graduación y, yo, como su asesora, nos hacemos responsables por el contenido y conclusiones del mismo.

Sin otro particular, me es grato saludarle.

Atentamente,

Inga. Ingrid Rodríguez de Loukota
Colegiada 5,356
Asesora

Ingrid Rodríguez De Loukota
INGENIERA EN ELECTRONICA
COLEGIADA 5,356

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERÍA

Ref. EIME 01. 2010
Guatemala, 10 de enero 2011.

Señor Director
Ing. Guillermo Antonio Puente Romero
Escuela de Ingeniería Mecánica Elèctrica
Facultad de Ingeniería, USAC.

Señor Director:

Me permito dar aprobación al trabajo de Graduación titulado: "DISEÑO DE CONTROL PARA TROQUELADORA DE LÁMINA DE MATRIZ POSICIONADA POR MEDIO DE SERVOMOTORES", del estudiante, CARLOS HUMBERTO LÓPEZ VALDEZ, que cumple con los requisitos establecidos para tal fin.

Sin otro particular, aprovecho la oportunidad para saludarle.

Atentamente,
ID Y ENSEÑAD A TODOS

Ing. Carlos Eduardo Guzmán Salazar
Coordinador de Electrónica

UNIVERSIDAD DE SAN CARLOS
FACULTAD DE INGENIERÍA
DIRECCION ESCUELA DE INGENIERIA
MECANICA ELECTRICA
GUATEMALA

CEGS/sro

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERÍA

REF. EIME 03. 2011.

El Director de la Escuela de Ingeniería Mecánica Eléctrica, después de conocer el dictamen del Asesor, con el Visto Bueno del Coordinador de Área, al trabajo de Graduación del estudiante; CARLOS HUMBERTO LÓPEZ VALDEZ titulado: "DISEÑO DE CONTROL PARA TROQUELADORA DE LÁMINA DE MATRIZ POSICIONADA POR MEDIO DE SERVOMOTORES", procede a la autorización del mismo.

Ing. Guillermo Antonio Puente Romero

UNIVERSIDAD DE SAN CARLOS
FACULTAD DE INGENIERIA
DIRECCION ESCUELA DE INGENIERIA
MECANICA ELECTRICA
GUATEMALA

GUATEMALA, 13 DE ENERO 2,011.

Universidad de San Carlos
de Guatemala

Facultad de Ingeniería
Decanato

DTG. 119.2011

El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería Mecánica Eléctrica, al trabajo de graduación titulado: **DISEÑO DE CONTROL PARA TROQUELADORA DE LÁMINA DE MATRIZ POSICIONADA POR MEDIO DE SERVOMOTORES,** presentado por el estudiante universitario **Carlos Humberto López Valdez,** autoriza la impresión del mismo.

IMPRÍMASE:

Ing. Murphy Olympo Paiz Recinos
Decano

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANO
FACULTAD DE INGENIERIA

Guatemala, 27 de abril de 2011.

/gdech

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

DISEÑO DE CONTROL PARA TROQUELADORA DE LÁMINA DE MATRIZ POSICIONADA POR MEDIO DE SERVOMOTORES

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería Mecánica Eléctrica, con fecha 10 de septiembre de 2009.

Carlos Humberto López Valdez

AGRADECIMIENTOS A:

Dios	Por haberme iluminado en todo momento y haberme dado la fuerza suficiente para cumplir esta meta.
Mi Mamá	Por ser ese apoyo incondicional que siempre me motivó a seguir adelante.
Mi Papá	Por ser más que un padre, un amigo para mí y mis hermanos.
Mi familia	Por apoyarme en todo lo que hago.
Mi asesora	Por toda la ayuda y la paciencia brindada durante el desarrollo de este trabajo.
Mis amigos	Por las innumerables anécdotas que surgieron durante la realización de los proyectos.
Mi Abuelo (q.e.p.d.)	Por haberme enseñado que la mejor forma de alcanzar mis metas es el trabajo duro.
La USAC	Por brindarme la oportunidad de ser un profesional egresado de esta casa de estudios.

A MIS AMIGOS:

“En el camino de tu vida, no es tan importante la distancia que has recorrido, sino la dirección que llevas.”

Anónimo

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
g(x)	Denota una función matemática en general
d/dt	Derivación matemática
KHz	Dimensional de frecuencia, representa mil hertz
E/S	Entrada/salida
↓	Flanco descendente
0	Grados de apertura respecto a un punto
∫	Integración matemática
C++	Lenguaje de programación
Φ	Letra griega phi, representa un ángulo
θ	Letra griega teta, representa un ángulo
m/s	Metros/segundo, dimensional de velocidad

%	Porcentaje
π	Representa el valor 3.14159265…
RPM	Revoluciones por minuto
=>	Transición de un lugar a otro

GLOSARIO

AlNico — Aleación formada principalmente de cobalto, aluminio y níquel, aunque también puede contener aluminio, cobre y raras veces titanio. Su uso principal es en aplicaciones magnéticas.

Amplificadores — Dispositivo que por medio del uso de energía magnifica la amplitud de un fenómeno.

Ansi C — Es un estándar publicado por el Instituto Nacional Americano de Estándares (ANSI) para el lenguaje de programación C.

Archivos .m — Es la extensión de los archivos que contienen el código de programación para aplicaciones desarrolladas en *Matlab*.

Archivos .bmp — Extensión de los archivos utilizados por *Windows* y *MS-DOS* para guardar imágenes, este sistema de archivos puede guardar imágenes de 24 *bits*, 8 *bits* y menos.

Archivos .DGW — Extensión para archivos electrónicos de dibujo computarizado, utilizado principalmente para el programa *AutoCAD.*

Archivos .DXF	Del inglés *Drawing Exchange Format*, es un formato de archivo informático para dibujos CAD, creado para que todos los archivos .DGW, usados por *Autocad* sean compatibles con las demás aplicaciones de diseño CAD.
Archivos .JPEG	*Joint Photographic Experts Group,* extensión de archivos de imágenes de 24 *bits* que han sido comprimidas por medio de un algoritmo que lleva el mismo nombre JPEG.
Archivos PDF	Es un formato de almacenamiento de documentos creado por la empresa Adobe *Systems*, el significado de sus siglas es Formato de Documento Portable.
Asbesto	También conocido como amianto, material utilizado para la creación de fricciones para frenos y como material aislante.
AutoCAD	Programa de dibujo asistido por computadora para diseños en dos y tres dimensiones.
Automoción	Capacidad que tiene una máquina de moverse sin la intervención del ser humano, es decir por sí misma.
AVI	Formato contenedor de audio y video lanzado por *Microsoft* en 1992.

C++	Lenguaje de programación basado en texto.
Catia	Programa para diseño CAD/CAM/CAE, similar a *solidworks*.
Cavidades hembra – macho	Juego de perforaciones y salientes en distintas piezas que al estar en contacto cazan a la perfección y sujetan dos piezas en su lugar, generalmente la saliente se conoce como macho y la perforación como hembra.
Centro de masa	Punto dentro de un cuerpo donde se considera que se concentra toda su masa.
Cinemática	Parte de la mecánica encargada de estudiar el movimiento de los cuerpos.
Cizallamiento	Es la acción de cortar un material debido a una fuerza aplicada en una dirección específica.
Coeficiente de fricción dinámico	Es el coeficiente de fricción que existe entre dos superficies cuando una de ellas está en movimiento.
Coeficiente de fricción estático	Es el coeficiente de fricción que existe entre dos superficies que no están en movimiento.

Colector	Es un anillo de láminas de cobre situado sobre el eje del rotor, cuya función es unir la bobinas del rotor con el circuito exterior por medio de las escobillas.
Compact fieldpoint	Es un sistema de control industrial y medición, completamente expandible y modular de tal manera que el usuario decide que módulos adquirir y para qué función.
Corrientes parásitas	Corrientes generadas dentro del conductor cuando éste atraviesa un campo magnético variable, se llaman parásitas porque dicha corriente genera electroimanes con efectos opuestos al campo magnético aplicado.
Croquis	Es el diseño preliminar de una idea.
Curva de saturación	Representa la relación entre la densidad de flujo magnético (B) y la intensidad de campo magnético (H) en un material ferromagnético. Se utilizan para diseñar y analizar el comportamiento de motores y transformadores.

DAQ	Significa adquisición de datos (*Data acquisition*) y hace referencia a la toma de muestras del mundo real para generar datos que puedan ser manipulados por una computadora.
Demoduladores	Conjunto de dispositivos electrónicos, configurados de tal manera que son capaces de extraer información de una señal portadora.
Densidad	Es la cantidad de masa contenida en un volumen definido, también se le conoce como peso específico.
Derivada	Operación matemática que muestra cómo cambia una función en la medida en que su entrada cambia.
Devanado de campo	Arrollamiento de alambre de cobre cubierto con un esmalte, también se le conoce como electroimán. Es el encargado de generar el campo magnético para que el motor funcione, llamado algunas veces devanado de excitación.

Drag and drop	Herramienta propia de *Windows* que permite arrastrar un elemento, manteniendo presionado el botón izquierdo del ratón, de un punto y depositarlo en otro liberando el botón izquierdo del ratón.
DSP	Procesador digital de señales, es un dispositivo que permite capturar datos analógicos en tiempo real y es capaz de realizar operaciones numéricas a muy alta velocidad.
ECAD	Herramienta utilizada para diseñar placas electrónicas y circuitos integrados en computadora , sus siglas significan *Electronic Computer Aided Design.*
Elastómeros	Clase de polímero con la particularidad de ser muy elástico, conocido comúnmente como caucho.
Embebido	Para este caso hace referencia a sistemas de computación que realizan diversas tareas, la mayoría de ellas en tiempo real.
Embrague	Mecanismo encargado de permitir o restringir la transmisión de energía mecánica de un punto a otro.

Entidades geométricas vectoriales	Es el nombre con que se conocen a los puntos, las líneas, los arcos y los polígonos dentro de los sistemas CAD.
Estado estacionario	Se le conoce así a cualquier sistema cuyas características no varían con el tiempo.
Estado transitorio	Hace referencia a los sistemas cuyas características varían durante cortos períodos de tiempo.
Excitación	Se refiere al inicio de generación de campo magnético por parte de los devanados de excitación.
Excitatriz	Es una bobina sobre un núcleo de hierro alimentado por corriente continua (c.c.), generalmente se utiliza en generadores eléctricos.
Filtros adaptivos	Tipo de filtro que puede cambiar su forma de comportarse de acuerdo a un algoritmo. La idea de este filtro es intentar modelar la relación de señales en tiempo real de forma repetitiva.

Fluencia	Fenómeno presente en algunos materiales, tiene lugar cuando el material pasa de sufrir una transformación elástica a una plástica, es decir es cuando el material cambia sus dimensiones y/o forma debido a un esfuerzo aplicado.
FPGA	*Field Programmable Gate Array*, es un dispositivo semiconductor que contiene bloques de lógica cuya interconexión y funciones se pueden programar.
Fuerza magneto-motriz	Está representada por el símbolo F y es aquella capaz de producir un flujo magnético entre dos puntos de un circuito magnético.
Funciones API	API significa plataforma de interfaz para el desarrollo de aplicaciones, y éstas son funciones que se encuentran compiladas y almacenadas en archivos .dll o .exe. Son utilizadas para realizar acciones sobre el sistema operativo.

GPIB	*General Purpouse Interface Bus*, es un estándar de bus de datos digitales utilizado para conectar dispositivos de medición como multímetros, osciloscopios, etc. con dispositivos que los puedan controlar como una computadora.
Grado de libertad	Un grado de libertad es la capacidad que tiene un mecanismo para realizar movimientos dentro de un eje específico. No necesariamente existe un grado de libertad por cada eje.
Gráfica de *Bode*	Es una representación gráfica logarítmica que sirve para representar la respuesta en frecuencia de un sistema.
Histéresis	Fenómeno por el cual un material ferromagnético se mantiene magnetizado luego de retirar el campo que lo magnetizó.
HMI	*Human Machine Interface*, se utiliza para referirse a la interacción entre humanos y máquinas por medio de dispositivos visuales.
Horquilla	Pieza mecánica en forma de Y invertida que sirve para sujetar otros elementos mecánicos que regularmente rotan con respecto a un eje central.

Inducido	Parte rotativa del motor en la cual tiene lugar la transformación de energía eléctrica en mecánica por medio de la inducción electromagnética, comúnmente se le llama así al rotor.
Inercia	Es la resistencia que opone un cuerpo al efecto de una fuerza que actúa sobre él.
Infografía de *marketing*	Estrategia utilizada por los mercadólogos para presentar de manera visual y menos aburrida los datos más relevantes acerca de un estudio, generalmente referentes a nuevos productos.
Inspección	Proceso industrial que cuenta con la capacidad de supervisar diversos procesos por medio de cámaras y equipos especiales, de tal manera que pueden notar cambios y errores durante el mismo.
Inventor	Paquete de modelación en 3 dimensiones que compite con otros programas de diseño CAD como *solidworks*, *Catia* y *solid edge*.
Java	Lenguaje de programación orientado a objetos desarrollado por *Sun Microsystems*.

Jerárquicos	Se refiere a que poseen una posición o espacio determinado de acuerdo a su nivel de importancia.
Layout	Hace referencia a distintas capas, en aplicaciones de diseño, las cuales poseen distintas características. Estas capas pueden ser duplicadas o bien crear nuevas para asignarles diversas características respecto de la original.
Leva	Elemento mecánico sujeto a un eje con un contorno de forma especial, de tal manera que el giro del eje hace que dicho contorno, con forma especial, toque, mueva o empuje una segunda pieza.
Líquido newtoniano	Fluido cuya viscosidad puede considerarse constante con respecto del tiempo.
Líquido no newtoniano	Fluido cuya viscosidad varía con las variaciones de temperatura y presión.
Microsoft pocket PC	Es una computadora de bolsillo también conocida como PDA, que ejecuta el sistema operativo *Windows* CE, el cual le proporciona capacidades similares a las computadoras de escritorio.

Modulación	Proceso por el cual un parámetro de la onda portadora cambia de valor con las variaciones de la señal moduladora.
Modulares	Hace referencia a bloques capaces de realizar diversas funciones y que pueden ser utilizados cuando sean requeridos.
Muelle	Conocido también como resorte, es un dispositivo capaz de almacenar y liberar energía sin sufrir deformación alguna.
Palm OS	Sistema operativo hecho por *Palmsource Inc*. para computadoras de mano (PDA's).
Par de frenado	Es el torque que realiza la fuerza de fricción, generalmente sobre una superficie circular.
PDA	Sus siglas significan Asistente Digital Personal, y es un sinónimo de una computadora de bolsillo.
Piezoactuadores	Dispositivo que produce movimiento, generalmente desplazamiento aprovechando el fenómeno físico de la piezoelectricidad.

Piezoelectricidad	Fenómeno presente en algunos cristales, los cuales al ser sometidos a tensiones mecánicas generan una pequeña cantidad de electricidad.
PLC	Controlador Lógico Programable, es un dispositivo electrónico utilizado en la automatización industrial cuyo fin es la obtención de datos, para luego ser trasladados y analizados por una computadora.
PLCopen	Organización compuesta por fabricantes de plc y desarrolladoras de *software*, cuyo principal objetivo es promocionar la aplicación del estándar de automatización en la industria.
Pro/engeneer	Aplicación CAD/CAM/CAE de diseño paramétrico muy popular en diseñadores mecánicos.
Programas de alto nivel	Son aplicaciones de programación caracterizadas por presentar los algoritmos a un nivel entendible por la capacidad cognitiva humana, en lugar de la capacidad ejecutora de las máquinas.

Radio frecuencia	Término aplicado a las frecuencias del espectro electromagnético cuyo rango en Hertz se encuentra entre 3 y 300 mil millones.
Renderizado	Término propio del lenguaje informático utilizado para referirse a la generación de una imagen desde un modelo.
Retroalimentación	Proceso mediante el cual una porción de la señal de salida de un sistema se redirige a la entrada.
Retroalimentación negativa	Tipo de retroalimentación en la cual la porción redirigida a la entrada se comporta de manera opuesta al comportamiento de la entrada, generalmente se utiliza para estabilizar señales de salida.
RIO	Significa *Reconfigurable I/O* (entradas/salidas), es un controlador industrial que combina un controlador en tiempo real, módulos de entradas y salidas reconfigurables, un módulo de FPGA y un chasis para *Ethernet.*

Rodamiento	Elemento mecánico que reduce la fricción entre un eje y las piezas conectadas a éste, sirve de apoyo y facilita el desplazamiento de las mismas. Generalmente se le conoce con el nombre de cojinete.
Señales IF	Las señales de frecuencia intermedia, son generalmente las señales utilizadas para recuperar o demodular radio AM, FM y televisión; los valores de dichas frecuencias varían de acuerdo a la aplicación y su utilidad radica en hacer más fácil la sintonización de un canal de TV o una estación de radio.
Servoactuadores	Mecanismos capaces de realizar movimiento, dicho movimiento es impulsado o generado por un servomotor.
Simulink	Entorno de programación visual que funciona bajo la aplicación de programación basado en texto *matlab*.
Solid edge	Programa para modelado de piezas en 3D basado en un *software* de diseño asistido por computadora, una aplicación de características similares a *Solidworks.*

SQL	Es un lenguaje de programación formal y estandarizado para la manipulación de información en bases de datos, sus siglas significan *Structured Query Languaje*.
Transductor	Dispositivo capaz de convertir fenómenos mecánicos en señales eléctricas.
Útiles	Se refiere a las piezas, en las máquinas herramienta, encargadas de realizar la función de cortado, desgastado o perforado del material.
Visual basic	Lenguaje de programación que combina la programación basada en texto y la programación gráfica, utilizado principalmente para el desarrollo de interfaces gráficas. Su principal función es combinar ambas herramientas (programación gráfica y de texto) para la simplificación en la programación de aplicaciones.

RESUMEN

Durante los últimos años hemos sido testigos de cómo la automatización, por medio del valioso recurso de la investigación ha evolucionado, capturando una gran cantidad de procedimientos industriales para mejorarlos y convertirlos en procesos mucho más productivos.

Buscando aportar un granito de arena a dicho recurso, es como ha surgido la creación de este trabajo de graduación, en el cual se hace una propuesta para el sistema de control de una troqueladora gobernada por medio de servomotores, simulada en comunión por los programas *Solidworks* y *LabVIEW*.

Al internarse en el trabajo de graduación encontrará información sobre características relevantes de servomotores, engranajes, servomecanismos y servo controladores. Luego se presenta una breve descripción de las máquinas herramienta comunes e información detallada sobre la troqueladora y algunas de sus aplicaciones. Posteriormente hay una descripción general de los programas *Solidworks* y *LabVIEW*, a manera de formar una idea global sobre el potencial de ambos programas.

Enseguida se presentan las herramientas específicas, de cada programa, utilizadas para lograr una simulación exitosa del sistema de troquelación. Aprenderá cómo configurar un mecanismo, cómo agregar motores y relaciones mecánicas al mismo, todo esto realizado en *Solidworks*.

Por el otro lado, de *LabVIEW* conocerá el modulo NI *softmotion* y cómo utilizar las funciones de movimiento, aprenderá a agregar sensores virtuales, a configurar un proyecto y a comunicar *LabVIEW* con *Solidworks* para simular la troqueladora.

Finalmente, encontrará el desarrollo de la aplicación de control, la explicación de su funcionamiento y la presentación final de la misma, seguida de varias imágenes de la aplicación funcionando.

OBJETIVOS

General

Diseñar una aplicación de control, en *LabVIEW*, para gobernar los movimientos de un mecanismo de troquelación simulado en *Solidworks*.

Específicos

1. Explicar que es un servomotor, los tipos que existen y las partes que los componen.

2. Hablar acerca de los componentes y el funcionamiento de un servomecanismo.

3. Presentar las estrategias más comunes sobre el control de procesos, que forman parte de la teoría de servocontrol.

4. Mostrar en forma general las máquinas herramienta más comunes de la industria.

5. Explicar sobre los elementos de una troqueladora y las funciones que realizará el prototipo creado.

6. Explicación de las características generales de cada *software*.

7. Explicación de las características específicas que permitirán comunicar ambos programas para desarrollar el prototipo virtual de la troqueladora.

8. Lograr fusionar de manera adecuada la ingeniería propia con las herramientas de *software* para el diseño final de la aplicación de control para el prototipo virtual de la troqueladora.

INTRODUCCIÓN

En estos días es normal observar como las computadoras tienen un rol vital dentro de nuestra vida, tanto profesional como personal. Existen aplicaciones para todo tipo de tareas, desde algo tan sencillo como una calculadora hasta aplicaciones que cuestan miles de dólares y que son capaces de realizar tareas complicadas en tiempos cortos.

Aplicaciones costosas y efectivas son excelentes pero tienen una parte oscura, por llamarle de algún modo, y es que requieren entrenamiento especializado para poder utilizarlas. Es precisamente ayudar con esta tarea como surge la creación de este documento.

La idea, es preparar al lector a lo largo del mismo, de manera que adquiera los conocimientos básicos sobre servomecanismos y sus componentes, sobre las partes básicas de un sistema de troquelación y sobre los programas de computadora capaces de realizar dicha tarea, buscando habilitarlo para poder utilizar el *software* de diseño en 3D *Solidworks*, en conjunto con el *software* para automatización y análisis de datos *LabVIEW*, de tal manera que, luego de haber leído el documento, cuente con la capacidad de simular el funcionamiento de un prototipo virtual de una troqueladora, creado en *Solidworks* y controlado desde *LabVIEW*.

1. PRINCIPIOS BÁSICOS

En este apartado de principios básicos se intentarán exponer de una forma aplicativa los conceptos más comunes de servomotores, servosistemas y servocontrol. Se pretende ayudar al lector a observar las cosas desde una perspectiva completamente distinta y así ayudar a disipar un poco las dudas que rodean a todo lo relacionado con los servomotores.

La intensión es tratar de introducir al lector en los servomotores sin tener que abrumarlo con excesiva teoría y matemática, sino hacerle ver lo interesante de los servos de una forma entretenida y accesible para cualquier tipo de persona con conocimientos básicos sobre electrónica y motores.

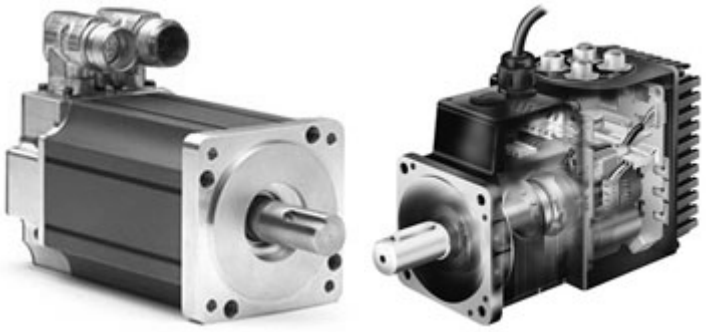
1.1. Servomotores

La palabra “servo” es derivada del latín esclavo y en efecto esta relación indica que un servomotor es un esclavo que responde a las instrucciones de equipos externos para el motor.

Un servomotor (de ahora en adelante servo) es un dispositivo que tiene un eje de rendimiento controlado. Éste puede ser llevado a posiciones angulares específicas al enviar una señal codificada. Existiendo una señal codificada en la línea de entrada, el servo mantendrá la posición angular del engranaje; cuando la señal codificada cambia, la posición angular de los piñones cambia.

Los servomotores son utilizados en las más variadas aplicaciones industriales donde una elevada dinámica, control de par, precisión de velocidad y posicionamiento son factores decisivos para el aumento de la calidad y productividad. Físicamente lucen como muestra la figura 1 y poseen todas estas características aliadas a un relativo bajo costo, elevado desempeño y robustez.

Figura 1. **Servomotores**



Fuente: http://www.electronicaindustrial.cl/images/servo2.jpg

1.1.1. Servomotores de corriente continua (c.c.)

Tal y como lo cita Irwing L. Kosow en su libro máquinas eléctricas y transformadores, los servomotores de corriente continua (c.c.) son motores de c.c. alimentados por corriente suministrada por amplificadores electrónicos de c.c. o por amplificadores de corriente alterna (c.a.) con demoduladores colocados interna o externamente, por rectificadores controlados de silicio o por alguno de los tipos de amplificadores rotativos. Los servomotores de c.c. tienen potencias comprendidas entre 0.05 y 1000 hp.

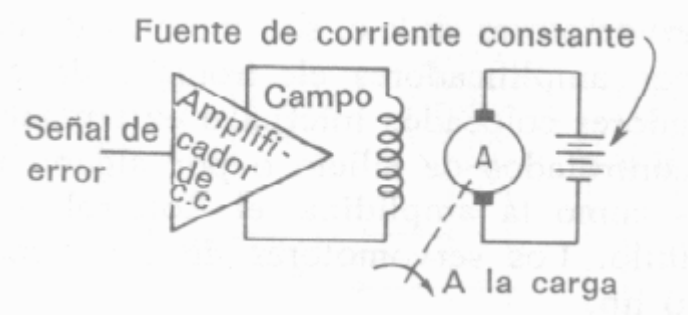
Las características fundamentales a cumplir por cualquier servomotor son dos, la primera de ellas es que el par de salida del motor es proporcional a la tensión de control que se aplica y la segunda es que el sentido del par viene determinado por la polaridad instantánea de la tensión de control.

Se utilizan cuatro tipos de servomotores de c.c. que serán tratados a continuación: el motor de derivación con excitación controlada, el motor de derivación con inducido controlado, el motor serie y el motor de derivación de imán permanente (excitación de campo fija).

1.1.1.1. Servomotor c.c. de excitación controlada

El esquema de este motor es como el mostrado en la figura 2, nótese que el par desarrollado por este motor es nulo cuando no existe excitación en el devanado de campo alimentado por el amplificador de error de c.c. y como la corriente en el inducido es siempre constante, el par varía directamente con el flujo de excitación y también con la corriente de excitación hasta que se alcanza la saturación.

Figura 2. **Servomotor de excitación controlada**



Fuente: Irving L. Kosow. Máquinas eléctricas y transformadores. p. 497

Si la polaridad de la excitación se invierte, el motor cambia sentido de giro. El control de la corriente de excitación por este método se usa sólo en pequeños servomotores debido a:

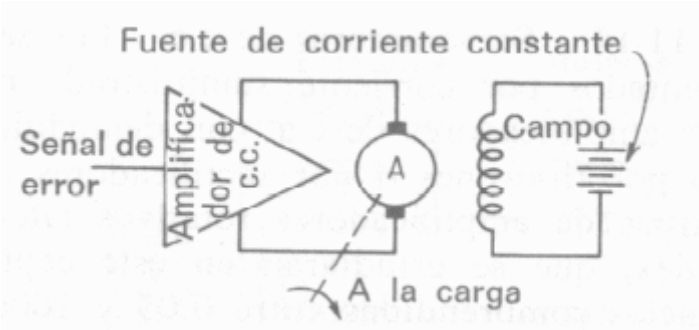
- no es deseable tener que alimentar una corriente grande y de valor fijo como la requerida por los grandes servomotores de c.c.

- la respuesta dinámica es más lenta que la de los motores de inducido controlado debido a la gran constante de tiempo del altamente inductivo circuito de excitación.

1.1.1.2. Servomotor c.c. de inducido controlado

Este servomotor trabaja con excitación fija alimentada por una fuente de corriente constante como la indicada en la figura 3, este tipo de control presenta ciertas ventajas sobre el de excitación controlada en lo que se refiere a la respuesta dinámica. Una variación brusca, grande o pequeña, de la tensión de inducido producida por la señal de error dará lugar a una respuesta casi inmediata del par, ya que el circuito del inducido es esencialmente resistivo si se le compara con el de excitación que es altamente inductivo.

Figura 3. **Servomotor de inducido controlado**



Fuente: Irving L. Kosow. Máquinas eléctricas y transformadores. p. 498

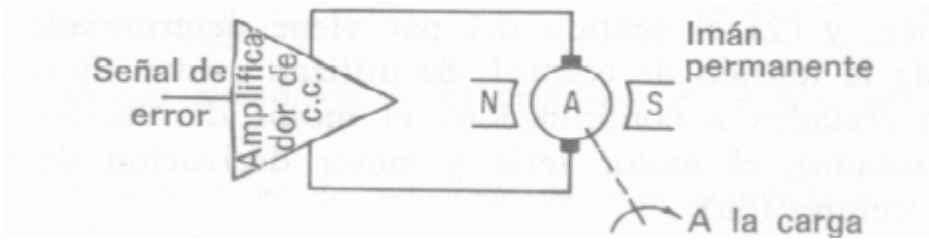
La excitación de este motor tiene un valor bastante inferior al correspondiente del codo de la curva de saturación con el fin de hacer que el par sea menos sensible a los pequeños cambios de la tensión de la fuente de corriente constante. Además, un flujo de excitación de valor elevado aumenta la sensibilidad de par del motor para el mismo valor de un pequeño cambio en la corriente de inducido. Funcionan de esta forma, controlados por la tensión de inducido, motores de inducido de hasta 1000 hp. Si se invierte la señal de error y la polaridad de la tensión del inducido, el motor cambia su sentido de giro.

Los grandes motores de derivación de c.c. controlados por el inducido se controlan generalmente mediante excitatrices de campos múltiples denominados amplificadores rotativos cuando los requerimientos del servomecanismo imponen la necesidad de dispositivos de control de alta potencia.

1.1.1.3. Servomotores c.c. de imán permanente

Se utilizan para instrumentos pequeños motores de c.c. de potencia fraccionaria que utilizan imanes permanentes como excitación constante, en lugar de una fuente de corriente constante de excitación, como se indica en la figura 4. Estos dispositivos se fabrican generalmente para tensiones nominales entre 6 y 28 V. La estructura de la excitación de estos motores consiste en una aleación Alnico VI moldeada en forma de anillo circular de aproximadamente 1'' de diámetro que rodea totalmente al inducido y que proporciona un flujo intenso.

Figura 4. **Servomotor de imán permanente**



Fuente: Irving L. Kosow. Máquinas eléctricas y transformadores. p. 500

Los motores de imán permanente se compensan por medio de devanados de conmutación para evitar la desmagnetización de los imanes de excitación cuando la tensión del inducido se invierte bruscamente.

Las corrientes parásitas y los efectos de histéresis son generalmente despreciables en este tipo de motores y las piezas polares son generalmente laminadas para reducir las chispas en las escobillas cuando tiene lugar un rápido cambio en la tensión de la señal. Estos dispositivos se controlan también por control de la tensión del inducido de la misma manera que los motores de derivación de inducido controlado.

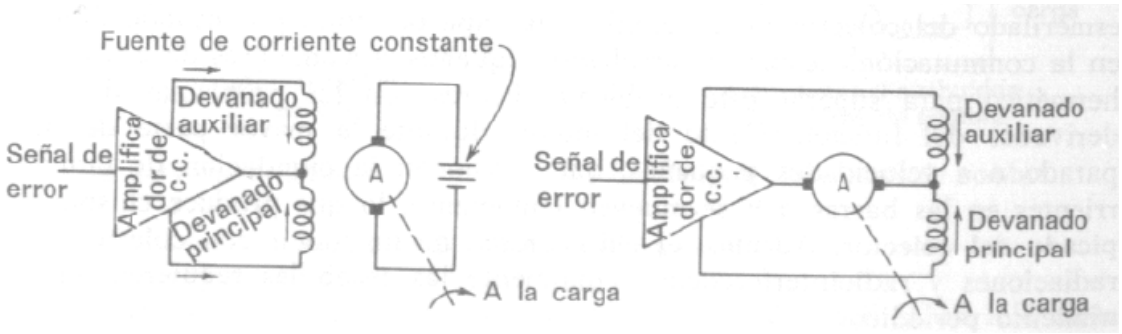
1.1.1.4. Servomotores c.c. con excitación serie partida

Los motores de c.c. con excitación serie partida de potencia fraccional pueden hacerse funcionar con excitación independiente controlada en la forma indicada en la figura 5a. Uno de los devanados se denomina principal y el otro auxiliar, si bien suelen producir iguales fuerzas magneto-motriz (fmm) y estar colocados en los polos de manera que tiendan a producir sentidos contrarios de rotación. Como indica la figura 5a, los motores pueden tener excitación independiente, estando alimentado el inducido por una fuente de corriente constante.

Las ventajas del control de la excitación por el método de la excitación partida son: primero la respuesta dinámica del inducido mejora al existir siempre excitación (no hay retraso debido a la constante de tiempo inductiva), y segundo se obtiene un control más fino al ser el sentido de rotación más sensible a las extremadamente pequeñas diferencias de corriente entre los devanados principal y auxiliar.

Los motores serie mayores funcionan según la configuración de la figura 5b debido a que es de difícil consecución la excitación independiente del inducido utilizando corrientes constantes. En esta disposición la corriente del inducido del motor de excitación partida es la suma de las corrientes en los devanados principal y auxiliar. Cuando las corrientes de estos devanados serie son iguales y opuestas no se produce par. Una pequeña disminución o aumento de la corriente del devanado auxiliar producirá un par instantáneo y un giro en el sentido correspondiente.

Figura 5. **Servomotor c.c. de excitación serie partida**



(a) Excitación independiente (b) Excitación directa

Fuente: Irving L. Kosow. Máquinas eléctricas y transformadores. p. 503

El servomotor serie proporciona un gran par de arranque y una rápida respuesta a señales de error. La regulación de velocidad de este tipo de motores es deficiente, pero esto no suele constituir un inconveniente grave en un servosistema ya que generalmente la carga es constante. La utilización de dos devanados en oposición reduce el rendimiento del motor, lo que no presenta problemas en el caso de los motores más pequeños.

En general, los motores de c.c. con excitación paralelo (*shunt)* y serie tienen rotores de mayor inercia que los de c.a. de igual potencia debido al mayor peso de los devanados de inducido de c.c. Si a ello añadimos la mayor resistencia mecánica derivada del rozamiento de las escobillas, se comprende la no utilización de motores de c.c. en servosistemas de instrumentos sensibles extremadamente pequeños.

La conmutación es también otro problema de los servomotores de c.c. aunque los polos auxiliares y los devanados de compensación lo reducen notablemente. Sin embargo, a grandes altitudes debido a la falta de oxigeno, el esmerilado del colector puede eliminar la capa de óxido provocando dificultades en la excitación.

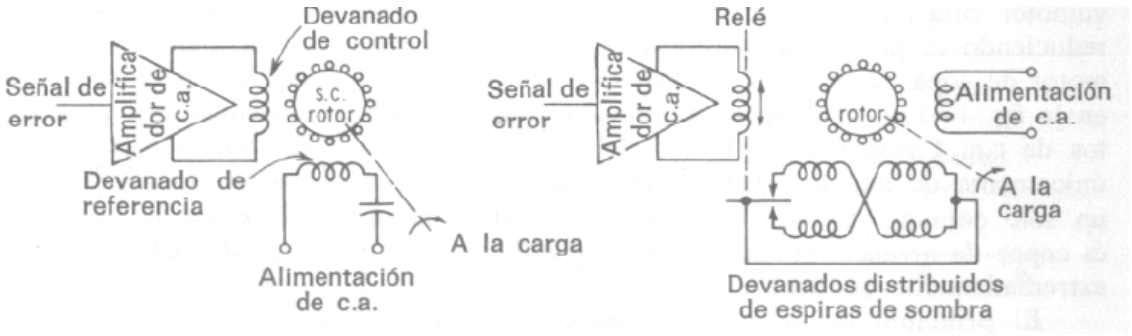
Se han desarrollado pequeños servomotores de c.c. con cierre hermético para superar este problema. Peores son los problemas del colector derivados del funcionamiento del motor, durante la mayor parte del tiempo, parado o a velocidades pequeñas que ocasionan la circulación de grandes corrientes en las barras que se mueven lentamente, lo que produce chisporroteo y picado del colector. Además el chisporroteo del motor con el colector produce radiaciones y radiointerferencias y finalmente las escobillas requieren mantenimiento periódico. Por todas las razones anteriores la mayoría de los servomotores pequeños utilizados en servomecanismos son de c.a. del tipo de inducción con espiras de sombra o bifásicos.

1.1.2. Servomotores de corriente alterna (c.a.)

La potencia mecánica de los servomotores de c.a. de espiras de sombra está comprendida entre 1/5000 y 1/8 hp. Para mayores potencias se utilizan siempre motores de c.c. a pesar de los inconvenientes antes mencionados, debido al poco rendimiento de los motores de c.a. de mayores potencias; si se construyeran con las características de par-velocidad que requieren los servomecanismos, se producirían dificultades de refrigeración en los motores de c.a.

En la figura 6a se muestra el dibujo esquemático de un servomotor bifásico y en la figura 6b el correspondiente al tipo de espiaras de sombra. El más empleado es el servomotor bifásico de cuatro terminales de la figura 6a. Se trata de un verdadero motor bifásico que tiene dos devanados estatóricos desplazados 90^0 dentro del estator. El devanado de referencia se alimenta de la fuente constante de c.a. a través de un condensador.

Figura 6. **Servomotores de c.a.**



(a) Servomotor bifásico (b) Servomotor con espiras de sombra

Fuente: Irving L. Kosow. Máquinas eléctricas y transformadores. p. 505

En ausencia de señal de error el rotor jaula de ardilla está en reposo. Una pequeña señal de error de polaridad definida en relación al devanado de referencia se amplifica en el amplificador de c.a. y se introduce en el devanado de control. El giro del motor se efectúa en el sentido adecuado para reducir la señal de error y el motor se para cuando se alcanza el equilibrio (señal de error nula).

El servomotor con espiras de sombra de la figura 6b utiliza un relé direccional para que sus contactos cierren en cortocircuito las espiras de sombra correspondientes al giro en el sentido deseado.

A la fuente de alimentación de c.a. se conecta un devanado de excitación monofásico. En presencia de una señal de error suficiente para accionar el relé se cortocircuitan un par de devanados de espiras de sombra; el servomotor gira hasta que se alcanza el equilibrio (el relé queda entonces sin excitación y abre contactos) y el motor se para. Una señal de error de polaridad opuesta haría que el relé direccional actuará cerrando en cortocircuito un par distinto de devanados que provocarían el giro del servomotor en sentido contrario.

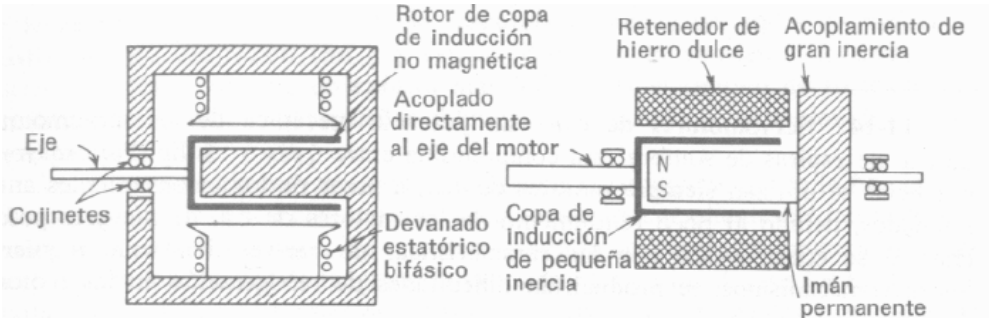
Es evidente que el motor bifásico de la figura 6a es el mejor de los dos tipos, ya que es capaz de responder a señales de error pequeñas. Un servomotor con espiras de sombra sólo responderá cuando la señal de error amplificada sea de valor suficiente para causar el funcionamiento del relé. La respuesta del servomotor bifásico a señales de control muy pequeñas puede mejorarse reduciendo el peso y la inercia del motor como en el tipo denominado “servomotor de copa de inducción”.

Estos servomotores de c.a. de bajo par mostrados en la figura 7a se adaptan extremadamente bien a servosistemas de instrumentos de c.a. Como todo el hierro del circuito magnético es fijo, el rotor consta únicamente de una fina vaina cilíndrica de cobre o latón y su eje se apoya en un solo cojinete. Debido a su pequeña inercia el motor de copa de inducción es capaz de arrancar incluso cuando se aplican a su devanado de control señales extremadamente pequeñas.

El principio de la copa de inducción se utiliza también para amortiguar servomotores de c.c. y c.a. que disminuyen lentamente su velocidad con el fin de que queden parados instantáneamente cuando la señal de error sea nula. Con ello se reduce la oscilación o el sobrealcance si se presenta una señal de error.

Como se indica en la figura 7b se acopla al motor una copa de inducción de pequeño peso e inercia. La copa de inducción rodea un imán permanente y a su vez está rodeada por retenedores de hierro dulce que conservan la retentividad del imán permanente. Cualquier cambio de velocidad por arranque, paro o inversión de sentido de giro producirá un efecto de amortiguamiento, las ventajas de este amortiguamiento son su larga duración y resistencia al desgaste.

Figura 7. **Servomotor de copa de inducido y amortiguador magnético**



(a) Servomotor de copa de inducción (b) Amortiguador magnético de copa de inducción

Fuente: Irving L. Kosow. Máquinas eléctricas y transformadores. p. 505

1.2. Servomecanismos

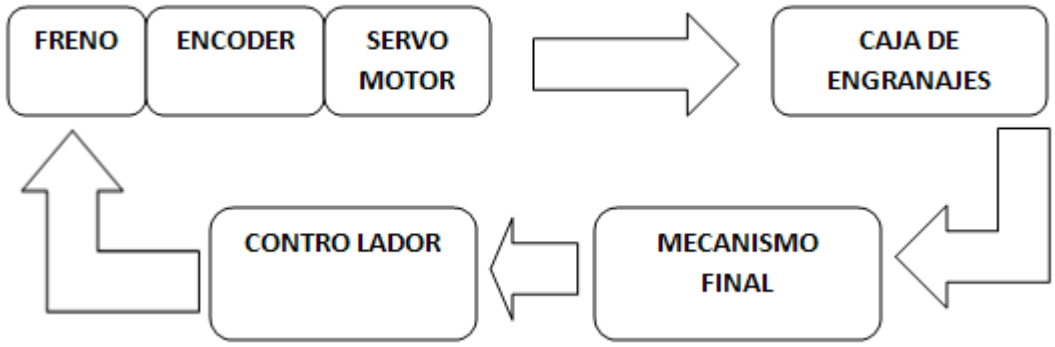
Los servomecanismos son sistemas de control por retroalimentación en los cuales la variable controlada es una posición mecánica. Esta es la definición recomendada por el *Feedback Control Commitee of the American Institute of Electrical Engineers.* Posteriormente el concepto se ha generalizado por lo que hoy se identifican por igual servomecanismos y servosistemas.

Así un servomecanismo o servosistema consiste en un sistema encargado de transmitir una orden o información de un punto a otro, comparando los resultados obtenidos en el extremo receptor con los que se deseaba obtener y deduciendo de esta comparación una señal correctora que introducida automáticamente en el sistema modifique los resultados para obtener el valor deseado, entonces la variable controlada ya no tiene por qué ser una posición mecánica, pudiendo tratarse de cualquier otra magnitud física.

Un servomecanismo es único entre los sistemas de control ya que controla un parámetro en función de la derivada con respecto del tiempo de ese parámetro. Por ejemplo un servomecanismo que controla una posición debe tener la capacidad de cambiar la velocidad del sistema debido a que la derivada de la posición con respecto del tiempo (tasa de cambio de la posición en el tiempo) es velocidad.

Un típico servosistema provee control de posición. Estos sistemas son comúnmente electrónicos o parcialmente electrónicos, usando un motor como su medio primordial para crear una fuerza mecánica. Usualmente utilizan el principio de la retroalimentación negativa, donde la entrada de control es comparada con la posición actual del sistema mecánico y medida por un transductor en la salida. Cualquier diferencia entre el valor actual y el valor deseado (una señal de error) es amplificada y utilizada para llevar al sistema en la dirección necesaria para reducir o eliminar el error, la figura 8 nos muestra las partes de un servosistema.

Figura 8. **Servomecanismo o servosistema**



1.2.1. Freno

El freno electromagnético ha encontrado un vasto campo de aplicaciones para el área de procesos sobre máquinas de las más diversas características operativas. Por su simplicidad constructiva, facilidad de aplicación, seguridad y precisión cuenta con la preferencia de los constructores y técnicos del sector mecánico industrial.

Por sus características técnicas y sus mínimas dimensiones de montaje el freno electromagnético se emplea preferentemente como elementos de comando para automatización de máquinas y aparatos en general. El freno se utiliza, por ejemplo, para realizar cambios de velocidad, comandos de accionamiento, efectuar posicionamiento o detención de la máquina en caso de daño y en general para cualquier proceso cíclico.

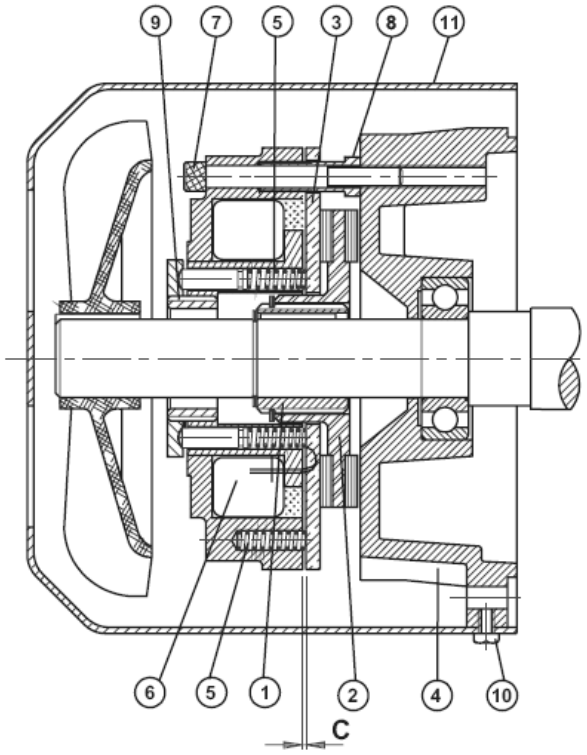
A continuación se hablará un poco sobre el mecanismo de frenado para motores más común que es, valga la redundancia, el freno electromagnético. También se tratará de explicar de una forma simple su funcionamiento y algunas características importantes que deben ser tomadas en cuenta en la selección del mismo.

1.2.1.1. Freno electromagnético de disco

Este equipo, tal como lo describe industrias *ramfe* en su documento frenos electromagnéticos, se compone de dos superficies de fricción, una bobina y unos resortes de carga. Al energizar la bobina se crea un campo magnético que evita que el freno esté accionado, permitiendo que el rotor del motor gire. Cuando la corriente de alimentación de las bobinas cesa, los resortes de carga presionan el disco de fricción contra la tapa del motor causando la detención del movimiento.

Los frenos electromagnéticos de disco son utilizados en aplicaciones donde es necesario obtener paradas rápidas del motor o cuando es preciso bloquear el motor en caso de un fallo o un corte de energía. Hay que destacar que estos frenos requieren escaso mantenimiento y su consumo de energía es mínimo. Para la alimentación de la bobina del freno se utiliza corriente continua, con voltajes entre 90 y 190 Volts. Los frenos electromagnéticos actúan en el momento de suspender la alimentación eléctrica al motor y la figura 9 muestra las partes que lo componen.

Figura 9. **Partes del freno electromagnético**



ITEM	CANT.	DENOMINACION
1	1	Tambor estriado
2	1	Disco de fricción
3	1	Disco de armadura
4	1	Tapa del motor
5	7	Resortes de carga
6	1	Bobina de excitación
7	3	Tornillos de fijación
8	3	Tornillos de calibración
9	1	Tuerca de ajuste de carga
10	3	Tornillos de fijación caperuza
11	1	Caperuza del motor

Fuente: Ramfe. Frenos electromagnéticos. p. E1

Los resortes de carga (5) ejercen una fuerza axial sobre el disco de armadura (3), presionándolo contra el disco de fricción (2) y éste contra la tapa del motor (4); produciendo un par de frenado sobre las dos superficies de fricción. El par de frenado aplicado al disco de fricción (2), es transmitido al eje del motor a través del tambor estriado (1) y su valor se puede reducir hasta en un 40% usando la tuerca de ajuste de carga (9). Al conectar la bobina (6) de freno, el campo magnético producido por la corriente de excitación, genera una fuerza axial resultante que atrae al disco de armadura (3) hacia la bobina, venciendo la fuerza aplicada por los resortes de carga (5); liberando así el eje del motor.

1.2.1.2. Consideraciones sobre los frenos

Procedamos a hablar sobre algunos parámetros relevantes a la hora de elegir el tipo de freno; el primero de ellos será el material de fricción. Éste se recomienda que se encuentre libre de asbestos, ya que de esta manera el coeficiente de fricción brindará una mejor respuesta aún a temperaturas elevadas. También se debe buscar un tipo de material que tenga un bajo coeficiente de rozamiento ya que otorga una mayor vida útil a todas las partes que están bajo constante roce y resulta ideal para aplicaciones donde los equipos están instalados en lugares de difícil acceso, también reducen el ruido a la hora de embragar convirtiéndose en la elección ideal para aplicaciones que requieren de bajo nivel de ruido.

La velocidad de rotación en un freno es una consideración importante en el momento de la selección. Se deben tener en cuenta varios factores como las RPM que permite el elemento según tablas, el torque requerido, la disipación térmica requerida, el efecto de la velocidad en la vida útil, y la respuesta del elemento a las bajas RPM. Cada una de estas consideraciones a veces afecta en forma individual o conjunta, pero siempre son importantes al momento de elegir el producto correcto para su aplicación.

Las RPM máximas son la consideración más importante respecto a la velocidad de rotación. Nunca se debe exceder la velocidad máxima para así evitar causar daños a personas o máquinas. Esta velocidad máxima está calculada a partir de la integridad estructural de los componentes rotativos, si se excede este límite se corre el riesgo de producir daños estructurales, roturas de los rodamientos o un desgaste prematuro del material de fricción.

Torque dinámico. Usualmente el torque dinámico a la mayor velocidad de patinamiento es el factor determinante al dimensionar el tamaño correcto de freno. Como se sabe el torque dinámico disminuye con la velocidad de patinamiento, así que si su aplicación involucra altas velocidades elija bien el tipo de freno para su aplicación.

Al contrario de lo que sucede con el torque dinámico, al aumentar las RPM, aumenta la capacidad de disipar el calor generado por la fricción. Cuando la placa móvil se encuentra girando, el efecto ventilador que esto provoca aumenta proporcionalmente con la velocidad. Cabe destacar que aún en reposo la unidad debe elegirse para poder disipar calor por convección y radiación, mas aún no colaboran con la refrigeración del equipo.

El desgaste de las superficies de fricción depende de la presión de atracción entre las caras de contacto y la velocidad relativa a la que éstas hacen contacto. Hay muchas variables involucradas al intentar predecir la vida útil de un freno, de las cuales la velocidad angular es probablemente la más influyente.

El desgaste será mayor cuanto más tiempo las dos superficies estén en contacto, o bien, a mayor velocidad en el acople, más tiempo patinarán las superficies y se producirá mayor desgaste.

La operación a baja velocidad presenta consideraciones a tener en cuenta. Los frenos que trabajan por debajo de las 100 RPM se comportan diferentes de aquellos que lo hacen por arriba y esto se debe al asentamiento de las superficies de fricción, que se tocará en el siguiente punto.

Cuando el freno es nuevo, las superficies de fricción, si bien se encuentran rectificadas, presentan rayas o rugosidades que generan puntos altos sobresalientes. Cuando las dos superficies se acoplan por primera vez sólo unos pocos puntos hacen contacto dando como resultado una superficie de contacto muy reducida y la consecuente merma en el torque.

Para superar este proceso de asentamiento usualmente se requieren unos pocos cientos de accionamientos, dependiendo de las condiciones de trabajo. A mayor velocidad relativa en el momento de acople y mayores inercias al acelerar y desacelerar, más corto es el proceso de asentamiento.

Por último aclarar como se dijo anteriormente; los frenos electromagnéticos tanto en motores c.c. como c.a. se mantienen activados o energizados para permitir el libre movimiento del rotor y frenan el mismo cuando la energía eléctrica ha sido retirada.

Las razones para que estos frenos electromagnéticos funcionen al revés es que por ejemplo para los servos de c.c. al iniciar el movimiento si el motor tuviera que vencer la resistencia mecánica del freno tendería a ser incorrecta la alineación del motor para ciertas aplicaciones ya que cuando el controlador sabe que el servo debió girar n vueltas, éste no se ha movido tratando de vencer la resistencia del freno.

Otra consideración importante sobre los frenos es que el eventual desgaste de las piezas del freno ocasionará que la respuesta del frenado sea más lenta con el pasar del tiempo, así que en aplicaciones de servosistemas en las cuales es sumamente importante la posición del motor para llevar a cabo una acción se debe estar muy atento para que los errores de posicionamiento producidos por el desgaste no afecten el proceso de producción.

1.2.2. *Encoder* rotativo

Un *encoder* rotativo, también llamado *encoder* de eje, es un dispositivo electromecánico que convierte la posición angular del eje en un código analógico o digital, convirtiendo al *encoder* en un transductor de ángulos. Los *encoders* son utilizados en muchas aplicaciones que requieren movimientos precisos del eje del motor, incluyendo controles industriales, robótica, lentes fotográficos de última generación, accesorios de entrada para computadoras (como los *mouse* optométricos) y radares de plataforma rotativa. Existen dos tipos principales de *encoders* el digital y el analógico.

1.2.2.1. *Encoders* digitales

Los *encoders* digitales son dispositivos que convierten el movimiento en una secuencia de pulsos digitales, ya sea que cuenten un solo *bit* o bien que decodifiquen una serie de *bits*; los pulsos pueden ser convertidos en medidas de posición absoluta y relativa. Existen dos tipos básicos de *encoders* digitales los *encoders* absolutos y los *encoders* incrementales o relativos.

Los *encoders* digitales rotativos cuentan con dos tipos básicos: el *encoder* absoluto donde una única palabra digital corresponde a cada posición en la rotación del eje, y el *encoder* incremental, el cual produce pulsos digitales conforme el eje del motor gira, permitiendo la medida de la posición relativa de dicho eje.

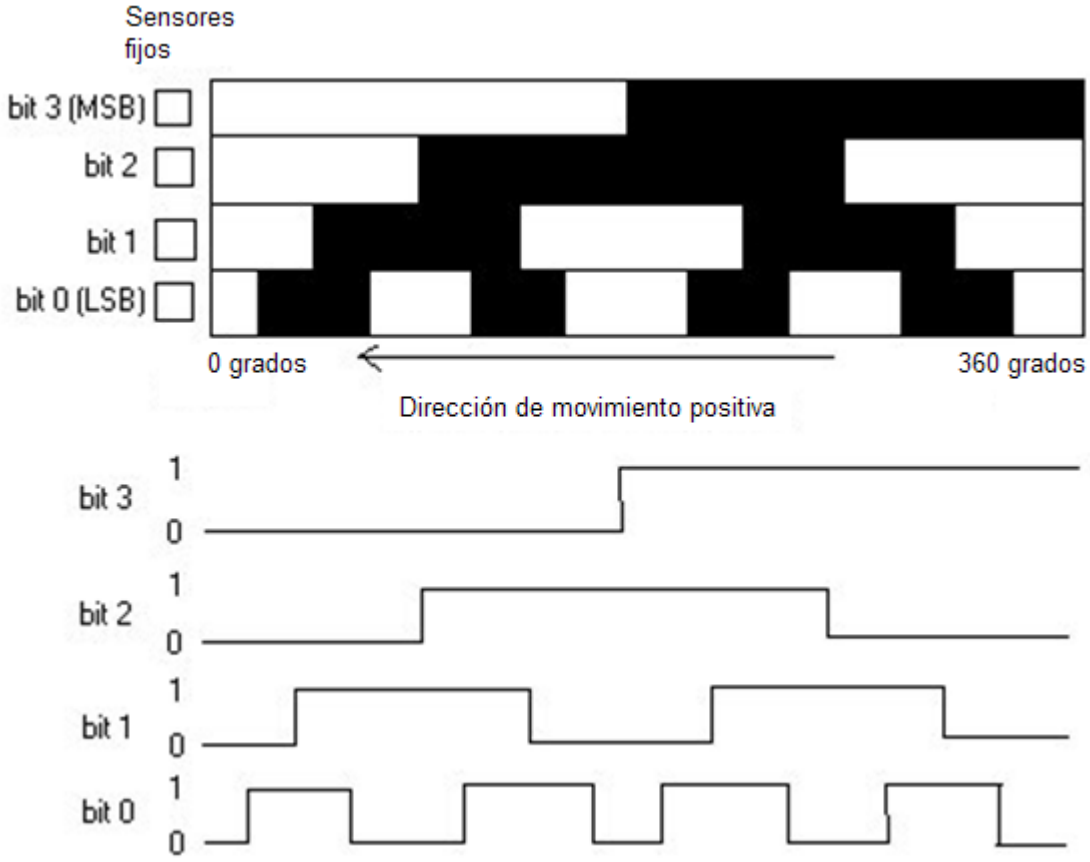
La mayoría de los *encoders* rotativos están hechos de discos plásticos o de vidrio, los cuales están codificados con un patrón fotográfico radial organizado en marcas. Así cuando las líneas en cada marca interrumpen el haz de luz entre el foto-emisor y el foto-detector, se producen los pulsos digitales.

1.2.2.1.1. *Encoders* absolutos

El disco óptico del *encoder* absoluto está diseñado para producir una palabra digital que distingue N posiciones distintas del eje. Por ejemplo, si tenemos 8 marcas en el disco, el *encoder* tiene la capacidad de producir 256 distintas posiciones o bien tiene una resolución angular de 1.406 grados (360/256). Los tipos más comunes de codificación numérica para el *encoder* absoluto son el código binario y el código *gray*.

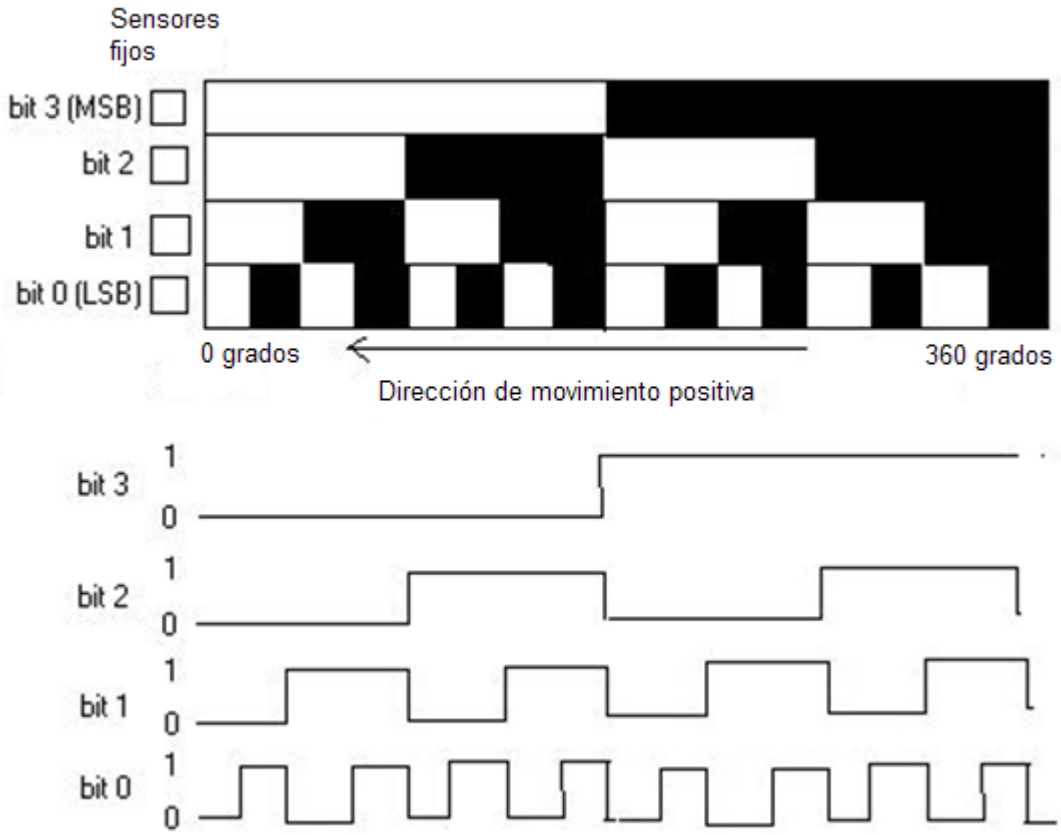
Para ilustrar la acción de un *encoder* absoluto, el patrón de marcas en un disco para código *gray* y para código binario natural de 4 *bits* se muestran en las figuras 10 y 11 respectivamente. Los patrones de las marcas y sus respectivos diagramas de temporización es lo que los foto-detectores leen conforme el disco gira junto con el eje. La salida del *encoder* para ambos códigos se muestra en la tabla I.

Figura 10. **Patrón de marcas y diagramas de temporización para código *gray* de 4 *bits***



Fuente: http://mechatronics.mech.northwestern.edu/design_ref/sensors/gray.jpg

Figura 11. **Patrón de marcas y diagramas de temporización para código binario natural de 4 *bits***



Fuente: http://mechatronics.mech.northwestern.edu/design_ref/sensors/binary.jpg

Tabla I. **Código binario y código *gray* de 4 *bits***

Código decimal	Rango de rotación en grados	Código binario	Código *gray*
0	0 - 22.5	0000	0000
1	22.5 - 45	0001	0001
2	45 - 67.5	0010	0011
3	67.5 - 90	0011	0010
4	90 - 112.5	0100	0110
5	112.5 - 135	0101	0111
6	135 - 157.5	0110	0101
7	157.5 - 180	0111	0100
8	180 - 202.5	1000	1100
9	202.5 - 225	1001	1101
10	225 - 247.5	1010	1111
11	247.5 - 270	1011	1110
12	270 - 292.5	1100	1010
13	292.5 - 315	1101	1011
14	315 - 337.5	1110	1001
15	337.5 - 360	1111	1000

Fuente: http://mechatronics.mech.northwestern.edu

El código *gray* está diseñado para que sólo una marca (1 *bit*) cambie de estado para cada transición de conteo, diferente al código binario natural en el cual múltiples marcas (múltiples *bits*) cambian en ciertas transiciones de conteo. Este efecto puede ser observado con claridad en la tabla I.

Para el código *gray*, la variación durante la transición es solo de un conteo, distinto al código binario, en el cual la variación puede ser de varias cuentas.

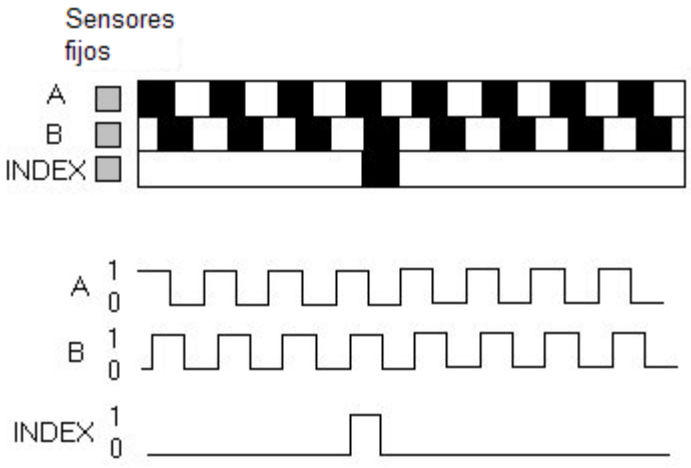
Entonces el código *gray* ayuda a no tener pérdida de datos entre transiciones de conteo, por ejemplo la transición de 134 a 135 grados listada en la tabla I, entrega una transición de salida en código binario de 0101 => 0110. Como se ve, para este movimiento de un rango a otro (grados) se requieren dos transiciones de conteo que el *LSB* cambie de 1 a 0 y que el siguiente *bit* cambie de 0 a 1, lo cual implica un mayor tiempo invertido en dicha transición.

Contrario al código binario el código *gray* pasa de 134 a 135 grados con una transición de 0111=> 0101 cambiando sólo el segundo *bit*, esta característica de pasar de un conteo al siguiente con un solo cambio de *bit* disminuye las posibilidades de perder datos, asunto de vital importancia en aplicaciones de posicionamiento de alta velocidad.

1.2.2.1.2. *Encoder* incremental

El *encoder* incremental, algunas veces llamado *encoder* relativo, es mucho más simple en diseño con respecto al *encoder* incremental. Consiste de dos marcas y dos sensores para cada una de las marcas, cuyas salidas son conocidas como el canal A y el canal B. Conforme el eje rota, trenes de pulsos ocurren en ambos canales a una frecuencia proporcional a la velocidad de rotación del eje y la relación de fase entre las señales indica el sentido de rotación. El patrón del disco codificado y las señales A y B se ilustran en la figura 12.

Figura 12. **Patrón del disco codificado y señales A y B del *encoder* incremental**



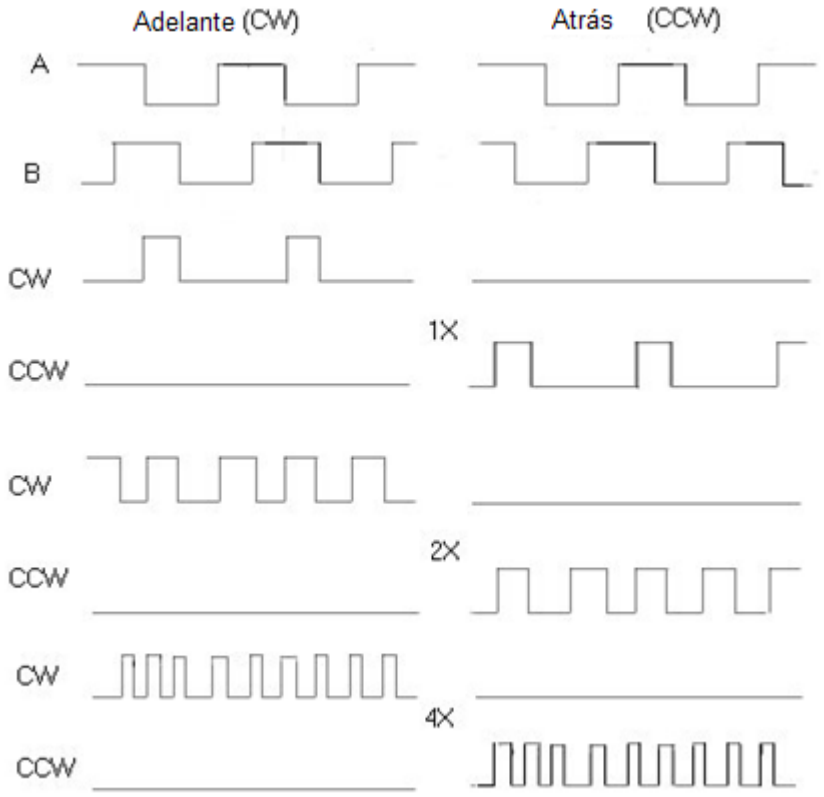
Fuente: http://mechatronics.mech.northwestern.edu/design_ref/sensors/increment.jpg

Realizando el conteo de los pulsos que genera el *encoder* y el conocimiento de la resolución del disco, es como se realiza la medición del movimiento angular. Los canales A y B son utilizados para determinar la dirección de rotación tomando en cuenta que canal guía al otro. Las señales de ambos canales se encuentran ¼ de ciclo desfasadas una de la otra y son conocidas como señales en cuadratura.

Con frecuencia un tercer canal de salida, llamado *indexado*, entrega un pulso por cada revolución, dicho pulso es muy útil a la hora de contar el número de revoluciones completas, además es de mucha utilidad para definir la posición cero o también llamado *home*.

Las señales de cuadratura A y B pueden ser decodificadas para entregar el sentido de rotación como se muestra en la figura 13. Decodificando las transiciones de A y B utilizando circuitos lógicos secuenciales de distintas maneras pueden proveer tres resoluciones distintas para los pulsos de salida, 1X, 2X y 4X. La resolución 1X provee un pulso simple por cada ciclo en una de las señales A o B, la resolución 4X entrega un pulso por cada flanco de transición en ambas señales A y B, con lo que se obtiene cuatro veces lo que nos brinda la resolución 1X.

Figura 13. **Determinación del sentido de giro por medio de las señales de cuadratura A y B**



Fuente: http://mechatronics.mech.northwestern.edu/design_ref/sensors/quadrature.jpg

El sentido de giro (horario o antihorario) es determinado por el nivel de una señal durante el flanco de transición de la otra señal. Por ejemplo, para la resolución 1X, si la señal A tiene un flanco descendiente (↓) y la señal B justo en ese instante tiene un valor de 1, implica un pulso en sentido horario, y si la señal B tiene un flanco descendente con A igual a 1 implica un giro antihorario. Si tuviéramos sólo una salida ya sea del canal A o el B sería imposible determinar el sentido de giro del motor.

1.2.2.2. Dispositivos analógicos

Los *encoders* digitales tienen su contraparte analógica, a la luz de la duda si se les puede llamar *encoders* analógicos porque no utilizan discos ni nada parecido, pero si nos apegamos firmemente a que un *encoder* es un dispositivo capaz de medir la posición del eje de un motor sin importar como esté compuesto, podríamos decir que el *resolver*, el dispositivo analógico que analizaremos a continuación, es un *encoder* analógico.

Estos dispositivos analógicos siguen utilizando señales pero en este caso, como es de suponer, ya no son trenes de pulsos o señales digitales sino entran en juego las señales seno y coseno las cuales, si se es un poco observador, se encuentran desfasadas 90^0 una de la otra. Además estos dispositivos funcionan bajo el fenómeno del electromagnetismo y su muy conocido principio de inducción para generar las señales en cuadratura.

1.2.2.2.1. *Resolver*

Un *resolver* es un tipo de transformador eléctrico rotativo utilizado para medir grados de rotación, es considerado un dispositivo análogo y es la contraparte del *encoder*.

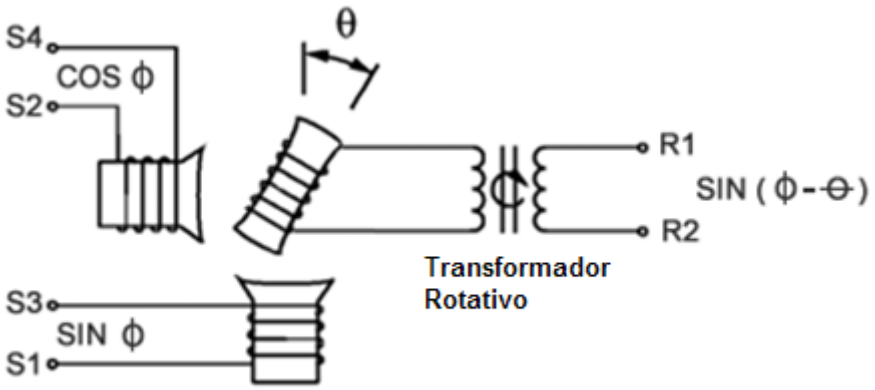
El tipo más común de *resolver* es el *resolver* de trasmisión sin escobillas, por fuera este tipo de *resolver* puede lucir como un motor eléctrico pequeño ya que tiene un rotor y un estator. Por dentro la configuración de sus cables arrollados en forma de bobina lo hacen diferente. La parte del estator aloja tres bobinas (cables arrollados), uno para la excitación y los dos restantes son bobinas bifásicas (usualmente marcados como X y Y). La bobina de excitación se encuentra en la parte de arriba, y de hecho es una de las bobinas de un transformador rotativo.

Este transformador energiza el rotor, eliminando la necesidad de escobillas o de limitar la rotación del mismo. Las dos bobinas restantes se encuentran en el fondo del estator enrolladas en un núcleo laminado, configuradas a 90^0 una de la otra. El rotor aloja una bobina, la cual es la bobina secundaria del transformador rotativo, y una bobina primaria en el estator (bobina de excitación), excitando las bobinas bifásicas que se encuentran en la laminación del fondo del estator.

La bobina primaria del transformador, fija en el estator, es excitada por una corriente eléctrica sinusoidal, la cual por inducción electromagnética logra que una corriente fluya por las bobinas secundarias (X y Y) situadas en el estator. Estas bobinas situadas en ángulo recto una con respecto de la otra en el estator, producen un corriente de retroalimentación seno y coseno por el mismo proceso de inducción.

Las magnitudes relativas de los voltajes bifásicos son medidas y utilizadas para determinar el ángulo del rotor relativo al estator. Una vez que se ha cumplido una revolución, las señales de retroalimentación repiten sus formas de onda, un esquema del *resolver* se muestra en la figura 14.

Figura 14. **Esquema de un *resolver***



Fuente: http://amci.com/tutorials/images/brushless-resolver-control-transformer.gif

Los *resolver* básicos son bipolares, esto significa que la información angular es el ángulo mecánico del estator. Estos dispositivos pueden entregar el ángulo absoluto de posición.

Otros tipos de *resolver* son los multipolares, éstos tienen 2*p polos, y por lo tanto pueden entregar p ciclos en una rotación del rotor (ángulo eléctrico = ángulo mecánico*p). Algunos *resolver* también incluyen ambos tipos, con el tipo bipolar usado para la posición absoluta y el multipolar para una posición más precisa.

Los *resolvers* multipolo pueden ser usados para monitorear motores eléctricos multipolo. Este dispositivo puede ser utilizado en cualquier aplicación en la cual la lectura exacta de la rotación de un objeto con respecto de otro es necesaria, tal como antenas rotativas o robots. En la práctica, el *resolver* es usualmente montado directamente en el motor eléctrico.

Resolver diferencial. Este tipo de *resolver* combina dos bobinas bifásicas para el primario y otro juego de dos bobinas bifásicas para el secundario. La relación del ángulo eléctrico entregado por el juego de bobinas secundarias y los demás ángulos es la siguiente, ángulo eléctrico secundario, ángulo mecánico y ángulo eléctrico primario. Estos *resolver* fueron utilizados, en algún tiempo, para calcular funciones trigonométricas sin ayuda de computadoras.

Otros tipos de *resolver* incluyen: *resolver* receptor, que son utilizados a la inversa de los *resolvers* transmisores. Las bobinas bifásicas son excitadas, la relación entre las señales seno y coseno representan un ángulo eléctrico. El sistema gira el rotor para obtener voltaje cero en la bobina del rotor. En esta posición, el ángulo mecánico del rotor es igual al ángulo eléctrico aplicado al estator.

1.2.2.3. Dispositivos analógicos versus digitales

La idea de esta sección del documento es hacer notar ciertas características, que pueden ser importantes, de los dispositivos analógicos y los dispositivos digitales. Para empezar los dispositivos analógicos hacen conteos de líneas por revolución y no de pulsos por revolución como los digitales.

Como se explicó el *resolver* utiliza señales seno y coseno para calcular la posición del eje, entonces se le denomina línea al ciclo completo de una señal seno o coseno, por ejemplo si un *resolver* entrega 100 ciclos seno y coseno en una revolución, entonces se dice que el *resolver* realiza un conteo de 100 líneas por revolución.

Un *resolver* es mucho más robusto que un *encoder* digital ya que éste puede ser instalado tanto en ambientes con temperaturas altas como bajas, es resistente a los impactos generados por maquinaria como brazos robóticos de ensamblaje. Por el contrario dichas características de operación no son idóneas para los *encoders* digitales ya que sus circuitos digitales no pueden soportar altas o bajas temperaturas y sus discos codificados están hechos de materiales delicados que no están diseñados para soportar los constantes impactos de ciertas aplicaciones industriales como por ejemplo los brazos robóticos de ensamblaje.

El ancho de banda que utiliza un *encoder* digital está limitado al medio por el cual se transmiten los datos, generalmente para estos dispositivos es cobre y entre mayor resolución tenga el *encoder* digital, mas *bits* serán requeridos y un mayor ancho de banda será demandado, una vez el cobre esté saturado será imposible mejorar dicho ancho de banda. Además las señales digitales son altamente vulnerables a la introducción de ruido en sus señales ocasionando errores al transmitir información.

Los dispositivos analógicos tienen otra ventaja en este sentido ya que a diferencia de los *encoders* digitales, los cuales obtienen información en base a los niveles de voltaje (alto o bajo) que genera el *encoder* y que es precisamente lo que los hace vulnerables al ruido, los dispositivos analógicos funcionan midiendo o contando los cruces por cero que realizan las ondas seno y coseno.

Vale la pena aclarar que el ruido si afecta a los dispositivos analógicos pero ya que estas señales son diferenciales, son mucho menos vulnerables a los errores causados por el ruido.

Las razones antes mencionadas entre otras son las causantes que los dispositivos analógicos para medir la posición del eje de un motor aún se encuentren vigentes dentro de un mundo que cada día se encuentra más inmerso en el mundo digital, y aclarar que los *resolvers* tampoco se salvan ya que existen convertidores analógicos-digitales que se utilizan para combinar las características robustas de estos equipos con la delicadeza y exactitud de la tecnología digital y hacer del *resolver* un equipo muy versátil para las diversas aplicaciones de la industria actual.

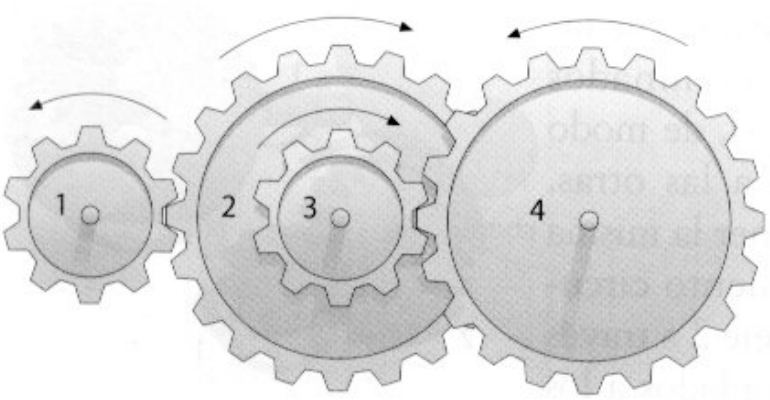
1.2.3. Engranajes

Antes de iniciar con el tema, los servomotores no se tocan en esta sección ya que de ellos de dió una descripción bastante detallada al inicio del capítulo, así que se continúa con la cuarta y quinta parte de los servosistemas, las cajas de engranajes y los mecanismos finales.

La razón de que ambos temas se toquen en esta parte del documento es básicamente eliminar la necesidad de redundar ya que tanto las cajas de engranajes como los mecanismos finales están hechos de lo mismo, engranajes. Así que a continuación se hablará sobre qué es un engranaje, las partes de los mismos y los tipos que existen.

Se denomina engranaje o rueda dentada al mecanismo utilizado para transmitir potencia de un componente a otro dentro de una máquina. Los engranajes están formados por dos ruedas dentadas, de las cuales la mayor se denomina corona y la menor piñón. Un engranaje sirve para transmitir movimiento circular mediante contacto de ruedas dentadas tal y como lo muestra la figura 15.

Figura 15. **Sistemas de engranajes**



Fuente: http://roble.pntic.mec.es/jprp0006/tecnologia/2eso_recursos/unidad5_mecanismos/ejercicios/hot_pot_mecanismos/imaxeneshot/engranaje_compuesto.jpg

Una de las aplicaciones más importantes de los engranajes es la transmisión del movimiento desde el eje de una fuente de energía, como puede ser un motor eléctrico, hasta otro eje situado a cierta distancia y que ha de realizar un trabajo, de manera que una de las ruedas está conectada por la fuente de energía y es conocido como engranaje motor y la otra está conectada al eje que debe recibir el movimiento del eje del motor y que se denomina engranaje conducido. La principal ventaja que tienen las transmisiones por engranajes respecto a las transmisiones por poleas es que no patinan como las poleas, con lo que se obtiene exactitud en la relación de transmisión.

1.2.3.1. Cálculo de engranajes

En un engranaje se deben diferenciar las partes importantes como la corona, que es la parte exterior, donde están tallados los dientes y el cubo, la parte central del engranaje, por medio de la cual se fija al eje. Además se deben conocer ciertas características y dimensiones fundamentales sobre las cuales se hablará a continuación.

La circunferencia que definirá la superficie por la cual el engranaje rueda sin deslizar la llamaremos circunferencia primitiva. El diámetro primitivo (d) es el que corresponde a la circunferencia primitiva. El número de dientes (z) es el número total de dientes de la corona del engranaje en toda su circunferencia. El paso (p) es el arco de la circunferencia, sobre la circunferencia primitiva, entre los centros de los dientes consecutivos. Con todo lo anterior se podrán calcular las relaciones principales de un engranaje tales como:

$$\text{Circunferencia primitiva} = \pi * d$$

$$\text{Circunferencia primitiva} = z * p$$

por lo tanto se observa que:

$$\pi * d = z * p$$

esto es:

$$\frac{d}{z} = \frac{p}{\pi} = m$$

de manera que se llamará módulo (m) de un engranaje a la relación que existe entre el diámetro primitivo y el número de dientes que es el mismo que el del paso y π.

El módulo es una magnitud de longitud, expresada en milímetros, para que dos engranajes puedan engranar tienen que tener el mismo módulo, el módulo podría tomar un valor cualquiera, pero en la práctica está normalizado según el siguiente criterio:

De 1 a 4 en incremento de 0.25 mm
De 4 a 7 en incrementos de 0.50 mm
De 7 a 14 en incrementos de 1 mm
De 14 a 20 en incrementos de 2 mm

Ahora serán discutidas las dimensiones de los engranajes, se iniciará con el diámetro exterior que es la circunferencia que pasa por la parte exterior de la cabeza de los dientes. Diámetro exterior (de) es el que corresponde a la circunferencia exterior. Circunferencia interior es la que pasa por la base de los dientes. Diámetro interior (di) es el que corresponde a la circunferencia interior.

Cabeza de diente (hc) es la parte del diente comprendida entre la circunferencia primitiva y la exterior, toma el valor del módulo hc = m. Pié de diente (hp) es la parte del diente comprendida entre la circunferencia interior y la primitiva, toma el valor de 1.25 veces el módulo. Altura del diente (h) es la distancia entre la circunferencia interior y la exterior por lo que tiene el valor de 2.25 veces el módulo y finalmente longitud de diente (b) que es la anchura de la corona, sobre la que se tallan los dientes, en general suele tener un valor de 10 veces el módulo.

En el sistema inglés de unidades, con la pulgada como unidad de longitud, el cálculo de engranajes emplea el denominado diámetro *pitch.* Para un engranaje dado, el diámetro *pitch* (Pt) es igual al número de dientes por pulgada en el diámetro primitivo, la relación del diámetro *pitch* y el módulo es:

$$m = \frac{25.4}{\mathrm{Pt}}$$

1.2.3.2. Tipos de engranajes

La principal clasificación de los engranajes se efectúa según la disposición de sus ejes de rotación y según los tipos de dentado, de acuerdo a estos criterios existen los siguientes tipos de engranajes de ejes paralelos y de ejes perpendiculares.

1.2.3.2.1. Engranajes de ejes paralelos

Los engranajes de ejes paralelos, como su nombre lo dice, los ejes sobre los cuales giran sus ruedas dentadas se encuentran con una relación espacial de paralelismo y a continuación se hablará sobre los diversos tipos.

1.2.3.2.1.1. Engranajes cilíndricos de dientes rectos

Los engranajes cilíndricos rectos, son el tipo de engranaje más simple y corriente que existe. Se utiliza generalmente para velocidades pequeñas y medianas, a grandes velocidades si no son rectificados o ha sido corregido su tallado, producen ruido cuyo nivel depende de la velocidad de giro que tengan. Las partes más importantes de un engranaje son: los dientes y el módulo.

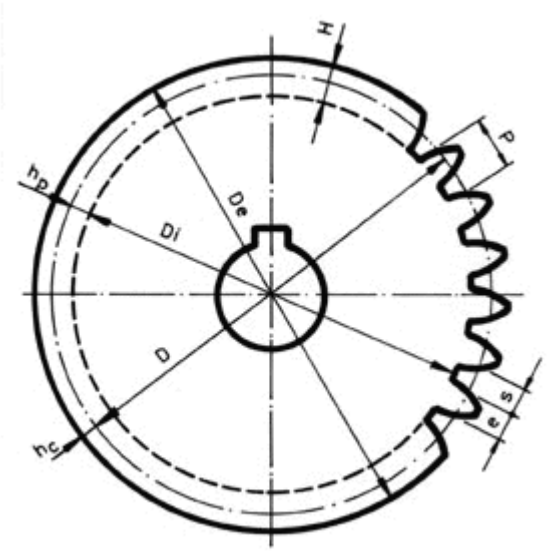
Los dientes son los que realizan el esfuerzo de empuje y transmiten la potencia desde los ejes motrices a los ejes conducidos. El perfil del diente, o sea la forma de sus flancos, está constituido por dos curvas envolventes del círculo, simétricas respecto al eje que pasa por el centro del mismo. El módulo de un engranaje es una característica de magnitud que se define como la relación entre la medida del diámetro primitivo expresado en milímetros y el número de dientes. En los países anglosajones se emplea otra característica llamada *Diametral Pitch*, que es inversamente proporcional al módulo.

El valor del módulo se fija mediante el cálculo de resistencia de materiales en virtud de la potencia a transmitir y en función de la relación de transmisión que se establezca, el tamaño de los dientes está normalizado, el módulo está indicado por números y dos engranajes que engranen deben tener el mismo módulo. La circunferencia primitiva, mencionada dentro de la descripción del módulo, es la circunferencia a lo largo de la cual engranan los dientes.

Con relación a la circunferencia primitiva se determinan todas las características que definen los diferentes elementos de los dientes de los engranajes. Otra característica importante sobre estos engranajes es el número de dientes, se simboliza con la letra Z y es fundamental para calcular la relación de transmisión.

El número de dientes de un engranaje no debe de estar por debajo de los 18 dientes cuando el ángulo de presión es 20^0 ni por debajo de 12 cuando el ángulo de presión es 25^0. La relación de transmisión es la relación de giro que existe entre el piñón conductor y la rueda conducida. La relación de transmisión recomendada tanto en caso de reducción como de multiplicación depende de la velocidad que tenga la transmisión, la figura 16 muestra las características más importantes de los engranajes cilíndricos rectos.

Figura 16. **Partes del engranaje cilíndrico recto**



Fuente: http://www2.ing.puc.cl/~icm2312/apuntes/engrana/dibujo1.gif

1.2.3.2.1.2. Engranajes cilíndricos de dentado helicoidal

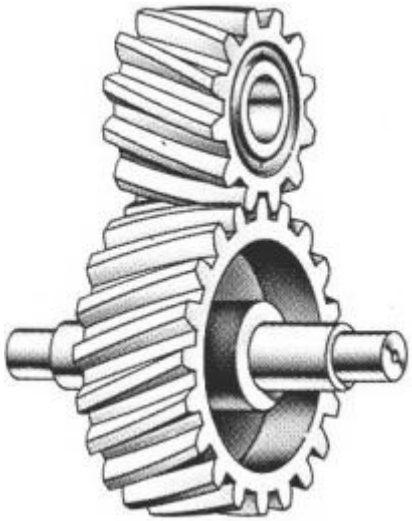
Los engranajes cilíndricos de dentado helicoidal están caracterizados por su dentado oblicuo con relación al eje de rotación. En estos engranajes el movimiento se transmite igual que en los cilindros de dientes rectos pero con mayores ventajas. Los ejes de los engranajes helicoidales pueden ser paralelos o cruzarse a 90^0. Para eliminar el empuje axial el dentado puede hacerse doble helicoidal.

Los engranajes helicoidales tienen la ventaja que transmiten más potencia que los rectos y también pueden transmitir más velocidad, son más silenciosos y más duraderos; además pueden transmitir el movimiento de ejes que se corten.

De sus inconvenientes se puede decir que se desgastan más que los rectos, son más caros de fabricar y necesitan generalmente más engrase que los rectos. Lo más característico de un engranaje cilíndrico helicoidal es la hélice que forma, siendo considerada la hélice como el avance de una vuelta completa del diámetro primitivo del engranaje. De esta hélice se deriva el ángulo β que forma el dentado con el eje axial.

Este ángulo tiene que ser igual para las dos ruedas que engranan pero de orientación contraria, o sea uno a la derecha y el otro a la izquierda. Su valor se establece a priori de acuerdo con la velocidad que tenga la transmisión, los datos orientativos de este ángulo son los siguientes: para velocidad lenta $\beta = 5\text{-}10^0$; para velocidad normal $\beta = 5\text{-}10^0$ y para velocidad elevada $\beta = 30^0$ y las relaciones de transmisión recomendadas son similares a las relaciones de los engranajes rectos, la figura 17 nos muestra como lucen los engranajes cilíndricos helicoidales.

Figura 17. **Engranaje cilíndrico con dientes helicoidales**



Fuente: http://cerezo.pntic.mec.es/rlopez33/bach/tecind1/Tema_6/imagenes/engra4.png

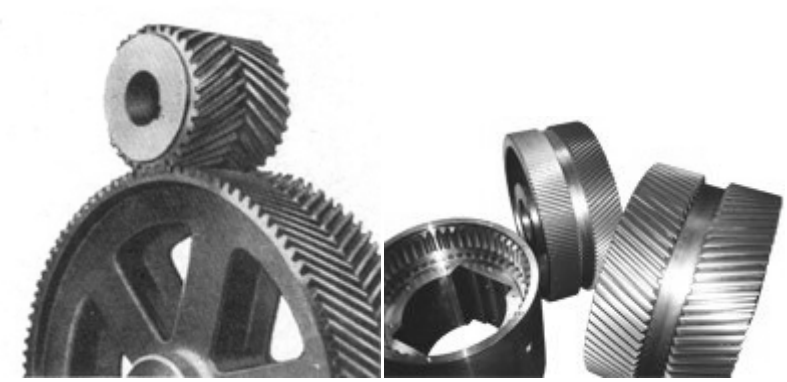
1.2.3.2.1.3. Engranajes dobles helicoidales

Este tipo de engranajes fueron inventados por el fabricante de automóviles francés André Citroën, y el objetivo que consiguen es eliminar el empuje axial que tienen los engranajes helicoidales simples. Los dientes de los dos engranajes forman una especie de V.

Los engranajes dobles son una combinación de hélice derecha e izquierda. El empuje axial que absorben los apoyos o cojinetes de los engranajes helicoidales es una desventaja de ellos y ésta se elimina por la reacción del empuje igual y opuesto de una rama simétrica de un engrane helicoidal doble.

Un engrane de doble hélice sufre únicamente la mitad del error de deslizamiento que el de una sola hélice o del engranaje recto. Toda discusión relacionada a los engranes helicoidales sencillos (de ejes paralelos) es aplicable a los engranajes helicoidales dobles, exceptuando que el ángulo de la hélice es generalmente mayor para los helicoidales dobles, puesto que no hay empuje axial, la figura 18 muestra este tipo de engranajes.

Figura 18. **Engranajes dobles helicoidales**



Fuente: http://www.raelca.it/15.jpg; http://www.engranajesdeusto.com/es/eng/17.jpg

1.2.3.2.2. Engranajes de ejes perpendiculares

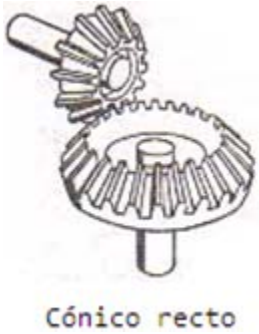
De manera similar a los engranajes de ejes paralelos los engranajes de ejes perpendiculares están formados por ruedas dentadas pero en este caso los ejes a los que están acopladas dichas ruedas tienen una relación espacial perpendicular, a continuación se hablará de los tipos de engranajes de ejes perpendiculares más comunes.

1.2.3.2.2.1. Engranajes cónicos de dientes rectos

Los engranajes cónicos se fabrican a partir de un tronco de cono, formándose los dientes por fresado de su superficie exterior. Estos dientes pueden ser rectos helicoidales o curvos. Esta familia de engranajes soluciona transmisión entre ejes que se cortan y que se cruzan.

Los engranajes cónicos de dientes rectos efectúan la transmisión de movimiento de ejes que se cortan en un mismo plano, generalmente en ángulo recto, por medio de superficies cónicas dentadas. Los dientes convergen en el punto de intersección de los ejes. Son utilizados para efectuar reducción de velocidad con ejes en 90^0 y se ilustran en la figura 19. Estos engranajes generan más ruido que los engranajes cónicos helicoidales. Se utilizan en transmisiones antiguas y lentas, en la actualidad su uso es bastante reducido.

Figura 19. **Engranajes cónicos de dientes rectos**



Fuente: http://1.bp.blogspot.com/_qJkurG0QM5A/SSnCtxpOk4I/AAAAAAAAADM/86ljlcQNmuw/s400/14.bmp

1.2.3.2.2.2. Engranajes cónicos de dientes helicoidales

Se utilizan para reducir la velocidad en un eje de 90^0. La diferencia con el cónico recto es que posee una mayor superficie de contacto. Es de un funcionamiento relativamente silencioso. Además pueden transmitir el movimiento de ejes que se corten y se ilustran en la figura 20.

Figura 20. **Engranajes cónicos con dentado helicoidal**

Fuente: http://1.bp.blogspot.com/_qJkurG0QM5A/SSnCtxpOk4I/AAAAAAAAADM/86ljlcQNmuw/s400/14.bmp

1.2.3.2.2.3. Tornillo sin fin y corona

Finalmente el último grupo de engranajes paralelos es el de los tornillos sin fin y coronas, el tornillo sin fin es un mecanismo diseñado para transmitir grandes esfuerzos y como reductores de velocidad aumentando la potencia de transmisión, generalmente trabajan con ejes que se cruzan a 90^0.

Tiene la desventaja de no ser reversible en el sentido de giro, sobre todo en grandes relaciones de transmisión y de consumir en rozamiento una gran cantidad de potencia. Con el fin de convertir el punto de contacto en una línea de contacto y así distribuir mejor la fuerza a transmitir, se suelen fabricar tornillos sin fin que engranen con una corona glóbica.

Otra forma de distribuir la fuerza a transmitir es utilizar una rueda helicoidal como corona y hacer el tornillo sin fin glóbico, de esta manera se consigue aumentar el número de dientes que están en contacto, el mecanizado de las coronas de engranaje para tornillo sin fin se puede realizar por medio de fresas normales o por fresas madre y el mecanizado del tornillo sin fin se puede hacer por medio de fresas biocónicas o fresas frontales. La figura 21, muestra una variedad de engranajes, los cuales han sido mencionados en párrafos anteriores.

Figura 21. **Tornillo sin fin y corona**

Fuente: http://roble.pntic.mec.es/jprp0006/tecnologia/1eso_recursos/unidad09_mecanismos/actividades/ejercicios/hot_pot_mecanismos/imaxeneshot/sinfin.jpg

1.2.3.3. Otros tipos de engranajes

A continuación se hablará sobre otros tipos de engranajes que funcionan para diversas aplicaciones, además algunos de ellos están formados por más de dos ruedas dentadas o bien son combinaciones de algunos tipos mencionados anteriormente.

1.2.3.3.1. Engranajes planetarios

Un engranaje planetario o engranaje epicicloidal es un sistema de engranajes (tren de engranajes) consistente en uno o más engranajes externos o satélites que rotan sobre un engranaje central o planeta. Típicamente, los satélites se montan sobre un brazo móvil o portasatélites que a su vez puede rotar en relación al planeta. Los sistemas de engranajes planetarios pueden incorporar también el uso de un engranaje anular externo o corona, que engrana con los satélites.

La velocidad de transmisión en un sistema de engranaje planetario es muy poco intuitiva, especialmente porque hay varias formas de convertir la rotación de entrada en una de salida. Los tres componentes básicos de un engranaje epicicloidal son el planeta, el portasatélites y la corona. El planeta es el eje central, el portasatélites sujeta a uno o más engranajes satélites periféricos, del mismo tamaño, engranados con el planeta y la corona que es un anillo externo con dientes en su cara interna que engrana con el o los satélites.

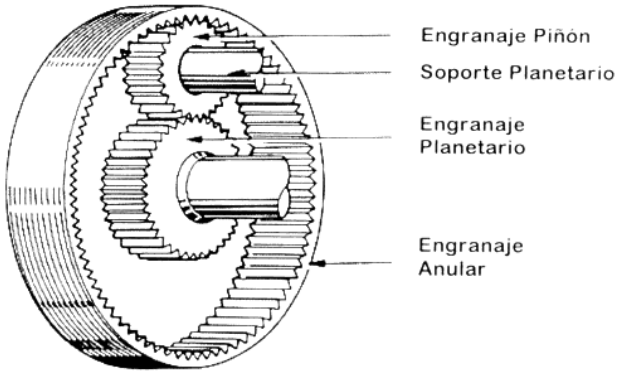
En cualquier sistema de engranaje planetario, uno de estos tres componentes básicos permanece estacionario, uno de los dos restantes es la entrada, proporcionando potencia al sistema, y el último componente es la salida, recibiendo la potencia del sistema. La relación de la rotación de entrada con la de salida depende del número de dientes de cada rueda y de qué componente permanezca estacionario.

Una situación es cuando el portasatélites permanece estacionario y el planeta se usa como entrada. En este caso, los satélites simplemente rotan sobre sus propios ejes a una velocidad determinada por el número de dientes de cada engranaje.

Si el planeta tiene *P* dientes y cada satélite tiene *S* dientes, entonces la relación es igual a *P*/*S*. Por ejemplo, si el planeta tiene 24 dientes y cada satélite tiene 16, entonces la relación es 24/16 o 3/2, lo que significa que cada giro en sentido horario produce 1,5 giros en sentido antihorario en los satélites.

Esta rotación de los satélites puede a su vez impulsar la corona, en una relación correspondiente. Si la corona tiene *C* dientes, entonces rotará *S*/*C* giros por cada uno de los satélites. Por ejemplo, si la corona tiene 64 dientes y los satélites 16, un giro en sentido horario de éstos resulta en 16/64 o 1/4 de giro en el mismo sentido de la corona. La figura 22 ilustra a los engranajes planetarios.

Figura 22. **Engranajes planetarios**



Fuente: http://www.naikontuning.com/mecanica/trasnmision/caja-automatica/03.gif

1.2.3.3.2. Engranajes de cremallera

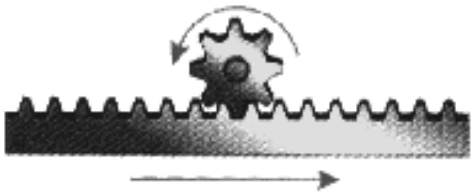
El mecanismo de cremallera aplicado a los engranajes lo constituyen una barra con dientes la cual es considerada como un engranaje de diámetro infinito y un engranaje de diente recto de menor diámetro, y sirve para transformar un movimiento de rotación del piñón en un movimiento lineal de la cremallera.

Quizás la cremallera más conocida sea la que equipan los tornos para el desplazamiento del carro longitudinal. Su velocidad se calcula de acuerdo a la siguiente ecuación:

$$v = (n * z * p) / 60[m / s]$$

donde n es la velocidad angular, z es el número de dientes de la rueda dentada y p es el paso. La figura 23 ilustra al mecanismo de engranaje de cremallera.

Figura 23. **Engranaje de cremallera**



Fuente: http://www.educaciontecnologica.cl/imagenes_motor_transmision/engra2.gif

1.2.3.3.3. Engranaje diferencial

El engranaje diferencial tiene por objetivo permitir que cuando el vehículo dé una curva, sus ruedas propulsoras puedan describir sus respectivas trayectorias sin patinar sobre el suelo. La necesidad de este dispositivo se explica por el hecho de que al dar una curva el coche, las ruedas interiores a la misma recorren un espacio menor que las situadas en el lado exterior, puesto que las primeras describen una circunferencia de menor radio que las segundas.

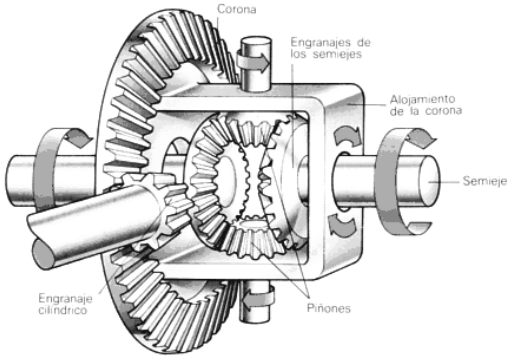
El engranaje diferencial está constituido por una serie de engranajes dispuestos de tal forma que permite a las dos ruedas motrices de los vehículos girar a velocidad distinta cuando circulan por una curva.

Así, si el vehículo toma una curva a la derecha, las ruedas interiores giran más despacio que las exteriores, y los satélites encuentran mayor dificultad en mover los planetarios de los semiejes de la derecha porque empiezan a rotar alrededor de su eje haciendo girar los planetarios de la izquierda a una velocidad ligeramente superior. De esta forma provocan una rotación más rápida del semieje y de la rueda motriz izquierda.

El engranaje diferencial está constituido por dos piñones cónicos llamados planetarios, unidos a extremos de los palieres de las ruedas y otros dos piñones cónicos llamados satélites montados en los extremos de sus eje porta satélites y que se engranan con los planetarios.

Una variante del diferencial convencional está constituida por el diferencial autoblocante que se instala opcionalmente en los vehículos todo-terreno para viajar sobre hielo o nieve o para tomar las curvas a gran velocidad en caso de los automóviles de competición, pero el mecanismo diferencial común se ilustra en la figura 24.

Figura 24. **Engranaje diferencial**



Fuente: http://www.automotriz.net/tecnica/images/conocimientos-basicos/39/operacion-diferencial.gif

1.2.3.3.4. Reductores de velocidad

El problema básico de las máquinas es reducir la alta velocidad de los motores a una velocidad utilizable por los equipos de las máquinas. Además de reducir se deben contemplar las posiciones de los ejes de entrada y salida y la potencia mecánica a transmitir.

Para potencias mayores se utilizan equipos reductores separados del motor. Los reductores consisten en pares de engranajes con gran diferencia de diámetros, de esta forma el engrane de menor diámetro debe dar muchas vueltas para que el de diámetro mayor de una vuelta, reduciendo la velocidad de giro. Para obtener grandes reducciones se repite este proceso colocando varios pares de engranes conectados uno a continuación del otro.

El reductor básico está formado por mecanismo de tornillo sin fín y corona. En este tipo de mecanismo el efecto del rozamiento en los flancos del diente hace que estos engranajes tengan los rendimientos más bajos de todas las transmisiones; dicho rendimiento se sitúa entre un 40 y un 90% aproximadamente, dependiendo de las características del reductor y del trabajo al que está sometido. Factores que elevan el rendimiento:

- Ángulos de avance elevados en el tornillo
- Rozamiento bajo (buena lubricación) del equipo
- Potencia transmitida elevada
- Relación de transmisión baja (factor más determinante)

Existen otras disposiciones para los engranajes en los reductores de velocidad, éstas se denominan conforme a la disposición del eje de salida (eje lento) en comparación con el eje de entrada (eje rápido). Así pues serían los llamados reductores de velocidad de engranajes coaxiales, paralelos, ortogonales y mixtos (paralelos + sin fin corona). En los trenes coaxiales, paralelos y ortogonales se considera un rendimiento aproximado del 97-98%, en los mixtos se estima entre un 70% y un 90% de rendimiento.

Además, existen los llamados reductores de velocidad de disposición epicicloidal, técnicamente son de ejes coaxiales y se distinguen por su formato compacto, alta capacidad de trasmisión de par y su extrema sensibilidad a la temperatura. Las cajas reductoras suelen fabricarse en fundición gris dotándola de retenes para que no salga el aceite del interior de la caja. La figura 25 muestra mecanismos reductores de velocidad.

Figura 25. **Mecanismos reductores de velocidad**

Fuente: http://dibujoindustrial.es/_Archivos/07_NOR14/Fig103.jpg

1.2.4. Servo controlador

Un servo controlador es un amplificador electrónico especial utilizado para alimentar motores o servomotores eléctricos, de acuerdo a la aplicación. Este dispositivo electrónico se encarga de monitorear e interpretar las señales de retroalimentación del motor y ajustar continuamente las desviaciones de un comportamiento predeterminado.

El servo controlador recibe una señal de comando de un sistema de control, amplifica esa señal, y transmite corriente eléctrica al servomotor para lograr generar un movimiento proporcional a la señal de comando. Típicamente la señal de comando representa la velocidad objetivo o velocidad deseada, pero puede también representar el torque requerido o bien una posición deseada.

Un sensor de velocidad acoplado al servomotor reporta constantemente la velocidad del motor al servo controlador. El servo controlador entonces compara la velocidad que constantemente retroalimenta el sensor de velocidad mencionado anteriormente, con la velocidad que debería tener el motor de acuerdo a la señal de comando. Para esto el servo controlador altera la frecuencia del voltaje que alimenta al motor (en el caso de motores de jaula de ardilla usados para servocontrol) y otras características de la alimentación del motor para que éste se mantenga a la velocidad predeterminada.

En un sistema correctamente configurado, el servomotor rota a una velocidad muy aproximada a la velocidad objetivo que recibe el servo controlador por parte del sistema de control. Una gran cantidad de parámetros tales como, la ganancia proporcional, la ganancia derivativa y la ganancia de retroalimentación entre otros, pueden ser ajustados para lograr el funcionamiento adecuado, a este proceso de ajustar las variables del sistema de control para mantener el funcionamiento deseado se conoce como *tunning.*

De lo citado en los párrafos anteriores se puede deducir que el controlador es el cerebro, la parte inteligente del servosistema, ya que si ésta parte no estuviera presente no servirían de nada los datos enviados por el *encoder* o *resolver* porque no habría quien los interprete. Además también tiene la capacidad de mejorar algunas características de los equipos que se encuentran unidos a él, un buen ejemplo es el caso del *resolver*.

Habiendo ya hablado del *resolver* y su mejor ancho de banda en comparación con el *encoder* digital, pues el controlador juega un papel importante dentro de esta ventaja ya que para que las líneas que cuenta el *resolver* son analógicas y el controlador es netamente digital, así que para que dichas cuentas (líneas por revolución) signifiquen algo para el controlador deben ser pasadas por un convertidor análogo digital, el cual cuantiza las señales analógicas (seno y coseno) y las conviertes en digitales.

Es durante este proceso que el ancho de banda del *resolver* se magnifica en una cantidad considerable ya que por ejemplo si la resolución analógica era de 100 líneas por revolución y la resolución del convertidor analógico-digital es de 512 el controlador hace posible que la resolución resultante sea la multiplicación de ambas, la del *resolver* y la del ADC.

Otro ejemplo de las mejoras que los controladores introducen a los servosistemas, en este caso para los dispositivos digitales, es evitar realizar el indexado para ubicar la posición en la que se encuentra el motor luego de que se ha parado la aplicación, ya se habló sobre esto en la parte de los *encoders* incrementales o relativos. Esto se logra a través de las señales de conmutación de las cuales hablaremos a continuación.

Como recordará, se hablo sobre las señales A, B y Z las cuales fungen como señales de cuadratura y señal de cruce por cero respectivamente. Ahora, las señales de conmutación le sirven al controlador para lograr ubicar de una manera mucho más eficiente la posición exacta del motor cuando se ha parado una aplicación, ya sea por fallo, paro de emergencia o cualquier otro motivo, y se desea iniciar de nuevo dicho proceso.

Las señales de conmutación transmiten impulsos generados por sensores *hall* que se encuentran instalados dentro del servomotor, como sabrá los sensores *hall* son sensores que generan pulsos al detectar los campos magnéticos.

Entonces al momento de energizar el motor, el cual tiene por lo general un sensor *hall* por polo, el sensor *hall* más cercano a la posición actual del motor detecta la presencia del campo magnético con mayor intensidad y envía una señal al controlador y al detectarla, éste conoce la posición del motor y en lugar de realizar varias revoluciones para encontrar el cruce por cero, literalmente se mueve solamente unos pocos grados y las señales de conmutación, conocidas como U, V, W, -U, -V y –W, le indican al controlador la posición del motor y así puede inicializar los parámetros de la aplicación y volver a funcionar.

Es también debido a estas señales de conmutación que los controladores utilizan un conector DB-15 para comunicarse con el servomotor. Si hacemos un conteo de las señales dentro del cable podemos justificar el tipo de conector, se empieza con las señales de cuadratura A y B, añadiendo la señal de cruce por cero Z; ahora como se ha visto las señales de conmutación son 6, con lo que se tienen nueve cables utilizados y añadiendo a estos nueve los cables de alimentación, cero y cinco voltios, se observa que tenemos 12 cables que sirven para entablar la comunicación del controlador con el servo y viceversa.

Es importante hacer énfasis en que el controlador únicamente sabe interpretar pulsos, el no sabe que hay un *encoder* dentro del motor, él únicamente entiende pulsos, sin embargo tiene la capacidad de procesar esos pulsos de varias formas. Una cuenta es el número de pulsos que se reciben por unidad del tiempo. Ahora el controlador es capaz de gobernar *encoders* análogos o digitales según la aplicación y también es capaz de procesar los conteos de acuerdo a la aplicación.

Además debe quedar claro que los controladores varían en sus capacidades de acuerdo a la aplicación y el precio que se esté dispuesto a pagar, además existen controladores versátiles a los cuales se les pueden agregar más funciones por medio de paquetes especializados para ciertas aplicaciones, también es vital dimensionarlos de acuerdo al equipo que controlan y configurarlos de una manera adecuada para obtener de ellos el mejor rendimiento.

1.3. Servocontrol

Un sistema de control está definido como un conjunto de componentes que pueden regular su propia conducta o la de otro sistema con el fin de lograr un funcionamiento predeterminado, de modo que se reduzcan las probabilidades de fallos y se obtengan los resultados buscados. Hoy en día los procesos de control son síntomas de la evolución que está sufriendo la industria a nivel general.

Estos sistemas se usan típicamente para sustituir a un trabajador pasivo que controla un determinado sistema (mecánico o eléctrico) con una posibilidad nula o casi nula de error y un grado de eficiencia mucho mayor que el del trabajador. Los sistemas más modernos en ingeniería automatizan procesos en base a muchos parámetros y reciben en nombre de Controladores de Automatización Programables (PAC).

Ahora que se conoce una definición de lo que es un sistema de control no será difícil imaginar, a grandes rasgos, que es el servocontrol. De hecho el servocontrol como tal es también un sistema de control, cuya finalidad es controlar las variables de una aplicación específica, ya sea un control de temperatura, control de humedad o bien como en nuestro caso el control de la posición, velocidad y aceleración del motor. Antes de continuar se definirán algunos términos básicos de los sistemas de control, para luego abordar el servocontrol y las estrategias que se utilizan para implementarlo.

El primer término a definir es la variable controlada, ésta es la variable que se debe mantener o controlar dentro de algún valor deseado, un ejemplo simple sería un control de temperatura en el cual dicha variable sería la temperatura.

El segundo término es el punto de control, el valor que se desea tenga la variable controlada. La variable manipulada, es la variable que se utiliza para mantener a la variable controlada en el punto de control y finalmente cualquier variable que ocasiona que la variable de control se desvié del punto de control se define como perturbación o trastorno, el origen de ésta es muy variado y ningún sistema por muy perfecto que sea está libre de ella (perturbación).

En algunos procesos la variable controlada se desvía del punto de control a causa de las perturbaciones. El termino control regulado se utiliza para referirse a los sistemas diseñados para compensar las perturbaciones. A veces la perturbación más importante es el punto de control mismo, ya que el punto de control puede cambiar en función del tiempo, lo cual es muy común en los procesos por lote, y en consecuencia la variable controlada debe ajustarse al punto de control; el término servocontrol se refiere a los sistemas de control que han sido diseñados con tal propósito, mantener en el punto de control o *set point* a las variables de control que varían en función de otros parámetros como el tiempo.

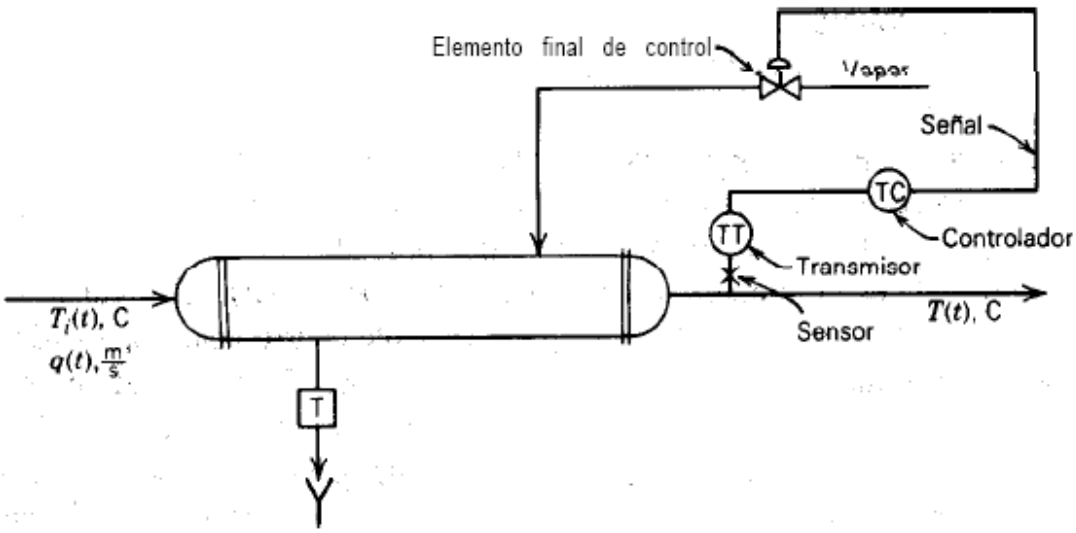
1.3.1. Estrategias de control

Las estrategias de control, como tal y como las describen Smith y Corripio en su libro control automático de procesos, son métodos que se utilizan para realizar el control adecuado basándose en las características propias de los sistemas, existen diversos tipos de estrategias de control y este apartado estará enfocado en las estrategias de control para lazo cerrado.

1.3.1.1. Control por retroalimentación

Este tipo de control también es conocido como control proporcional. En este procedimiento se toma la variable controlada y se retroalimenta al controlador para que éste pueda tomar una decisión. Es necesario comprender el principio de operación del control por retroalimentación para conocer sus ventajas y desventajas, para ayudar a dicha comprensión se presenta el ejemplo de un intercambiador de calor en la figura 26.

Figura 26. **Sistema de control del intercambiador de calor**



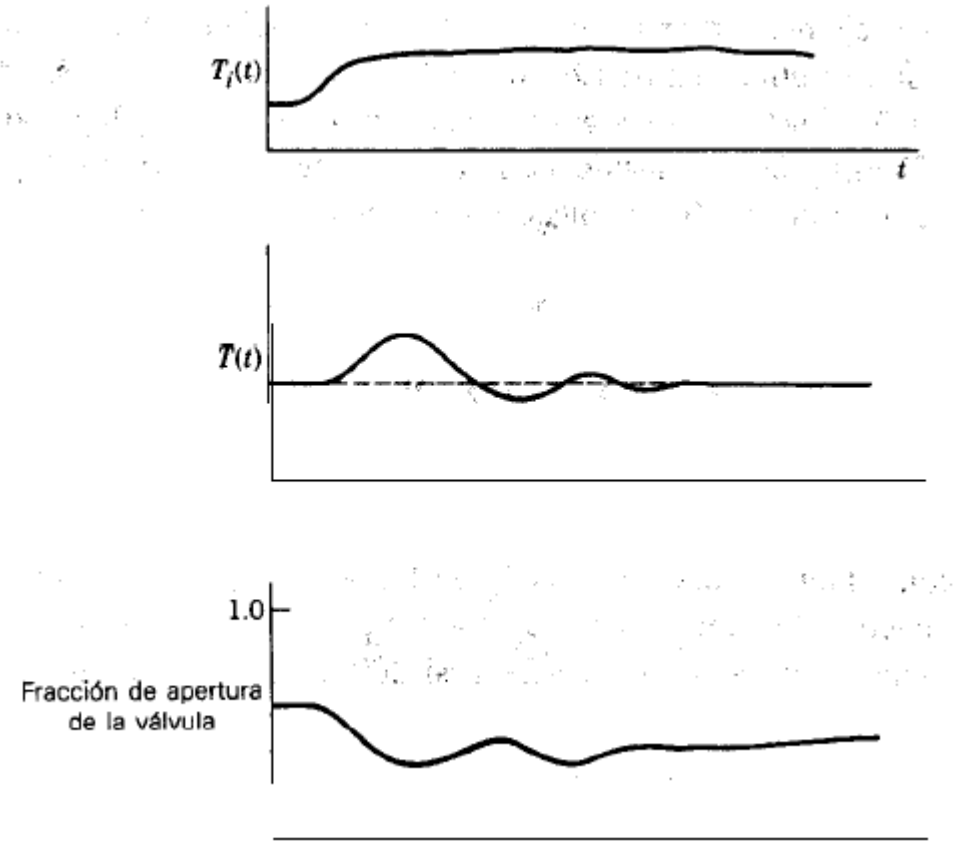
Fuente: Smith y Corripio. Sistemas de control automático. p. 19

Si la temperatura de entrada al proceso aumenta y en consecuencia crea una perturbación, su efecto se debe propagar a todo el intercambiador de calor antes de que cambie la temperatura de salida. Una vez que cambia la temperatura de salida, también cambia la señal del transmisor al controlador, en este momento el controlador detecta que debe compensar la perturbación mediante un cambio en el flujo de vapor, el controlador señala a la válvula cerrar su apertura y de este modo decrece el flujo de vapor.

Es interesante hacer notar que la temperatura de salida primero aumenta a causa del incremento en la temperatura de entrada, pero luego desciende, incluso por debajo del punto de control y oscila alrededor de éste hasta que se estabiliza. Esta respuesta oscilatoria demuestra que la operación del sistema de control por retroalimentación es esencialmente una operación de ensayo y error.

Cuando el controlador detecta que la temperatura de salida aumentó por arriba del punto de control, indica a la válvula que cierre, pero ésta cumple con la orden mucho más allá de lo necesario, en consecuencia la temperatura de salida desciende por debajo del punto de control. El controlador al percatarse de esto, señala a la válvula que se abra nuevamente un tanto para elevar la temperatura. El ensayo y error continúa hasta que la temperatura alcanza el punto de control donde permanece posteriormente, este comportamiento se ilustra en la figura 27.

Figura 27. **Respuesta de un control por retroalimentación**



Fuente: Smith y Corripio. Sistemas de control automático. p. 22

La ventaja del control por retroalimentación consiste en que es una técnica muy simple que compensa todas las perturbaciones. Cualquier perturbación puede afectar la variable controlada, cuando ésta se desvía del punto de control, el controlador cambia su salida para que la variable retorne al punto de control. El circuito de control no detecta que tipo de perturbación entra en el proceso, únicamente trata de mantener la variable controlada en el punto de control y de esta manera compensar cualquier perturbación.

La desventaja del control por retroalimentación estriba en que únicamente puede compensar la perturbación hasta que la variable controlada se ha desviado del punto de control, esto es, la perturbación debe propagarse por todo el proceso antes que la pueda compensar el control por retroalimentación.

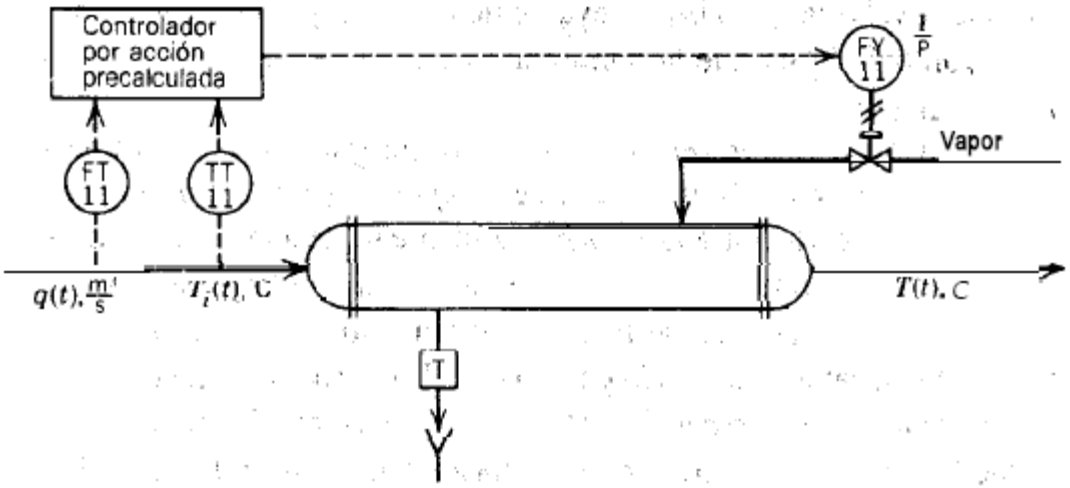
1.3.1.2. Control por acción precalculada

El control por retroalimentación es la estrategia de control más común en las industrias de procesos, ha logrado tal aceptación por su simplicidad, sin embargo, en algunos procesos de control por retroalimentación no proporciona la función de control que se requiere, para estos procesos se deben diseñar otros tipos de control.

El objetivo del control por acción precalculada es medir las perturbaciones y compensarlas antes de que la variable controlada se desvíe del punto de control. Un ejemplo concreto del control por acción precalculada se puede ilustrar de nuevo con el intercambiador de calor.

Supóngase que las perturbaciones más serias, son la temperatura de entrada y el flujo de proceso; para establecer el control por acción precalculada primero se deben medir estas dos perturbaciones y luego tomar una decisión sobre la manera de manejar el flujo de vapor para compensar los problemas. La figura 28 muestra esta estrategia de control; el control por acción precalculada decide como manejar el flujo de vapor para mantener la variable controlada en el punto de control en función de la temperatura de entrada y el flujo de proceso.

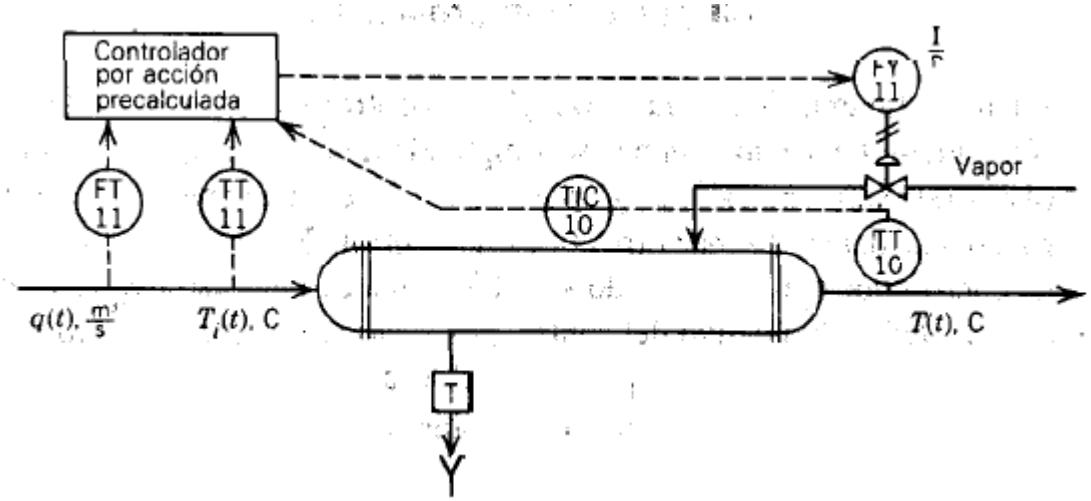
Figura 28. **Intercambiador de calor con acción precalculada**



Fuente: Smith y Corripio. Sistemas de control automático. p. 24

Anteriormente se mencionó que existen varios tipos de perturbaciones, el sistema de acción por acción precalculada que se muestra en la figura 30, sólo compensa a dos de ellas, si cualquier otra perturbación entra en el proceso no se compensará con esta estrategia y puede originarse una desviación permanente de la variable respecto al punto de control. Para evitar esta desviación, se debe añadir alguna retroalimentación de compensación al control por acción precalculada, esta estrategia combinada se ilustra en la figura 29. Ahora el control por acción precalculada compensa las perturbaciones más serias, la temperatura de entrada T (t) y el flujo de proceso q (t) mientras que el control por retroalimentación compensa todas las demás.

Figura 29. **Control por acción precalculada con compensación por retroalimentación**

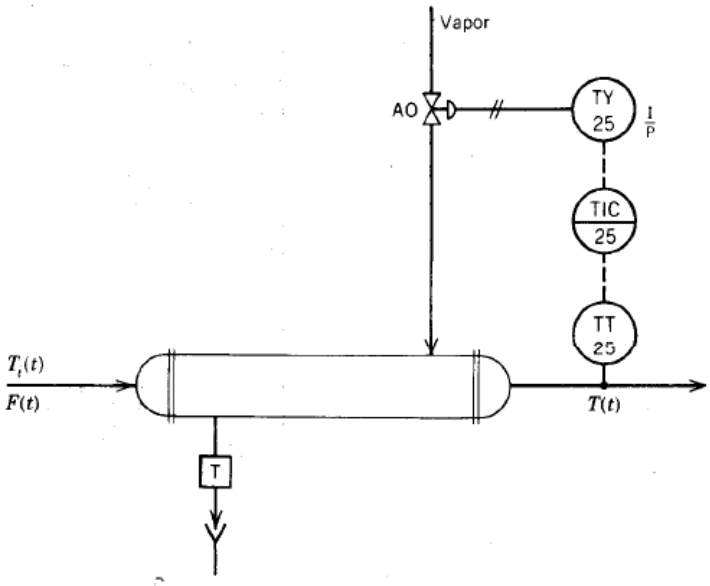


Fuente: Smith y Corripio. Sistemas de control automático. p. 24

1.3.1.3. Control en cascada

El control en cascada es una técnica de control muy común y útil en las industrias de proceso; en esta sección se presentarán sus principios mediante el intercambiador de calor tratado en explicaciones anteriores. En este sistema la temperatura con que sale el líquido que procesa se controla mediante la manipulación de la válvula de vapor. Se notará que no se manipula el flujo de vapor, éste depende de la posición de la válvula de vapor y de la caída de presión a través de la válvula; si se presenta una elevación de presión en la tubería de vapor, es decir, si la presión aumenta antes de la válvula, se cambia el flujo de vapor, esta perturbación puede controlarse, por medio del circuito de control que se ilustra en la figura 30, únicamente después de que la temperatura de proceso se desvía del punto de control.

Figura 30. **Circuito para el control de temperatura del intercambiador de calor**



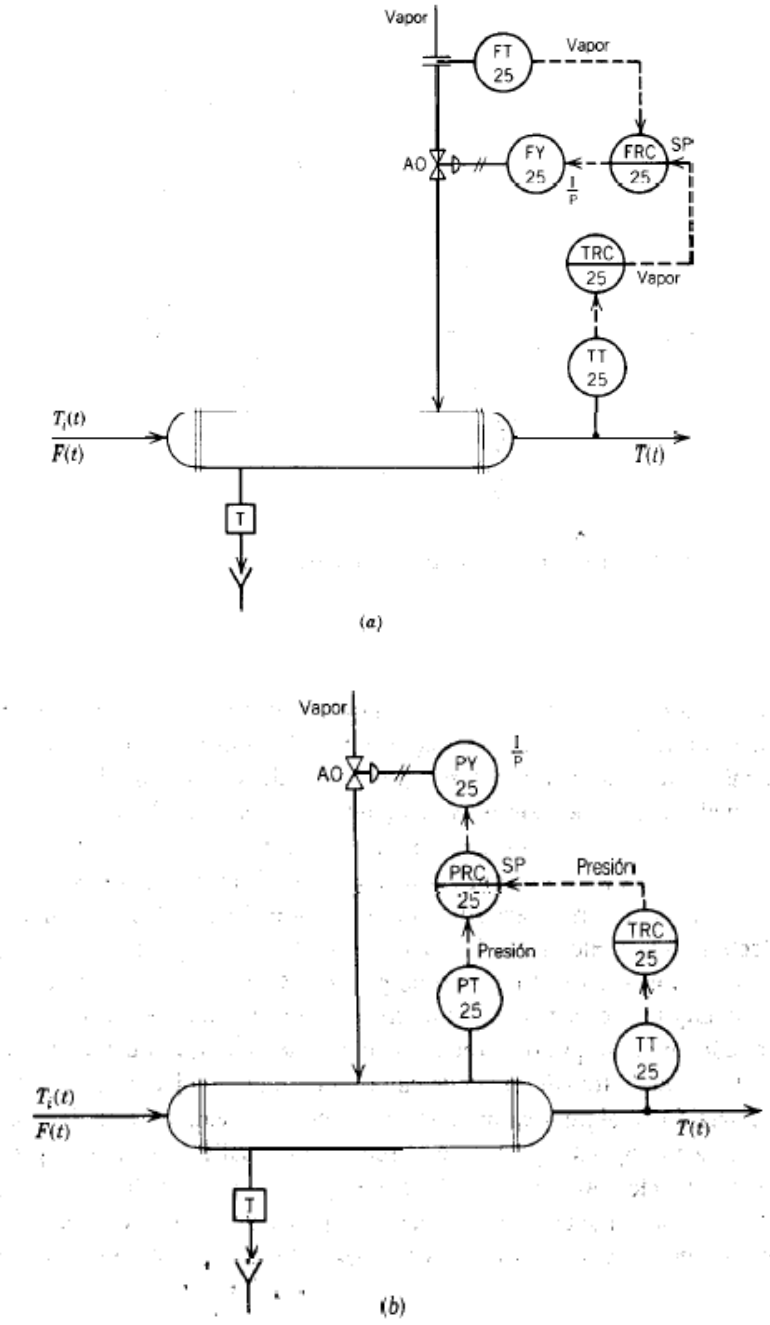
Fuente: Smith y Corripio. Sistemas de control automático. p. 445

A continuación se mostraran dos esquemas en cascada con que se puede controlar esta temperatura cuando los cambios en la presión del vapor son importantes. La figura 31a muestra un sistema en cascada al que se añadió un circuito de flujo, el punto de control del controlador de flujo se reajusta con el controlador de temperatura, ahora cualquier cambio en el flujo se compensa por el circuito de flujo. El significado físico de la señal que sale del controlador de temperatura es el flujo de vapor que se requiere para mantener la temperatura en el punto de control.

Con el esquema en cascada que se muestra en la figura 31b se logra el mismo control, pero ahora la variable secundaria es la presión de vapor en el casquillo del intercambiador; cualquier cambio en el flujo de vapor afecta rápidamente la presión en el casquillo y cualquier cambio de presión, se compensa entonces con el circuito de presión. Con el circuito de presión también se compensa cualquier perturbación en el contenido calorífico del vapor.

Antes de concluir con esta sección es importante mencionar algunas cosas importantes sobre la acción de los controladores en un sistema en cascada. En la figura 31a el controlador de flujo es de acción inversa, es decir, por los requerimientos del proceso y la acción de la válvula de control. El controlador de temperatura también es de acción inversa; esta decisión se toma con base a los requerimientos del proceso, esto es si se incrementa la temperatura de salida, entonces se requiere que el flujo de vapor decremente y por lo tanto se debe disminuir la salida del controlador de temperatura hacia el controlador de flujo. En síntesis un sistema en cascada es un sistema que cuenta con un controlador maestro y un controlador esclavo, el cual obedece al maestro de acuerdo al cambio que presenten las variables del proceso.

Figura 31. **Aplicación del esquema de control en cascada para intercambiador de calor**



Fuente: Smith y Corripio. Sistemas de control automático. p. 446

1.3.1.4. Control PID

El control PID es un mecanismo de control por retroalimentación que se utiliza en procesos industriales. Este control corrige el error entre un valor medido y el valor que se quiere obtener calculándolo y luego sacando una acción correctora que puede ajustar al proceso acorde. El algoritmo del control PID se da en tres parámetros distintos, el proporcional, el integral y el derivativo. El proporcional determina la reacción del error actual, el integral genera una corrección proporcional a la integral del error, lo cual nos asegura que aplicando un esfuerzo de control suficiente, el error de seguimiento se reduce a cero y el derivativo determina la reacción del tiempo en que el error se produce.

1.3.1.4.1. Control proporcional

La parte proporcional consiste en el producto de la señal de error y la constante proporcional como para lograr que el error en estado estacionario sea casi nulo, pero en la mayoría de los casos, éstos valores sólo serán óptimos en una determinada porción del rango de control, siendo distintos los valores óptimos para cada porción de control. Sin embargo existe un valor límite en la constante proporcional a partir del cual, en algunos casos, el sistema alcanza valores superiores a los deseados, este fenómeno es conocido como sobre oscilación.

La parte proporcional no considera el tiempo, por lo tanto, la mejor manera de solucionar el error permanente y hacer que el sistema contenga alguna componente que tenga en cuenta la variación del tiempo es incluyendo y configurando las acciones integrales y derivativas. La fórmula del control proporcional está dada por la expresión $P_{sal} = K_p * e(t)$ donde e (t) es el error en función del tiempo.

1.3.1.4.2. Control integral

El modo de control integral tiene como propósito disminuir y eliminar el error en estado estacionario, provocado por el modo proporcional. El control integral actúa cuando hay una desviación entre la variable y el punto de consigna, integrando esta desviación en el tiempo y sumándola a la acción proporcional. El error es integrado, lo cual tiene la función de promediarlo o sumarlo por un período de tiempo determinado para luego ser multiplicado por una constante I.

I representa la constante de integración, posteriormente la respuesta integral es adicionada al modo proporcional para formar el control P+I con el propósito de obtener una respuesta estable del sistema sin error estacionario. La ganancia total del lazo de control debe ser menor a 1 y así inducir una atenuación en la salida del controlador para conducir el proceso a la estabilidad. El control integral se utiliza para obviar el inconveniente del *offset* (desviación permanente de la variable con respecto al punto de consigna). La formula integral está dada por $I_{sal} = K_i \int_0^t e(t)dt.$

1.3.1.4.3. Control derivativo

La acción derivativa se manifiesta cuando hay un cambio en el valor absoluto del error, por lo tanto se puede decir que cuando el error es constante solamente actúan los modos proporcional e integral. La función de la acción derivativa es mantener el error al mínimo corrigiéndolo proporcionalmente con la misma velocidad que se produce y de esta manera es como se evita que el error se incremente.

Se deriva con respecto del tiempo, se multiplica por una constante D y luego se suma a las señales anteriores (P+I). Es importante adaptar la respuesta de control a los cambios en el sistema ya que una mayor derivativa corresponde a un cambio más rápido. El control derivativo se caracteriza por el tiempo de acción derivada en minutos de anticipo. La acción derivada es adecuada cuando hay retraso entre el movimiento de la válvula de control y su repercusión en la variable controlada.

Cuando el tiempo de la acción derivada es grande, hay inestabilidad en el proceso. Cuando el tiempo de la acción derivada es pequeño, la variable oscila demasiado con respecto al punto de consigna. Suele ser poco utilizada debido a la sensibilidad al ruido que manifiesta y a las complicaciones que ello conlleva. El tiempo óptimo de acción derivativa es el que retorna la variable controlada al punto de consigna con las mínimas oscilaciones. La formula derivativa está dada por $D_{sal} = K_d \frac{de}{dt}$.

1.3.1.4.4. Significados de las constantes

P constante de proporcionalidad: se puede definir como el valor de la ganancia del controlador o el porcentaje de banda proporcional y la acción que realiza esta constante en una aplicación sería, por ejemplo, cambiar la posición de una válvula proporcionalmente a la desviación de la variable respecto al punto consigna. La señal P, mueve la válvula siguiendo fielmente los cambios de la variable controlada multiplicados por la ganancia.

La I es la constante de integración o constante integral y nos indica la velocidad con la que se repite la acción proporcional. La constante D es la constante derivativa, hace presente la respuesta de la acción proporcional duplicándola, sin esperar a que el error se duplique. El valor indicado por la constante de derivación es el lapso de tiempo durante el cual se manifestará la acción proporcional correspondiente a dos veces el error y luego desaparecerá.

Tanto la acción integral, como la acción derivativa, afectan la ganancia dinámica del proceso. La acción integral sirve para reducir el error estacionario, por ejemplo corrige la posición de una válvula proporcionalmente a la velocidad de cambio de la variable controlada. La señal D, es la pendiente (tangente) de la curva descrita por la variable.

La salida de estos tres términos, el proporcional, el integral y el derivativo son sumados para calcular la salida del modo de control PID. Definiendo u(t) como la salida final del controlador, la forma final del algoritmo PID es $u(t) = MV(t) = K_p * e(t) + K_i * \int_0^t e(t)dt + K_d * \frac{de}{dt}$.

2. FUNCIONAMIENTO DE UN SISTEMA DE TROQUELACIÓN

En las siguientes páginas se expondrà todo lo referente a la troqueladora, desde que es troquelar, pasando por la explicación de las partes que componen dicha máquina herramienta, hasta llegar a algunas aplicaciones de las que forma parte.

2.1. ¿Qué es troquelar?

La palabra troquelar se deriva del griego *troque* que quiere decir corte y *lar* que quiere decir forma, de lo cual podemos decir que troquelar a grandes rasgos por sus orígenes semánticos significa cortar a su forma o bien cortar a la forma.

En términos más simples se denomina troquelar a la acción mecánica que se utiliza para realizar agujeros en chapas de metal, láminas de plástico, papel o cartón. Para realizar esta tarea, se utilizan desde simples mecanismos de accionamiento manual hasta sofisticadas prensas mecánicas de gran potencia.

Los elementos básicos de una troqueladora lo constituyen el troquel que tiene la forma y dimensiones del diseño que se quiere realizar y la matriz de corte por donde se inserta el troquel cuando es empujado de forma enérgica por la potencia que le proporciona la prensa mediante un accionamiento excéntrico que propicia un golpe seco y contundente sobre el material a troquelar produciendo un corte limpio del mismo.

Uno de los mecanismos de troquelado más simples y sencillos que existen, lo hemos tenido en nuestros hogares y lo hemos utilizado todos y cada uno de nosotros en los últimos 30 años es el perforador o más comúnmente llamado sacabocados.

Es importante hacer ver que existen varias técnicas de troquelado y que no necesariamente un troquelado implica la perforación del material, de hecho la técnica de resaltar una silueta o figura en una de las caras de un material delgado también se conoce con el nombre de troquelar. Por ejemplo las placas de los automóviles fueron hechas con una troqueladora y cómo podemos notar en ningún momento se ve perforada la hoja de lámina, simplemente se le resaltan, en una de sus caras, los números y las letras que corresponden.

La razón de que a este proceso también se le llamé troquelado reside en cómo se realiza la función, por medio de una prensa que aplica presión a un molde para hacer resaltar la figura, similar al proceso por medio del cual se realiza el troquelado que perfora el material, pero en este trabajo de graduación, siempre que se haga referencia al termino troquelar significará el proceso de perforación del material.

2.2. Funcionamiento de la troqueladora

Antes de exponer de lleno los detalles de cómo funciona la troqueladora y sus aplicaciones, es vital aclarar el concepto de máquina herramienta; ya que la troqueladora pertenece a este tipo de mecanismos.

2.2.1. Máquina herramienta

La máquina herramienta es un tipo de máquina que se utiliza para dar forma a materiales sólidos, principalmente metales. Su característica principal es su falta de movilidad, ya que son máquinas estacionarias. El modelado de la pieza se realiza por la eliminación de una parte del material, que se puede realizar por arranque de viruta, estampado, corte o electroerosión.

El término máquina herramienta se suele reservar para herramientas que utilizan una fuente de energía distinta del esfuerza humano, aunque también pueden ser impulsadas por personas si se instalan adecuadamente o cuando no hay otra fuente de energía. Muchos historiadores de la tecnología consideran que las autenticas máquinas herramientas nacieron cuando se eliminó la actuación directa del hombre en el proceso de dar forma o troquelar los distintos tipos de materiales.

La máquina herramienta puede utilizar una gran variedad de fuentes de energía. La energía humana y la animal son opciones posibles, aunque en la actualidad su uso es prácticamente nulo, como también lo es la energía obtenida a través del uso de ruedas hidráulicas. Sin embargo, el desarrollo real de la máquina herramienta comenzó tras la invención de la máquina de vapor, aunque hoy en día la mayoría funcionan con energía eléctrica.

Las máquinas herramienta pueden controlarse manualmente o por medio de control automático, las primeras máquinas utilizaban volantes para estabilizar su movimiento y poseían sistemas complejos de engranajes y palancas para controlar la máquina y las piezas en que trabajaba. Unos años más tarde se desarrollaron los sistemas de control numérico.

Las máquinas de control numérico utilizaban una serie de números perforados en una cinta de papel o tarjetas perforadas para controlar su movimiento. En los años 60, con el auge de las computadoras, estas fueron añadidas a sus sistemas de control para aumentar la flexibilidad del proceso. Tales máquinas se empezaron a llamar máquinas CNC o máquinas de control numérico por computadora. Las máquinas de control numérico y CNC pueden repetir secuencias una y otra vez con precisión y pueden producir piezas mucho más complejas que las producidas por el operario más experimentado.

2.2.1.1. Tipos de máquina herramienta

Por la forma de trabajar, las máquinas herramienta se pueden clasificar en tres tipos, las de desbaste o desbastadoras que dan forma a la pieza por arranque de viruta; este tipo de máquinas también se les conoce como del tipo convencional. Las del tipo prensa que dan forma a las piezas mediante el corte, el prensado o el estirado y finalmente las máquinas herramienta especiales, que dan forma a la pieza mediante técnicas diferentes como láser, electroerosión, ultrasonidos, plasma, etc.

2.2.1.1.1. Máquinas herramienta convencionales

Entre las máquinas herramienta convencionales más comunes se encuentran las siguientes:

El torno, es una de las máquinas más antiguas y trabaja mediante el arranque del material, realizando ésta acción apoyándose de herramientas cortantes y brocas. Para ello la pieza gira y mediante un carro en el que se sitúa la herramienta se va desgastando la pieza obteniendo partes cilíndricas y cónicas. Si se coloca una broca en la posición correspondiente, se pueden realizar agujeros.

Los taladros, destinados a la perforación, estas máquinas herramienta son, junto con los tornos, las máquinas más antiguas. En ellas el útil es el que gira y la pieza permanece fija en una mordaza o colocación. El útil suele ser, en los taladros, una broca que se encuentra debidamente afilada y que realiza el agujero correspondiente. También se pueden realizar otras operaciones como avellanar y escariar.

La fresadora, con la finalidad de la obtención de superficies lisas o de una forma concreta, son máquinas complejas en las cuales el útil es el que gira y la pieza permanece fija a una bancada móvil. El útil utilizado es la fresa, que suele ser redonda con diferentes filos cuya forma coincide con la forma que se quiere dar a la pieza. La pieza se coloca sólidamente fijada a un carro que la acerca a la fresa en las tres direcciones, esto es en los ejes X, Y y Z.

2.2.1.1.2. Máquinas herramienta de prensa

Este tipo de máquinas funcionan en una forma completamente distinta a las máquinas convencionales, ya que este tipo de mecanismos no realizan arranque de viruta, éstas se encargan de dar forma al material mediante el corte o cizalla, el golpe para el doblado y la presión. Se suelen utilizar troqueles y matrices como útiles. Los procesos son extremadamente rápidos y son máquinas con un alto índice de accidentes laborales. Dentro de este tipo de máquinas, como podrá suponer, se encuentra la troqueladora que es sobre la que se hablará más adelante.

2.2.1.1.3. Máquinas herramienta no convencionales

Dentro de estos mecanismos se encuentra la máquina de electroerosión, la cual desgasta el material mediante chispas eléctricas que van fundiendo partes minúsculas del mismo. Hay dos tipos de máquinas de electroerosión.

Las de arco de plasma, utilizan un chorro de gas a gran presión y temperatura para el corte del material, también tenemos las máquinas de láser, que mediante un potente y preciso rayo láser realizan cortes en el material vaporizándolo.

Y finalmente nos encontramos con las máquinas ultrasónicas, que funcionan haciendo vibrar el útil a una velocidad ultrasónica, regularmente por encima de los 20 KHz y utilizando un material abrasivo y agua se va realizando el mecanizado de la pieza por la fricción de las piezas abrasivas. Para trabajar estas máquinas se utilizan materiales muy duros como el vidrio y el diamante.

2.2.2. El proceso de troquelado

En este apartado se dará una explicación un tanto general, por llamarla de algún modo, de cómo se lleva a cabo el troquelado de un material cualquiera. Se dará una explicación de esta manera debido a que la finalidad de este trabajo de graduación no es el diseño de la máquina como tal, sino el diseño de su sistema de control; de hecho los diseños de las máquinas troqueladoras varían de acuerdo a su aplicación y eventualmente habrán máquinas a las cuales se adapte el sistema de control, como también otras a las cuales no sea funcional; pero la descripción que se dará a continuación se hará de acuerdo a la idea general de un sistema de troquelación encargado de perforar un material con una forma previamente determinada por el usuario.

Entonces como se ha mencionado anteriormente una troqueladora cuenta con dos partes principales las cuales son el troquel y la matriz, el troquel o macho es la herramienta encargada de realizar el corte del material. El corte limpio del material depende de la herramienta que se utiliza y de las características propias de la misma como por ejemplo el filo, el tratamiento de endurecimiento por el que pasó el material, la potencia del motor que controla la herramienta, etc.

La matriz es la parte a la cual se fija el material que va a ser perforado, la matriz se considera como la hembra del troquel, debido a que ella es la encargada de sostener firmemente el material para que el troquel pueda aplicar la fuerza necesaria para realizar un corte limpio y profesional sin que el material se doble o se levante de la superficie debido a la tendencia que tiene el material a adherirse al troquel.

Este comportamiento, del material, de adherirse al troquel tiene su origen en que cuando el material es perforado su elasticidad tiende a ceder y dejar pasar al troquel, pero cuando el troquel ha realizado el corte y se dirige de nuevo hacia arriba, la elasticidad del material hace que éste se contraiga y se adhiera a la herramienta que realizó la perforación.

La acción de perforar un material lleva envuelta un poco de teoría de materiales la cual se describirá a continuación de una forma sencilla y breve para que tengamos una mejor idea de que sucede cuando realizamos estos procedimientos. La acción ejercida entre el troquel y la matriz actúa como una fuerza de cizallamiento en el material a procesar, una vez que el troquel ha logrado perforar éste, dicho material se encuentra bajo los efectos de esfuerzos que rebasan su límite elástico, produciendo la ruptura o desgarramiento en ambas caras aproximadamente en el mismo lapso de tiempo y conforme el troquel penetra más y más se produce la separación del material completando el proceso.

En este momento ya se cuenta con una idea general de lo que es un sistema de troquelado, así que se procederá a unir todas las partes que se han explicado por separado y lograr integrar a un solo concepto, cómo se realiza el proceso de troquelar un material.

La idea del sistema de troquelación bajo la cual se fecundó la elaboración de este trabajo de graduación consiste en un sistema que sea capaz de realizar la perforación de un material con la forma deseada por el cliente. Una vez que el diseño del perforado ha sido definido será cargado a la máquina y la máquina, controlada desde una computadora, se encargará de enviar información hacia los actuadores de la máquina para realizar la perforación en el material sin que el operario intervenga, mayoritariamente dentro del proceso.

Para realizar esta acción se plantean dos escenarios, en el escenario uno suponga que se necesita realizar una perforación con la forma de un rectángulo con ciertas dimensiones, entonces el operario carga los datos de la forma a perforar el material en la PC, luego verifica que todo esté en su lugar para poder iniciar el proceso. Entonces inicia el proceso y en este primer escenario el troquel sólo podrá moverse hacia arriba y hacia abajo, es decir el únicamente se encargará de colocar el punto inicial de la perforación, es la matriz que controlada por medio de servomotores tendrá la capacidad de moverse en un espacio de dos dimensiones, ejes X y Y, para realizar la forma del rectángulo, mientras el troquel permanecerá fijo únicamente perforando el material.

Por el otro lado en el escenario dos el troquel es quien realiza todo el proceso, es decir que tiene libertad de movimiento en los tres ejes X, Y y Z ya que el iniciará el perforado descendiendo de su posición inicial hacia el material a perforar (eje Z) y además será él quien también por medio de servomotores y mecanismos adecuados tendrá la libertad de moverse en un espacio bidimensional (ejes X y Y) para realizar la forma del perforado previamente cargada en la PC, limitando la funcionalidad de la matriz únicamente a mantener firmemente sujeto el material mientras el troquel realiza toda la tarea.

2.3. Aplicaciones del sistema de troquelación

Como podrá imaginar las distintas aplicaciones para el sistema de troquelación, a lo largo de la industria son exageradamente variadas, además el término troqueladora ha quedado reservado para operaciones de grabado, pre-cortado o estampado de materiales. En la actualidad las máquinas que realizan acciones similares a la que se describió anteriormente se conocen con el nombre de cortadoras.

Al final de cuentas el nombre es lo de menos, lo importante mas allá que sea una troqueladora, una cortadora o cualquier otro nombre con que se le conozca, es el producto final, el perforado del material con una forma específica, y a continuación se hablará sobre las aplicaciones más conocidas para estos sistemas.

2.3.1. Cortadora por láser

Una cortadora por láser es una máquina que utiliza un láser para cortar materiales. La cortadora por láser funciona haciendo que el láser pase a través del material por uno de sus ejes, generalmente el eje perpendicular. Si un corte tiene lugar en una localización donde no existen bordes cercanos , se utiliza una técnica en la cual primero se realiza una pequeña perforación, conocida como *piercing*, antes de continuar con el corte. Durante el *piercing* un pulso de alta potencia del haz del láser se concentra en un punto y poco a poco se aumenta la intensidad del mismo hasta lograr perforar el material.

Al configurar una cortadora láser hay dos configuraciones para elegir, al configuración hibrida y el sistema de vuelo óptico. La diferencia entre estas configuraciones es la forma como se mueve el haz de láser sobre el material que está cortando. Una similitud entre ambas configuraciones es que todas utilizan los ejes X y Y como ejes del movimiento.

La configuración híbrida realiza el trabajo por medio de un trabajo en conjunto entre la cabecera del láser y la pieza de trabajo. La mesa de trabajo con la pieza se mueve en un eje, a menudo el eje X, y la cabecera con el láser se mueve en el eje opuesto, eje Y. Este trabajo de cooperación permite un trabajo limpio y exacto pero de un nivel un poco inferior al modo de vuelo óptico.

La configuración de vuelo óptico, tiene una cabeza de corte en movimiento con una pieza de trabajo estacionaria. Esta configuración normalmente no requiere de mecanismos de sujeción de la pieza y los cortes suelen ser bastante limpios. Además esta configuración es la más rápida ya que las características de velocidad y aceleración de los perfiles de movimiento son superiores en comparación con el modo hibrido. La figura 32 ilustrará la forma de una cortadora por láser.

Figura 32. **Cortadora por láser**

Fuente: http://sp.ymlaser.com/upload/2008/2/cma-6040l.jpg

2.3.2. Cortadora por chorro de agua

A pesar de su precio, existen numerosas ventajas del corte de agua frente al corte de láser. El corte por agua permite cortar de todo y hasta un espesor más ancho que el que el láser puede hacer, aunque este ultimo sea más rápido. El sistema de corte se realiza a traves de un chorro de agua de unos poco milímetros de diametro a presiones que superan los 4000 bares, mezclando el chorro de agua con una arena abrasiva.

La máquina está compuesta por una mesa que va cubierta con agua, una especie de piscina, y el agua actua como barrera del chorro. Para realizar el corte primero hay que realizar el dibujo en la computadora y desdea ahí se le da la orden a la máquina. Es decir se realiza primero el dibujo en autocad y despues se pasa al programa especifico que tiene la máquina.

El operario introduce la clase de pieza que se va a cortar, el espesor y el tipo de material. Este menú de piezas es el que da la dureza y los parametros de tiempo necesarios para realizar el corte. Según todos estos parametros, el tiempo empleado puede variar mucho desde 15 minutos hasta 40 horas y tambien de acuerdo a estos parametros se deduce el costo del corte.

Tambien es importante tener en cuenta el tipo de corte que se desea obtener, ya que puede ir desde el mas basto hasta el mas perfecto, independientemente del material que sea. Todo ello depende de la utilidad que se le quiere dar despues de cortada a la pieza.

Otra ventaja que tiene esta máquina es que a pesar de que esté muchas horas en funcionamiento no se calienta. La máquina puede cortar todo tipo de materiales metalicos y plasticos, asi como marmol, vidrio, etc.

Con ella se pueden hacer grandes trabajos pero tambien pequeños detalles. El proceso de corte no afecta a los materiales porque no los calienta, endurece ni deforma, ademas el trabajo con esta teconologia es muy limpio y eficiente.

El proceso inicia al acelerar la conduccion del agua por una boquilla dirigida a una velocidad de 100 metros por segundo, esto se logra a traves de un intensificador de presion, como se menciono al inicio, de hasta 4000 bares de alta tecnologia.

Para obtener cortes sobre materiales de alta resistencia, se mezcla el agua con abrasivos controlados, alcanzando cortes de hasta 15 cm de espesor en aceros, y mayore en materiales mas suaves con alta precision en diseños sofisticados, obteniendo contornos terminados y piezas de gran calidad, imposibles de lograr con herramientas tradicionales.

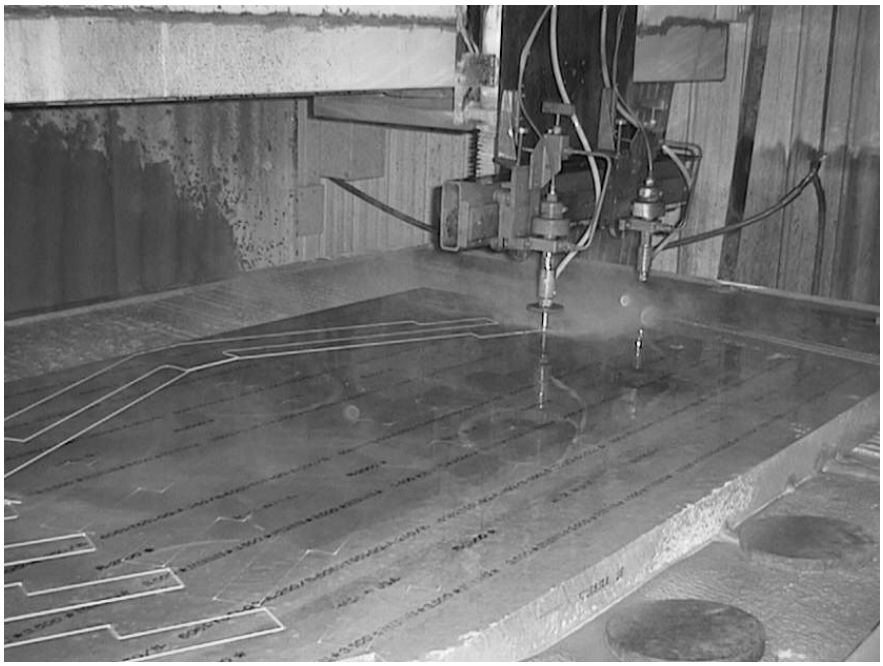
El corte con chorro de agua de alta presion puede cortar sin abrasivo materiales como espuma, papel, cartón, goma, plastico, fibra de vidrio, materiales para empaque, cuero para tapiceria automotriz y cualquier otro material blando no metalico. El chorro de agua con abrasivo corta todos los metales acero, acero inoxidable, acero de carbon, acero templado, aluminio, ligas de niquel, titanio, granito, vidrio, vidrio blindado, ceramica, azulejo y cualquier otro material con alto indice de dureza y de grandes espesores.

Ninguna otra máquina corta una variedad tan amplia de productos, corta materiales muy finos y delicados y tambien de gran espesor y dureza, ademas el *software* define los parametros de trabajo para todos los materiales, por lo que no es necesario la ejecucion y mudar de herramienta al cambiar el material a ser cortado.

Requiere apenas unos pocos minutos para el ajuste y fijacion del material a cortar. Permite aumentar la cantidad producida a traves del acomodo de varias placas del material al mismo tiempo y el corte de multiples piezas en una única etapa. Tiene la capacidad de utilizar multiples cabezas de corte aumentando la productividad y el chorro con abrasivo corta con el minimo de desperdici de material, optimizando el espacio entre las piezas a ser cortadas.

El chorro de agua corta en frio y por erosión, produciendo excelente calidad en el borde de los materiales, sin zonas afectadas por la induccion del calor o por el desgaste mecánico, no daña el medio ambiente, ya que no crea polvo y por lo tanto no contamina el aire, ademas no requiere del uso de soluciones dañinas para realizar su trabajo, la figura 33 nos muestra como lucen estas máquinas fabulosas.

Figura 33. **Cortadora por chorro de agua**



Fuente: http://www.npl.illinois.edu/ftp/G0/sms/pictures/waterjet/oct_cut2.jpg

Y así como éstas, existen otras muchas aplicaciones de los sistemas de troquelación, como por ejemplo la cortadora plasma, que es básicamente una cortadora láser, pero en lugar de utilizar la fuerza del haz de láser como herramienta de corte, lo realiza por medio de gas calentado a cientos de grados que funden el material y le dan la forma deseada.

También existen combinaciones de ambas máquinas como una cortadora de agua guiada por láser que se utiliza para realizar piezas muy pequeñas y con gran cantidad de detalles. Pero en fin las aplicaciones son muchísimas y no sería práctico ni cumpliría con los objetivos de este trabajo de graduación mencionarlas todas.

3. HERRAMIENTAS DE *SOFTWARE*

Acá se abordará la descripción de las herramientas de *software* a utilizar para llevar a cabo la simulación del mecanismo. Como primer paso se describirá cada *software* de una forma general para brindar al lector una idea general de las capacidades del programa.

Luego serán descritas en forma breve las diversas herramientas con que cuenta cada programa, entre ellas se incluirán las características y funciones por las cuales fueron elegidos estos programas para la realización de la simulación del sistema mecánico o prototipo virtual.

3.1. Herramientas de diseño asistido

Se denominan herramientas de diseño asistido a un conjunto de herramientas que permiten el diseño asistido por computadora. Es frecuente utilizar las siglas CAD, del inglés *Computer Aided Design*, para designar al conjunto de herramientas de *software* orientadas fundamentalmente, más no exclusivamente, al diseño (CAD), la fabricación (CAM) y el análisis (CAE) asistidos por computadora en los ámbitos científico e industrial.

Inicialmente estos programas se limitaban a pequeñas aplicaciones centradas en el dibujo técnico en dos dimensiones que venían a sustituir al tradicional tablero de dibujo, ya que ofrecía ventajas para la reproducción de los planos y reducía el tiempo de dibujo, permitiendo además utilizar elementos repetitivos y agilizar los cambios. Para hacer una analogía las ventajas de estos programas es equiparable a las ventajas presentadas por los primeros procesadores de texto frente a las máquinas de escribir.

Sus comienzos se vieron frenados por estar destinados a un grupo de usuarios muy reducido y requerían, además, de *hardware* muy potente por no hablar de la resistencia de muchos profesionales a adoptar estas tecnologías. Pero su potencial, el incremento de potencia del *hardware* y la importancia de las empresas que los usaban, entre las que ha destacado la industria de la automoción, permitieron que poco a poco estas herramientas alcanzaran las tres dimensiones y fueran incluso incluyendo curvas complejas, superficies y finalmente sólidos.

Hasta llegar a los complejos sistemas que permiten realizar todo el diseño de un automóvil o un avión, someterlos a pruebas de choque, temperaturas etc., realizar toda la infografía de *marketing*, realizar prototipos y por supuesto fabricarlos programando y controlando las máquinas que los fabrican y comprobando después los resultados obtenidos, todo ello en tiempos impensables hace veinte años.

Actualmente estos sistemas están conectados a los sistemas de gestión y producción de tal forma que ya desde la fase del diseño se puede conocer el costo del producto final, controlar los *stocks* de componentes y materiales para su fabricación, y en fin, todo lo que uno pueda imaginar.

Se ha pasado de tener una presentación de un plano en pantalla a tener un modelo virtual del que se puede obtener datos, montar en otros modelos, hacerlo adaptivo, imprimirlo y fabricarlo. El siguiente paso fueron los llamados sistemas expertos que permiten recoger reglas y normas de forma que el sistema guía al usuario en la toma de decisiones. Y ahora se persigue recoger el conocimiento y la experiencia del usuario y que el sistema aprenda, teniendo en cuenta estética, ingeniería, fabricación y calidad.

A ciencia cierta es imposible saber si lograran hacer que un programa de computadora sea tan inteligente como se pretende, pero ante la premisa que hace veinte años se pensaba que era imposible que un computador hiciera todo lo que hace hoy, es inevitable pensar que se está muy cerca de presenciar este paso final.

La evolución de estos sistemas ha permitido avances impresionantes en la industria en general, ya que gracias a estos programas hoy se benefician desde industrias tan avanzadas como la industria aeronáutica y espacial (NASA) hasta la industria de productos domésticos.

3.1.1. Diseño asistido por computadora CAD

Como se mencionó en el apartado anterior, el diseño asistido por computadora más conocido por sus siglas en ingles CAD (*Computer aided design*), es el uso de un amplio rango de herramientas computacionales que asisten a ingenieros, arquitectos y otros profesionales del diseño en sus respectivas actividades. También se puede llegar a encontrar denotado con las siglas CADD (*Computer Asisted Drawing and Design*), es decir dibujo y diseño asistido por computadora.

Estas herramientas se pueden dividir básicamente en programas de dibujo en dos dimensiones (2D) y modeladores en tres dimensiones (3D). Las herramientas de dibujo en 2D se basan en entidades geométricas vectoriales como puntos, líneas, arcos y polígonos, con las que se puede operar a través de una interfaz gráfica. Los modeladores en 3D añaden superficies y sólidos.

El usuario puede asociar a cada entidad una serie de propiedades como color, estilo de línea, nombre, definición, geometría, etc., que permiten manejar la información de forma lógica. Además pueden asociarse a las entidades, o conjunto de éstas, otro tipo de propiedades como material, peso, etc., que permiten enlazar a las herramientas CAD con los sistemas de gestión y producción.

De los modelos pueden obtenerse planos con cotas y anotaciones para generar la documentación técnica específica de cada proyecto. Los modeladores en 3D pueden, además, producir pre-visualizaciones foto-realista del producto, aunque a menudo se prefiere exportar los modelos a programas especializados en visualización y animación. La figura 34 muestra un sólido y la figura 35 muestra su respectivo plano generado en *solidworks software* de diseño del cual se hablará a continuación.

Figura 34. **Solido en 3D creado en *solidworks***

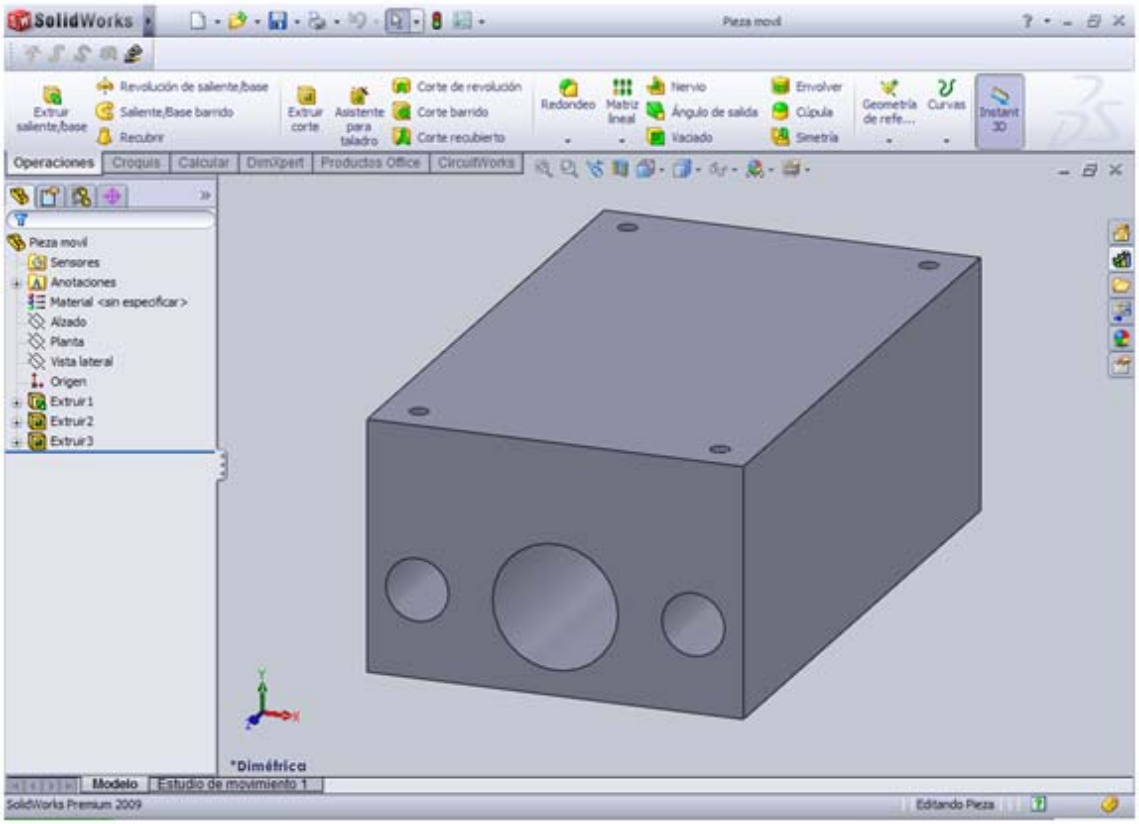
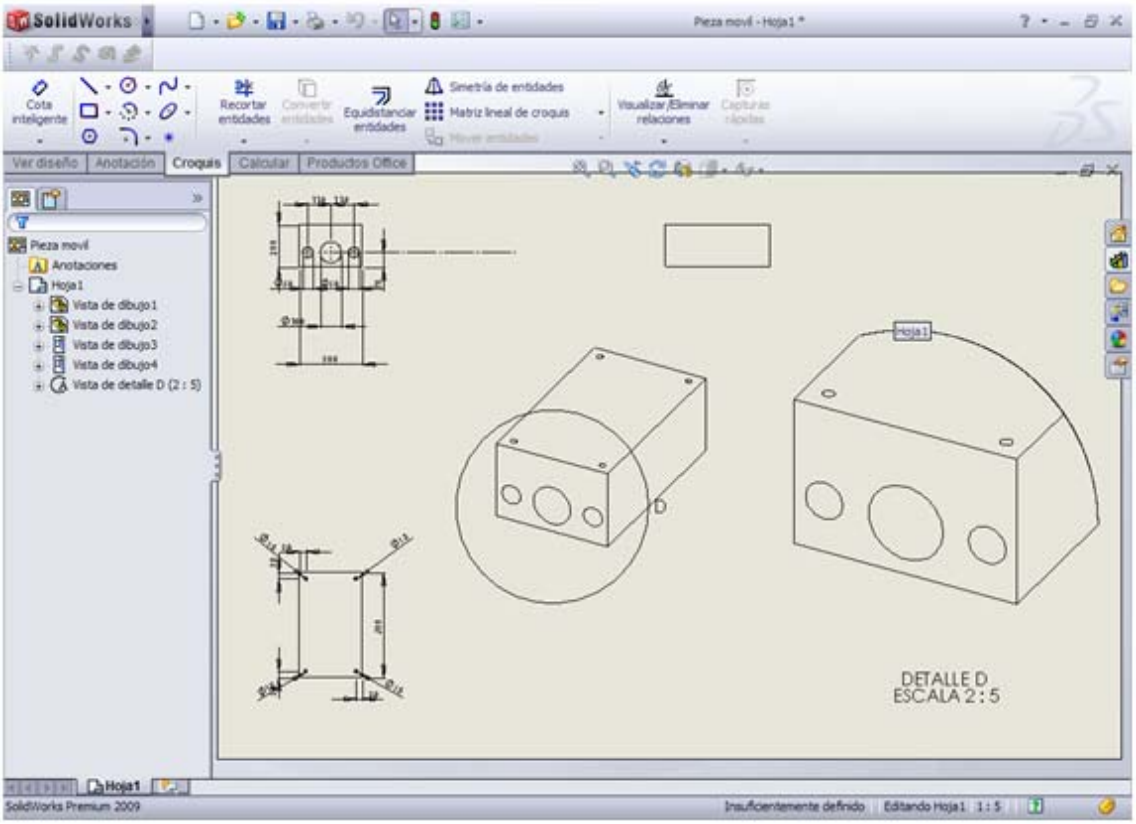


Figura 35. **Plano del solido 3D generado en *solidworks***



3.2. DS *Solidworks*

DS *Solidworks*, o bien, simplemente *Solidworks* es uno de los tantos *software* que existen en la actualidad para diseño asistido por computadora, pero que es lo primero que le viene a la mente cuando piensa en *Solidworks*, seguramente un programa para diseño CAD en 3D ¿no? Bueno, permítame decirle que está en lo cierto, ya que es debido a esta característica que es uno de los *software* de diseño más utilizado alrededor del mundo, pero es importante saber que *Solidworks* es mucho más.

Solidworks se esfuerza por ofrecer a los ingenieros, diseñadores y otros profesionales creativos las herramientas que precisan para diseñar los productos más increíbles del mundo. Además del diseño CAD, también ofrece una amplia gama de productos de simulación para ayudar a perfeccionar al máximo los diseños antes de que éstos sean enviados a producción.

Solidworks es un *software* ampliamente versátil y cuenta con una extensa gama de paquetes adicionales que permiten realizar diseños de una forma profesional y efectiva. Estos paquetes están clasificados en tres grandes grupos que son los paquetes de diseño mecánico, los paquetes para gestión de datos y paquetes de análisis. A continuación daré una breve descripción sobre las aplicaciones con las que cuentan los paquetes de diseño mecánico y paquete de análisis. Se omitirá el paquete de gestión de datos ya que estas funciones se desvían un poco de los objetivos planteados para este capítulo.

3.2.1. Paquetes de diseño mecánico

Este primer grupo de paquetes como su nombre lo indica, cuenta con funciones que ayudan en la realización de diseños mecánicos tanto en dos dimensiones como en tres dimensiones, además de funciones que proporciona un entorno gráfico bastante amigable y completamente compatible con las funciones más conocidas de la plataforma *Windows* entre otras. A continuación veremos en forma resumida las funciones más importantes de este paquete.

3.2.1.1. *SWIFT*

La función *SWIFT* (*Solidworks Intelligent Feature Technology*) le permitirá dedicar su tiempo a crear productos que funcionen correctamente, en lugar de intentar que el *software* funcione de la forma esperada. *SWIFT* permite automatizar tareas y técnicas, que de otro modo le llevarían mucho tiempo diagnosticar y solucionar problemas relativos al orden de las operaciones, relaciones de posición, relaciones de croquis y aplicación de dimensiones. Como resultado los usuarios de cualquier nivel verán incrementar su eficacia y su capacidad de innovación.

3.2.1.2. Funciones de ayuda para el principiante

Estas funciones son bastante especiales, ya que permiten empezar a utilizar el programa de forma inmediata. *Solidworks Command Manager* agrupa de forma lógica funciones similares para facilitar el acceso, de vital ayuda si es primera vez que utiliza el *software* y no conoce el lugar de cada función. Además, los tutoriales integrados proporcionan demostraciones de flujo de trabajo de aprendizaje con referencias visuales, las cuales ayudan a memorizar qué función realizan los distintos botones con que cuenta la interfaz gráfica y también ayudan a conocer los pasos lógicos para realizar las funciones en el menor tiempo posible brindándole la posibilidad de aprovechar al máximo su tiempo en el diseño en lugar de invertirlo en intuir el proceso para hacerlo.

3.2.1.3. Interfaz de usuario

Con la interfaz de usuario obtendrá un conjunto completo y totalmente personalizable de vistas en pantalla y de funciones de control a través del ratón. Esto permite reducir las iteraciones de diseño, minimizando la necesidad de utilizar cuadros de dialogo evitando en gran medida la confusión visual, logrando mantener al usuario concentrado en lo que hace en lugar de preocuparse por constantes mensajes que lo distraen.

3.2.1.4. Manejo de archivos DGW

Solidworks incluye herramientas de conversión de datos y documentación de ayuda para usuarios de *autoCAD*, por lo que podrá convertir los archivos DGW en modelos en tres dimensiones sin ninguna complicación. También podrá utilizar *DGWeditor*, incluido con el *software Solidworks*, para editar, manipular y mantener los archivos *autoCAD* DGW existentes en su formato nativo.

3.2.1.5. Modelado de piezas

Instant 3D proporciona el medio más rápido y sencillo de crear y modificar la geometría de las piezas creadas en tres dimensiones. Basta con hacer clic y arrastrar para crear operaciones y modificar su tamaño con precisión, incluso en unidades de sección. Durante la preparación del diseño de producción podrá utilizar *DimXpert* para crear precisos dibujos en 2D, o bien obtener datos sin dibujo, para asegurar que las piezas encajarán debidamente durante el proceso de fabricación.

3.2.1.6. Diseño de piezas soldadas

Esta herramienta permite realizar un croquis para el marco y seleccionar un perfil de soldadura. *Solidworks* generará de manera automática un diseño de soldadura en tres dimensiones. También brinda la opción de modificar, validar y reforzar el diseño y seguidamente, mejorarlo con piezas estándar que se podrán arrastrar y soltar desde la biblioteca de diseños o desde 3D *ContentCentral*. Luego podrá generar los respectivos dibujos de fabricación de gran precisión incluyendo longitudes de corte para cada segmento.

3.2.1.7. Modelado de ensamblajes

Solidworks proporciona las herramientas necesarias para dar en el clavo, en la fase de diseño y creación en pantalla, con lo que ahorra tiempo y costos asociados con la creación de prototipos físicos o la repetición de la fabricación. Podrá relacionar la posición de distintos componentes seleccionando bordes, superficies, curvas y vértices independientes, crear relaciones mecánicas entre componentes, realizar pruebas de interferencia, colisión y alineación de orificios o vincular el movimiento de engranajes y poleas. También podrá ensamblar de forma automática los cierres, automatizar el ensamblaje de los componentes más utilizados y utilizar *MateXpert* para solucionar conflictos tales como la restricción de un componente.

3.2.1.8. Simulación de movimiento de ensamblajes

Esta herramienta permite, con un simple clic y arrastrar los distintos componentes, realizar pruebas de movimiento y colisión del ensamblaje. Además podrá simular el movimiento físico de piezas relacionadas, contactos y muelles, o el efecto de la gravedad.

Solidworks motion también proporciona mediciones precisas de velocidad, aceleración y fuerza a las que se someten los componentes como consecuencia del movimiento. De este modo, tendrá una visión realista de sus decisiones de diseño antes de elaborar el prototipo del producto.

3.2.2. Paquetes de simulación

Estos paquetes cumplen una función vital dentro del complicado diagrama de flujo del diseño de prototipos virtuales ya que de ellos depende que el producto sea sometido a pruebas de movimiento, fuerzas y otros efectos en ambientes y/o escenarios idénticos o casi idénticos a los que se presentan la realidad, entregando un alto índice de confiabilidad en el producto final.

3.2.2.1. Validación de tolerancia al apilado

TolAnalist analiza totalmente un diseño en función del orden y la forma en que se han ensamblado las piezas, así como las dimensiones y tolerancias aplicadas con *DimXpert*. Determine si el diseño cumple con los requisitos de ensamblaje y funcionalidad sin la necesidad de efectuar cálculos tediosos manualmente en los que es fácil equivocarse.

3.2.2.2. Simulación de ensamblajes

Antes de que se cree confusión alguna, esta herramienta está relacionada con la mencionada en el paquete de diseño mecánico, mas no es la misma; con la aclaración hecha, se hablará sobre los beneficios de esta herramienta.

Estudia las interacciones de los distintos componentes de un ensamblaje en pantalla antes de incurrir en los costos de los prototipos físicos. Estudia también de forma precisa las cargas estáticas y dinámicas para determinar el rendimiento del diseño en condiciones de tensión, deformación o desplazamiento. Además cuenta con una escala multicolor que le indica los puntos donde existe mayor o menor esfuerzo según el tipo de simulación que realice con el ensamblaje.

3.2.2.3. Simulación de mecanismos

Aplica una gran variedad de modelos basados en la física para simular las condiciones de funcionamiento reales de los diseños. Compruebe choques entre piezas, obtenga datos numéricos y gráficos de resultados así como animaciones de las pruebas realizadas.

3.2.2.4. Predicción de errores del producto

Ahorra tiempo y costos de fabricación de prototipos y crea productos más seguros y durables. Predice los umbrales de fallos estructurales debido exceso de tensión, sobrecalentamiento, pandeo o fatiga. Evalúa los errores tanto de materiales comunes como de materiales avanzados como plásticos, elastómeros y compuestos a capas.

La predicción de pandeo, aplica de manera virtual distintas fuerzas, presione, gravedad y fuerzas centrifugas a los diseños, permitiendo monitorear el comportamiento de los diversos materiales.

Las funciones de análisis térmico permiten estudiar con facilidad los efectos de la temperatura en el diseño por medio de la simulación de condiciones térmicas externas, flujo de fluidos, interacciones térmico-estructurales y los efectos de la radiación en aplicaciones a altas temperaturas.

3.2.2.5. Dinámica no lineal

Realiza pruebas a profundidad de los diseños con una amplia gama de capacidades avanzadas entre las que se incluyen el análisis de desplazamiento no lineal, análisis de pandeo no lineal, y eventos *snap-trough*, análisis de materiales no lineales, optimización de diseños de materiales hiperelásticos, análisis elástico-plástico de cesión y posterior a la tensión, efectos de fluencia y cambios en materiales relacionados con la temperatura y análisis de respuesta dinámica de piezas y ensamblajes.

3.2.2.6. Simulación de flujo de fluidos

Estudia el flujo de líquidos, que incluye todos los líquidos desde los newtoneanos hasta los no newtoneanos como el dentífrico, la sangre o el cemento, también simula el flujo de gases en el interior de los diseños y alrededor de éstos.

Examina el rendimiento de sistemas de refrigeración electrónicos, válvulas y reguladores, sistemas de administración de medicamentos, maquinaria giratoria y objetos móviles.

3.2.3. Configuraciones para *Solidworks*

Solidworks cuenta con distintas configuraciones las cuales incluyen más o menos paquetes, brindando al cliente la posibilidad de adaptar la funcionalidad del *software* de acuerdo a sus necesidades. Las distintas configuraciones incluyen desde las funciones básicas del programa hasta funciones especializadas para aplicaciones puntuales, dando vía libre para que todo mundo tenga la posibilidad de utilizar *Solidworks* sin importar si es una persona con una fuente de recursos limitados o una empresa multinacional quien adquiere el programa. Estas configuraciones se dividen en tres las cuales serán mencionadas a continuación.

3.2.3.1. *Solidworks* estándar

Ofrece funciones para la modelación de piezas complejas, superficies, laminado, facilita el diseño de grandes ensamblajes y la creación automática de planos de ingeniería. Permite el análisis de esfuerzos y deformaciones en documentos de piezas.

Dentro de sus herramientas cuenta con barras de herramientas 2D y 3D que ayudan a convertir un dibujo de dos dimensiones en una pieza de tres dimensiones. El dibujo en 2D puede ser importado directamente desde un archivo de otra aplicación como archivos DGW y con la ayuda de estas herramientas puede ser convertido fácilmente en un modelo sólido en 3D.

Para modelos con componentes electrónicos pueden importar el archivo del circuito impreso y crear modelos sólidos de los circuitos impresos y sus componentes. El modelo es una sola pieza con la placa y cada componente como operaciones de extrusión.

Incluye una gran variedad de traductores CAD y también el usuario puede hacer rápidamente *layouts* de mecanismos en 2D. Estos *layouts* pueden ser manipulados dinámicamente para asegurarse de que el ajuste y las funciones son correctos antes de crear los ensamblajes en 3D. Usando los bloques en bosquejos brinda la oportunidad de crear complicados mecanismos en dos dimensiones y con un par de clics convertirlos automáticamente a componentes en tres dimensiones, también incluye simulaciones en dos dimensiones y librería de partes que es un documento que contiene partes o características que se usan con frecuencia.

Con *solidworks* estándar es posible agregar movimiento a los componentes de un ensamblaje para así detectar fallas por colisiones inesperadas o bien ajustar las medidas de su modelo para un funcionamiento óptimo del mecanismo. La simulación física le permite simular los efectos de motores resortes y la gravedad sobre sus ensamblajes.

También posee una herramienta, que entre otras cosas, tiene la capacidad para detectar interferencias entre componentes, visualizar el volumen de la interferencia como un volumen sombreado, seleccionar para omitir las interferencias que se desean excluir, como por ejemplo taladros de ajuste forzado, las interferencias de cierre roscado, etc.

Solidworks animator es otra herramienta de la configuración estándar que permite crear de forma fácil e intuitiva la animación de partes y ensamblajes en formato de video AVI. Una vez guardadas en formato AVI, esas animaciones pueden visualizarse por cualquier usuario que tenga *Windows* sin necesidad de contar con algún *software* CAD, facilitando la presentación de prototipos ante posibles clientes.

Además, brinda la posibilidad de crear una lista de materiales con el número de elementos, cantidades, números de piezas y propiedades personalizadas de los dibujos de ensamblajes. Estas listas de materiales llamadas BOM se pueden colocar, mover y editar.

Otra función interesante es la capacidad de obtener y mostrar información de vistas (lateral, frontal, superior, vistas en sección, etc.) listado de materiales, dimensiones y exportarlos a formatos DGW, DXF, PDF, BMP, JPEG.

Es posible, también, crear superficies complejas utilizando herramientas intuitivas y fáciles de usar para poder modelar piezas con características de forma especial. Mejora la estética ergonomía de los diseños generando formas avanzadas en su diseño por medio de la manipulación de las superficies de una forma interactiva. Además puede crear cavidades hembra-macho a partir del modelo en 3D. Las herramientas de moldes incluyen superficies de llenado de huecos y de particionado, creación automática de cavidades, revisión de espesores y análisis de ángulos de salida.

SWIFT, fue mencionado dentro de los paquetes de diseño mecánico y esta herramienta cuenta con gran cantidad de funciones que son de mucha ayuda para el diseño entre ellas tenemos la herramienta *MateXpert*, la cual ayuda a comprender errores y alertas de partes compañeras, proporcionando unos indicadores rojos y amarillos en la parte superior de la pantalla, los cuales aparecen directamente en la ventana de gráficos y claramente indican la problemática. Por lo tanto el problema se podrá resolver fácilmente removiendo las zonas en las que aparezcan los indicadores.

DimXpert es una herramienta del *SWIFT* que configura automáticamente las cotas y tolerancias geométricas en los componentes, ofreciendo información necesaria para cada equipo de diseño, por ejemplo, brinda comentarios visuales sobre si el modelo se ha descrito adecuadamente y si está listo para su fabricación. Así mismo, crea automáticamente vistas, cotas y tolerancias en los dibujos en 2D para completar la documentación del diseño.

AssemblyXpert analiza el rendimiento de los ensamblajes y sugiere posibles acciones para mejorarlos. Esto resulta útil al trabajar con ensamblajes grandes y complejos. En algunos casos puede optar por implementar los cambios sugeridos automáticamente.

Y finalmente se hablará un poco sobre las herramientas de análisis como *FlowXpress*, que permite analizar flujos de toda clase de líquidos y está incluido en todas las configuraciones de *Solidworks.* También está *DFMXpress* que es una herramienta de validación de los diseños y *COSMOSXpress* que ofrece una idea de cómo funcionará la pieza en un entorno real.

COSMOSXpress realiza el análisis de esfuerzos y deformaciones de las piezas paso por paso. Además es una herramienta de análisis estructural gran utilidad en la fase inicial del ciclo de diseño ya que tiene la capacidad de detectar cualquier error antes de seguir adelante.

3.2.3.2. *Solidworks* profesional

Solidworks profesional o *Solidworks office* profesional como también se le conoce a esta configuración está enfocado a incrementar la productividad de su empresa, acortar los tiempos de diseño y reducir los costos generados por prototipos. Con esta configuración podrá generar animaciones e imágenes renderizadas para tener una mejor presentación de su producto final.

Solidworks profesional cuenta con todas las características de la configuración estándar, es decir, ofrece modelación de piezas complejas, superficies, laminados, simulaciones y todo lo mencionado en el apartado anterior y además incluye herramientas de productividad y presentación, cuyas características más importantes serán descritas a continuación.

Se empezará con las herramientas de presentación, entre las cuales tenemos *eDrawing professional*, que es un *software* estándar para visualización de archivos *CAD* en 3D. Dicho programa es compatible con los archivos de programas como *autoCAD*, *Inventor*, *Pro/Engineer*, *Catia V5* y *Solid Edge* y al visualizar los modelos 3D permite marcar, agregar notas y medir de una forma bastante sencilla.

PhotoWorks es una herramienta que nos brinda la posibilidad de crear imágenes foto-realistas de los diseños en 3D, para producir presentaciones más atractivas y convincentes por medio de efectos visuales sofisticados con una biblioteca completa de materiales, texturas, luces y sombras y otros elementos que pueden modificarse, proporcionando una idea exacta del aspecto del diseño. .

Existen otras herramientas de presentación con las que cuenta *Solidworks* profesional, pero no serán mencionadas ya que lo único que hacen es brindar características extras a las herramientas mencionadas anteriormente por lo que sería un tanto redundante hablar de ellas, por lo que se hablará sobre las herramientas de productividad.

Una de las herramientas de productividad es *Solidworks Toolbox*, una librería de partes estándar para ahorrar tiempo, utilizando la "tecnología de partes inteligentes" que permite seleccionar de forma automática la tornillería adecuada de acuerdo al diseño. La librería incluye componentes de rodamientos, pernos, levas, engranes, tornillos, tuercas, arandelas, pasadores, poleas y anillos de retención.

Solidworks Utilities es otra de las herramientas de productividad que puede encontrar en la configuración profesional, este *software* permite trabajar de forma eficiente en cualquier proyecto que requiera múltiples cambios y revisiones, al usar *Solidworks utilities* se pueden encontrar las diferencias entre dos versiones de la misma pieza o localizar, modificar y suprimir características de un modelo.

Design Checker, Solidworks desing checker es una herramienta que verifica los elementos de diseño, como por ejemplo normas de acotación, fuentes, materiales y croquis para asegurar que el documento de *Solidworks* cumple con criterios de diseño predefinidos. El usuario configura los requisitos para realizar la evaluación y a continuación *design checker* evalúa el documento.

El administrador de tareas de *Solidworks* le permite configurar tareas que se ejecutaran en el futuro. Por ejemplo, si necesita imprimir todos los planos de un proyecto en particular, puede programar el administrador de tareas para que lo haga automáticamente indicando la hora y la fecha requerida. Puede programar que se realice una tarea una sola vez o bien de manera diaria, semanal o mensual.

Como se puede observar *Solidworks* profesional está orientado a convertir un sinfín de tareas que son tediosas en tareas simples que por medio de entornos gráficos guiados se vuelven bastante intuitivas y permiten optimizar el tiempo para aprovecharlo en tareas propias del diseño, además ofrece herramientas para mejorar la presentación de los prototipos y también cuenta con herramientas propias de la oficina que permiten una mejor comunicación entre los distintos departamentos de una empresa ya que desde *Solidworks* puede crear archivos con formatos compatibles con cualquier computadora con el sistema operativo *Windows* instalado en ella, evitando la necesidad de que las personas que no conocen o no tienen relación con *software* de diseño tengan que instalarlo para poder realizar su trabajo.

3.2.3.3. *Solidworks premium*

Esta es la configuración más completa para *Solidworks,* está enfocada a soluciones para empresas que necesitan realizar el diseño de productos, analizar y validar proyectos, mejorar las presentaciones del producto, administrar los datos del producto de una manera efectiva y facilitar y hasta automatizar la etapa de diseño.

Solidworks premium incluye *Solidworks* estándar y todas sus herramientas, también incluye todo lo mencionado unos párrafos atrás referente a las herramientas de diseño y presentación de *Solidworks* profesional y además tiene nuevas características como *COSMOSWorks Designer, COSMOSMotion, Solidworks routing y circuit works* entre otros de las cuales se dará una breve descripción a continuación.

COSMOSWorks designer es una herramienta para mejorar la calidad del producto identificando las aéreas que son más susceptibles a error y que son más débiles. Además podrá reducir los costos recortando el exceso de material y minimizando la necesidad de realizar prototipos físicos.

COSMOSWorks designer proporciona potentes herramientas para estudiar y optimizar los ensamblajes. Puede efectuar análisis directamente sobre todo el ensamblaje con las condiciones adecuadas o bien analizar sólo una parte del ensamblaje o sub-ensamblaje.

Esta herramienta también incluye varios tipos de cargas y restricciones para representar situaciones de la vida real. Todas las cargas y restricciones están asociadas con la geometría y se actualizan de forma automática con los cambios realizados en el diseño. Además cuenta con una serie de herramientas de automatización para simplificar el proceso de análisis y ayudar a trabajar de forma más eficaz, por ejemplo puede realizar varios estudios sobre el mismo proyecto con la tecnología "*drag and drop"* de *Windows.*

COSMOSWorks designer ofrece una gama de herramientas de visualización de resultados que le permiten obtener varios conocimientos del rendimiento de los modelos, de tal manera que no tendrá problemas a la hora de interpretar si el modelo es apto para realizar sus objetivos o no lo es.

Y finalmente *COSMOSWorks designer* permite colaborar y compartir los resultados de análisis de forma eficaz entre las personas relacionadas con el proceso de desarrollo del producto, puede ser por medio de un reporte *HTML* que se genera automáticamente al finalizar el estudio, o bien un video del producto finalizado en formato *AVI.*

COSMOSWorks profesional, es una aplicación que ofrece una serie de poderosas herramientas para ayudar a los ingenieros que se familiarizan con los conceptos de validación del diseño a realizar pruebas virtuales y análisis de partes y ensamblajes. Los ingenieros que necesitan capacidades más específicas de análisis de diseño pueden utilizar *COSMOSWorks* profesional para predecir el comportamiento físico prácticamente de cada parte bajo cada condición de carga a que será sometida.

COSMOSWorks profesional, aparte de contar con todas las ventajas mencionadas anteriormente de *COSMOSWorks designer*, provee de otras herramientas para diversos tipos de análisis como por ejemplo evaluaciones de transferencia de calor en estado estacionario y la transferencia de calor en estado transitorio con condiciones variables en el tiempo, con esta herramienta podrá evaluar las variaciones de temperatura que se producen en las piezas mecánicas y estructuras; las cuales pueden afectar notablemente el rendimiento del producto.

También tendrá la capacidad de estudiar los esfuerzos, la velocidad y las aceleraciones cuando los objetos caen desde diferentes alturas y orientaciones, simula pruebas de caídas virtuales sobre diferentes superficies de pavimento y descubra, sí el producto diseñado sufriera una caída sobrevivirá o no.

Además contará con la capacidad de realizar predicciones de la vida de los ensamblajes con componentes que tienen diferentes propiedades de materiales y características de fallo, estudie los efectos de las condiciones de funcionamiento de carga cíclica y fatiga. Vea los efectos de la fatiga en el ciclo de vida global de la pieza o ensamblaje para descubrir cuánto durará y que cambios en el diseño ampliaran su vida útil.

Y así como los estudios y análisis mencionados anteriormente, esta aplicación cuenta con otro montón de pruebas a las que pueden ser sometidas las piezas o ensambles para poder realizar proyecciones bastante precisas de la calidad y durabilidad del producto final.

COSMOSMotion es un robusto *software* de simulación y análisis cinemático y dinámico de mecanismos integrado a *Solidworks* y *COSMOSWorks* que permiten asegurar el correcto funcionamiento de un diseño antes de su construcción.

COSMOSMotion permite crear un modelo virtual de su sistema mecánico y verificar el correcto funcionamiento del mismo antes de su construcción. Esto significa una reducción importante en el número de prototipos físicos a construir y acelera el ciclo de desarrollo del producto.

Para utilizar este *software* primero se definen las restricciones de contactos, fuerzas, actuadores, desplazamientos, velocidades, aceleraciones, tiempo y movimiento. Seguidamente se calcula el movimiento del mecanismo para posteriormente revisar el movimiento del mecanismo mediante animaciones, gráficos y curvas X-Y, verificación de interferencias, para pasar a la etapa de resultados.

Dentro de los resultados podrá determinar motores y actuadores, calcular el gasto de potencia, definir uniones, entender el funcionamiento de levas y engranajes, dimensionar muelles y amortiguadores, determinar el contacto entre piezas del mecanismo, y finalmente obtener las fuerzas de inercia y reacciones entre los componentes que serán cargas listas para introducir automáticamente en el análisis por elementos, que se puede realizar con la herramienta *COSMOSWorks*, para cada una de las distintas piezas que forman el ensamblaje o bien el mecanismo.

Solidworks routing puede ahorrar mucho tiempo en el diseño de maquinaria y equipos que contengan subsistemas hidráulicos o neumáticos. *Solidworks routing* incluye una biblioteca de accesorios ya preparados que reduce aún más el tiempo de diseño, además de un filtro de selección de componentes que le permite cumplir con los estándares de una forma más sencilla.

También ofrece la posibilidad de generar documentación de mangueras de cables, lo cual optimiza la productividad en el diseño de subsistemas eléctricos. Podrá agregar recorridos de tubos, cañerías, y cables eléctricos en las primeras etapas del proceso de diseño.

Para los diseñadores de sistemas eléctricos y electrónicos, esta aplicación brinda herramientas adicionales que permitirán ahorrar tiempo en la generación de documentos de fabricación de mangueras y cables.

Solidworks premium cuenta con *Circuit Works* que es un traductor electrónico de gran alcance del CAD (ECAD) que permite crear modelos exactos en 3D de las tarjetas de circuitos en *solidworks,* podrá diseñar mejores productos compartiendo datos entre los equipos de diseño electrónico, eléctrico y mecánico, y además tendrá la opción de manipular una gran variedad de archivos de los distintos sistemas de ECAD.

Entonces como puede observar, *Solidworks premium* es una versión completísima de herramientas integradas de la forma adecuada para facilitar los pasos que se llevan a cabo para diseñar un producto cualquiera, sus herramientas y aplicaciones dan el lujo, al usuario, de preocuparse y desgastarse casi exclusivamente en pensar un diseño, ya que comprobar si el diseño cumple o no con las especificaciones y objetivos planteados es tarea de *Solidworks.*

Cabe resaltar que desde que *Solidworks* es un *software* de un altísimo nivel y debido a la enorme cantidad de estudios, pruebas y herramientas con que dispone es casi imposible, además de poco factible, que una persona sepa utilizarlo todo, por lo que otra característica importante de este *software* es que es multidisciplinario, ya que envuelve campos como la mecánica, electrónica, termodinámica, dinámica de fluidos, etc. y por tal razón puede ser utilizado por las diversas disciplinas de la ingeniería. No muy distinto es el caso de *LabVIEW*, que será el próximo *software* sobre el cual se hablará.

3.3. *National Instruments LabVIEW*

National Instruments LabVIEW, NI *LabVIEW* o simplemente *LabVIEW,* es un *software* industrial que utiliza un lenguaje de programación gráfico ampliamente adoptado por la industria, educación y laboratorios de investigación como *software* estándar para adquisición de datos y el control de instrumentos. *LabVIEW* es un poderoso y flexible *software* multiplataforma para sistemas de instrumentación y análisis. *LabVIEW* puede ser ejecutado en *Windows, MAC OS* X y *Linux*, también puede correr en *PDAs*, en plataformas de tiempo real e incluso programas embebidos en chips de FPGA y microprocesadores de 32 *bits*. Crear un programa propio, o instrumento virtual como se le conoce, es muy simple. La interfaz de usuario intuitiva de *LabVIEW* hace de escribir y utilizar programas algo divertido.

LabVIEW parte de la naturaleza secuencial de la programación tradicional y su principal característica es su entorno de programación gráfico fácil de usar, que incluye todas las herramientas necesarias para la adquisición de datos (*DAQ*), análisis de datos y presentación de resultados. Con su lenguaje de programación gráfico, también conocido como lenguaje G, las aplicaciones se programan usando un diagrama de bloques gráfico que compila todo a lenguaje de máquina. Ideal para un sinfín de aplicaciones en las ramas de la ciencia y la ingeniería, *LabVIEW* ayuda a resolver varios problemas en una fracción de tiempo y sin la molestia de hacerlo programando en código convencional.

3.3.1. ¿Qué es exactamente *LabVIEW?*

LabVIEW es un ambiente de programación en el cual las aplicaciones son creadas utilizando notación gráfica, es decir, conectando nodos funcionales por medio de cables a través de los cuales fluyen los datos; bajo esta consideración, difiere de lenguajes de programación tradicionales como C++ o Java, en los cuales se programa con texto. Sin embargo *LabVIEW* es mucho más que solo un lenguaje de programación. Es un programa interactivo para sistemas de desarrollo y ejecución, diseñado por personas como científicos e ingenieros, para quienes programar es su trabajo.

LabVIEW puede crear programas que corren o que se ejecutan en plataformas como *Microsoft Pocket PC*, *Microsoft Windows*, *Palm OS*, y una gran variedad de sistemas embebidos, incluyendo *Field Programmable Gate Arrays* (FPGA), Procesadores digitales de señales (DSP) y microprocesadores.

Utilizando el poderoso lenguaje de programación gráfica, que muchos usuarios de *LabVIEW* llaman lenguaje G, programas que tomaban semanas para ser escritos utilizando los lenguajes de programación convencionales pueden ser completados en horas utilizando *LabVIEW*, ya que está diseñado específicamente para realizar mediciones, analizar la información y presentar los resultados al usuario. Y debido a que *LabVIEW* cuenta con una interfaz de usuario, es fácil de programar, resulta una herramienta ideal para realizar simulaciones, programación en general e incluso enseñar los conceptos básicos de programación.

LabVIEW también ofrece mayor flexibilidad que los instrumentos estándar de los laboratorios ya que está orientado a la programación, de manera que el usuario, no el fabricante del instrumento, define la funcionalidad del instrumento. Los dispositivos periféricos de un computador, así como *LabVIEW,* se convierten en un sistema de instrumentación virtual con las características necesarias para cumplir con las tareas requeridas. Utilizando *LabVIEW*, se puede crear exactamente el tipo de instrumento virtual que se necesite, cuando lo necesiten y a una fracción del costo de instrumentos tradicionales; y si las necesidades cambian sin ningún problema se puede modificar el instrumento virtual para adecuarse a las nuevas necesidades.

LabVIEW trata de hacer de la programación una situación tan libre de complicaciones como sea posible. Cuenta con una gran cantidad de librerías de funciones y subrutinas para ayudar al usuario con la mayoría de las tareas de programación sin las molestas complicaciones que presentan los lenguajes de programación convencionales.

LabVIEW entre otras cosas cuenta librerías de de código para aplicaciones especificas de adquisición de datos (DAQ), bus de interface para propósitos generales (GPIB), instrumentos de control serial, análisis de datos, presentación de datos, almacenamiento de datos y comunicación a través de internet. La librería de análisis contiene una enorme cantidad de funciones útiles, incluyendo generación de señales, procesamiento de señales, filtros, estadísticas de ventana, regresiones, algebra lineal y arreglos aritméticos.

Debido a la naturaleza gráfica de *LabVIEW*, es inerte a la aplicación un paquete de presentación de datos y resultados. Los datos de salida pueden ser presentados de distintas formas, tantas como el usuario lo desee. Cuadros, gráficas y gráficas definidas por el usuario, comprenden sólo una parte de las opciones de salida o presentación de datos disponibles.

Los programas creados en *LabVIEW* son portables entre plataformas, de tal manera que el usuario puede crear un programa en una *Macintosh* y luego cargarlo y ejecutarlo en una máquina con *Windows* sin tocar absolutamente nada dentro de la aplicación. Y como se mencionó al inicio *LabVIEW* es una aplicación sumamente versátil que es utilizada en un gran número de industrias, desde cualquier rama de la ingeniería y control de procesos hasta biología, farmacia, psicología, química, física, enseñanza y otras muchas más.

3.3.2. Flujo de datos y lenguaje gráfico de programación

El ambiente para el desarrollo de programas en *LabVIEW*, es diferente a los ambientes de desarrollo de programas estándar como C o Java en un aspecto importante; mientras que otros sistemas de programación usan lenguajes basados en texto para crear líneas de código, *LabVIEW* usa un lenguaje de programación gráfico para crear programas en un espacio pictórico llamado diagrama de bloques (*block diagram*).

La programación gráfica elimina una buena cantidad de detalles sintéticos asociados con los lenguajes basados en texto, tales como saber dónde colocar puntos y comas (;) y corchetes ([ ]). De hecho si el usuario no sabe cómo utilizar estos símbolos en el lenguaje basado en texto, no tiene porque preocuparse ya que con *LabVIEW* no los necesitará en absoluto.

La programación gráfica permite al usuario concentrarse en el flujo de los datos dentro de la aplicación, ya que la sintaxis simple que se utiliza (símbolos gráficos) no oculta lo que el programa está haciendo. Las figuras 36 y 37 muestran una simple interfaz de usuario con *LabVIEW* y el código detrás de dicha interfaz.

Figura 36. **Interfaz de usuario**

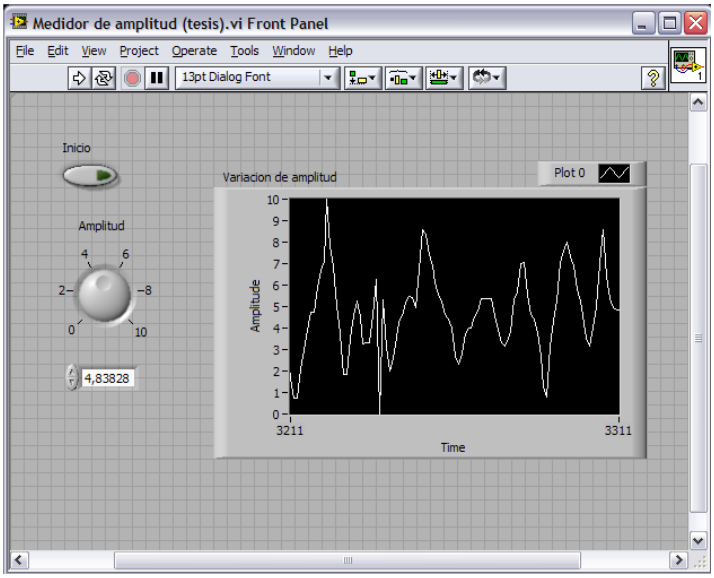
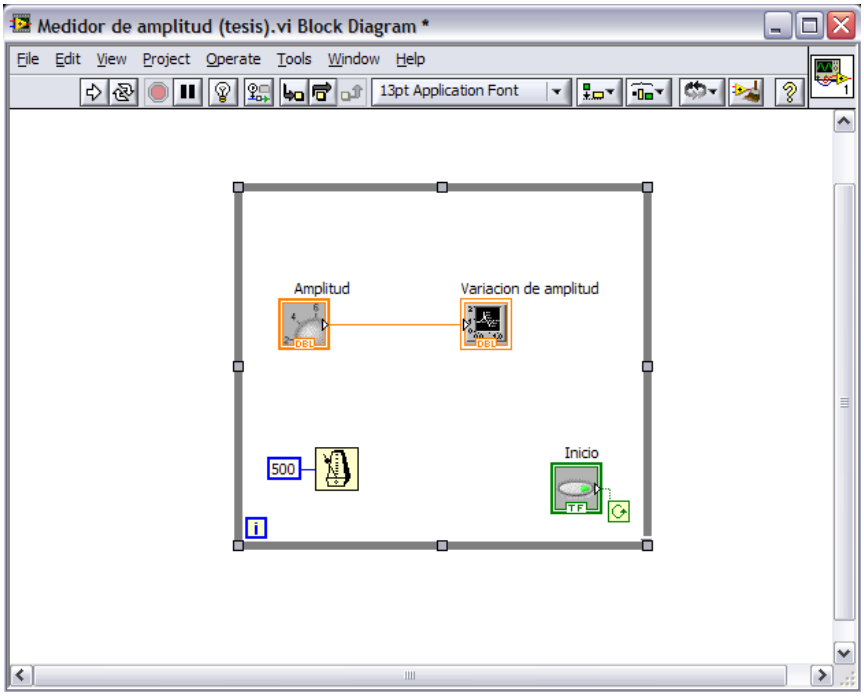


Figura 37. **Código gráfico**



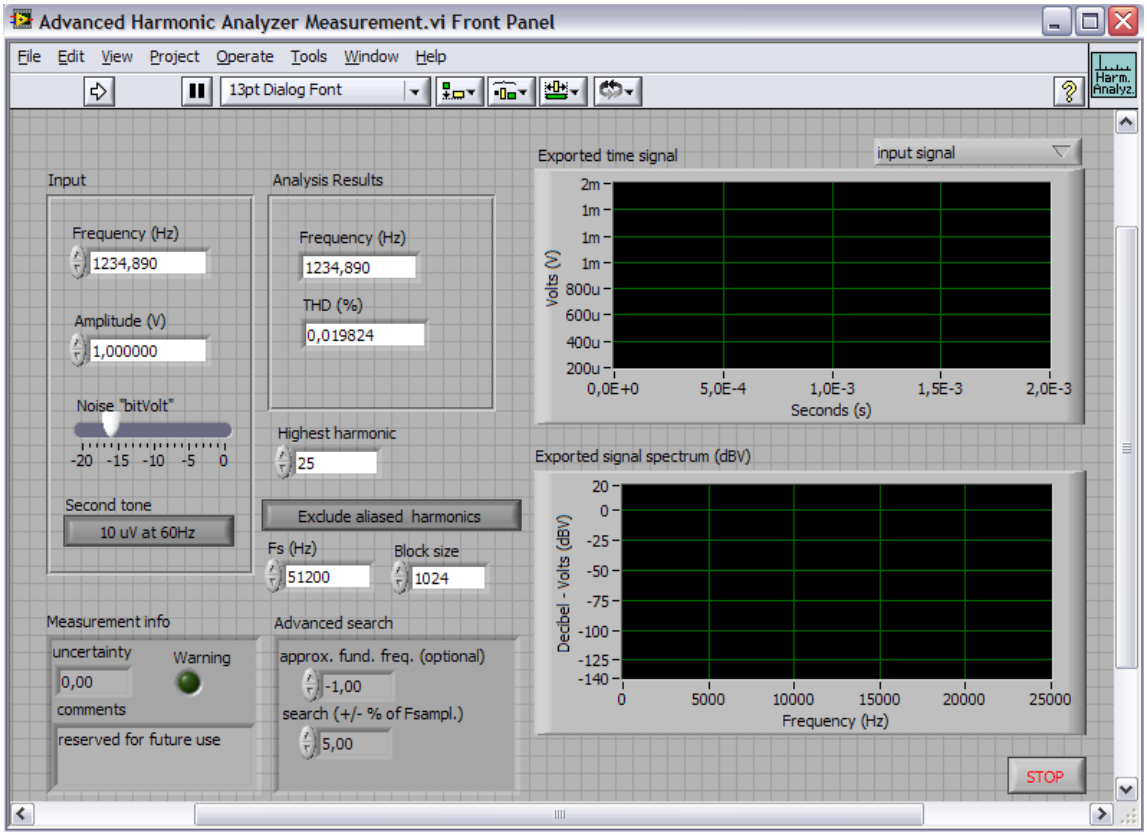
LabVIEW utiliza terminología, íconos e ideas familiares para científicos e ingenieros y recae en los símbolos en lugar del lenguaje textual para definir las acciones de un programa. La ejecución del programa está basada en el principio del flujo de datos, en el cual las funciones se ejecutan únicamente luego de recibir los datos necesarios. Debido a estas características, el usuario puede aprender *LabVIEW* incluso con mínima o ninguna experiencia. Sin embargo muchos concuerdan en que el conocimiento de las bases de programación es de mucha ayuda.

3.3.3. ¿Cómo funciona *LabVIEW*?

Un programa de *LabVIEW* consiste de uno o varios instrumentos virtuales (*VIs*). Los instrumentos virtuales son llamados así porque su apariencia y operación a menudo imitan algún instrumento físico existente. Sin embargo, tras bambalinas, ellos son análogos a los programas principales, funciones y subrutinas de los lenguajes de programación populares como C o *Basic*. De ahora en adelante se hará referencia hacia los programas creados en *LabVIEW* como VI (*Virtual Instrument*, no el número romano 6 como la gente suele interpretarlo). Entonces, debemos entender que un programa creado en *LabVIEW* siempre será llamado VI, sin importar si su apariencia u operación está relacionado con un instrumento o no.

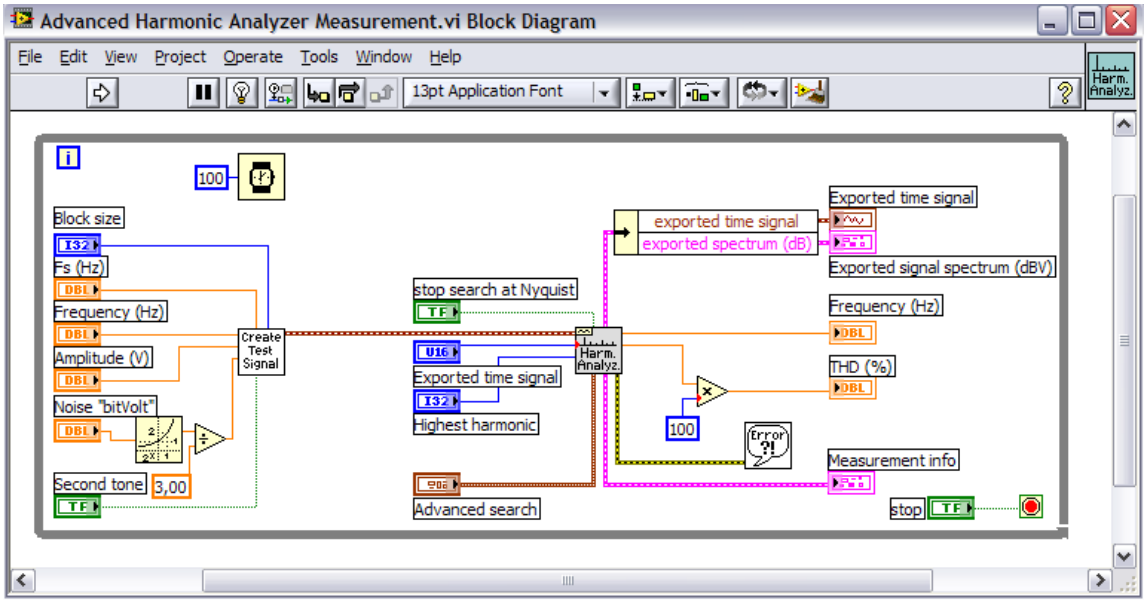
Un VI tiene tres partes principales: el panel frontal, el diagrama de bloques y los íconos; a continuación se dará una breve descripción de cada una de las partes. El panel frontal es la interfaz de usuario interactiva del VI, nombrada así porque simula el panel frontal de un instrumento físico. El panel frontal puede tener perillas, pulsadores, gráficas y muchos otros controles (que son las entradas de datos del usuario) y también indicadores (que son las salidas del programa). El usuario puede ingresar datos utilizando el ratón o el teclado, y luego observar los resultados producidos por el programa en la pantalla, la figura 38 muestra el panel frontal de un VI.

Figura 38. **Panel frontal de un VI**



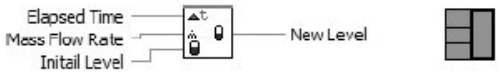
El diagrama de bloques es el código fuente del VI, construido con el lenguaje de programación gráfico de *LabVIEW*. El diagrama de bloques es el programa que se ejecutará. Los componentes del diagrama de bloques son *VIs* de bajo nivel, con funciones, constantes y estructuras de ejecución de programas. El usuario puede dibujar alambres para conectar los objetos apropiados y definir un flujo de datos entre ellos. Los objetos del panel frontal tienen terminales correspondientes en el diagrama de bloques para realizar el intercambio de datos entre el usuario y el programa, la figura 39 lo ilustra.

Figura 39. **Diagrama de bloques de un VI**



Una característica importante de los VI es que si el usuario desea utilizar alguno como una subrutina en el diagrama de bloques de otro VI, el VI utilizado como subrutina debe tener un ícono con conectores, los cuales se ilustran en la figura 40. Un VI que es utilizado en otro VI se conoce como *subVI* y es análogo a una subrutina. El ícono es una representación pictórica de un VI y puede ser utilizado como un objeto en el diagrama de bloques de otro VI. Los conectores de un VI son los mecanismos utilizados para conectar o cablear los datos en el VI de otros diagramas de bloques cuando el VI es utilizado como un *subVI*, semejante a los parámetros de una subrutina, los conectores definen las entradas y salidas del VI.

Figura 40. **Ícono de un VI (izquierda) y conectores de un VI (derecha)**



Los instrumentos virtuales son jerárquicos y modulares. El usuario puede colocarlos como programas de nivel alto o bien como subprogramas. Con esta arquitectura, *LabVIEW* promueve el concepto de programación modular. Primero el usuario divide la aplicación en una serie de simples subtareas, luego se construye un VI para completar cada una de las subtareas y finalmente se combinan los VI creados en un diagrama de bloques de nivel alto para completar la tarea larga.

La programación modular es un *plus* porque el usuario puede ejecutar cada *subVI* desde el mismo VI, facilitando la tarea de depurar los programas cuando éstos no funcionan de la manera esperada. Es más, muchos *subVIs* de nivel bajo, con frecuencia ejecutan tareas que son utilizadas por varias aplicaciones y pueden ser usados de manera independiente para cada una de ellas.

Para aclarar un poco más lo que se acaba de mencionar se hará una analogía entre *LabVIEW* y un lenguaje convencional (C++), para *LabVIEW* un VI es lo que para C++ es un programa, una función es para C++ un método, un *subVI* es para C++ una subrutina, el panel frontal es para C++ interfaz de usuario, y el diagrama de bloques es para C++ el código del programa.

3.3.4. *Toolkits* y módulos para *LabVIEW*

LabVIEW cuenta con módulos y *toolkits* especiales para incrementar la funcionalidad y versatilidad del *software*. Además nuevos *toolkits* y módulos son creados frecuentemente, de tal manera que si el usuario tiene alguna meta o aplicación en particular, puede optar por *toolkits* ya existentes que se adecuen a sus necesidades, o bien, buscar nuevos *toolkits*. La mayoria de los *toolkits* existentes fueron creados por *National Instruments* (NI), otros son creados por terceras empresas, con frecuencia llamadas empresas de alianza, y a continuación serán descritas de forma general las características de los distintos módulos y *toolkits* que existen para *LabVIEW*.

3.3.4.1. *Toolkits* y módulos para diseño embebido

National Instruments (NI) proporciona una solución de desarrollo completa para el diseño de sistemas gráficos de aplicaciones embebidas, de tal manera que el usuario pueda diseñar, generar prototipos y desplegar de manera eficiente su sistema en una sola plataforma de *software*. Esta plataforma es capaz de ejecutarse en una variedad de objetivos de procesamiento incluyendo sistemas comerciales en tiempo real y basado en arreglos de compuertas programables de campo (FPGA), así como microprocesadores y microcontroladores personalizados.

3.3.4.1.1. Módulo *LabVIEW Real Time*

El módulo de *LabVIEW Real Time* de *National Instruments* es un componente incorporable para el sistema de desarrollo de *LabVIEW*. Al ser instalado este *software* compila código gráfico de *LabVIEW* y lo optimiza para el objetivo de tiempo real seleccionado. Con el módulo *LabVIEW* real time el usuario puede desarrollar y desplegar aplicaciones a todos los objetivos *hardware* de NI, incluyendo PXI, *Compact Field Point*, *Field Point*, *Compact* RIO y *PC* de escritorio estándar. El sistema operativo en tiempo real, conocido como RTOS (*Real Time Operating System*), embebido para estos objetivos es sólo un *kernel* dedicado que proporciona máxima flexibilidad para código embebido.

3.3.4.1.2. *Real time execution trace toolkit*

Real time execution trace toolkit es una herramienta interactiva para analizar y verificar la ejecución del código de *LabVIEW* real time. Con mínimas modificaciones a su código embebido, estas herramientas muestran gráficamente la ejecución de código de hilos múltiples y destacan intercambios de hilos y asignación de memoria. Esta información ayuda a optimizar su código de tiempo real para lazos de control más rápidos y mayor rendimiento determinístico. El usuario puede utilizar puede verificar este *toolkit* para verificar la ejecución de código en objetivos *LabVIEW* real time incluyendo controladores PXI, controladores *compact field point*, *field point*, sistemas NI *compact vision* y tarjetas PCI insertables para tiempo real.

3.3.4.1.3. Módulo *labview* FPGA

LabVIEW y el módulo FPGA (*Field Programmable Gate Array*) proporcionan un ambiente de desarrollo gráfico par arreglos de compuertas programables en campo en objetivos de *hardware* de E/S reconfigurables (RIO). Con el módulo FPGA, el usuario podrá desarrollar *VIs* FPGA en un servidor ejecutando *Windows* y *LabVIEW* compila e implementa el código de *hardware*. Puede crear *VIs* de FPGA embebidos que combinan el acceso directo a E/S con la lógica de *LabVIEW* definida por el usuario para definir *hardware* personalizado para aplicaciones como protocolos de comunicación digital, simulación de control en *hardware* y rápida generación de prototipos de control.

3.3.4.1.4. Módulo NI *LabVIEW Mobile*

El módulo NI *LabVIEW Mobile* extiende el entorno de desarrollo gráfico de *LabVIEW* a dispositivos portátiles, así el usuario puede crear fácilmente aplicaciones personalizadas para ejecutar sobre *Microsoft Windows Mobile* para dispositivos *pocket* PC. El módulo *LabVIEW Mobile* es compatible con diferentes dispositivos de adquisición de datos de NI para dispositivos móviles, al usar estos dispositivos de *hardware*, el usuario puede construir sistemas de medida de bolsillo para aplicaciones que van desde diagnostico de campo hasta monitoreo fisiológico. Además el módulo para dispositivos móviles funciona con el multímetro digital NI PCMCIA-4050 de tal manera que el usuario puede construir un multímetro completamente funcional en su dispositivo móvil.

3.3.4.1.5. Módulo *LabVIEW DSP*

Con el módulo *LabVIEW* DSP de NI, el usuario puede programar gráficamente varias tarjetas *DSP* de tiempo real. Con base en *LabVIEW*, una herramienta para diseño, medición y control estándar en la industria, la naturaleza interactiva y fácil de usar del módulo DSP le ayuda a construir aplicaciones de una forma rápida y fácilmente, aún si el usuario no está familiarizado con el DSP. El módulo DSP viene con una extensa biblioteca de algoritmos para procesamiento de señales para ayudar al usuario a diseñar, probar y generar prototipos para su aplicación. También trabaja con múltiples objetivos DSP brindando, así varias opciones de *hardware* para la implementación final.

3.3.4.2. *Toolkits* y módulos para control y simulación

National Instruments ofrece un conjunto de herramientas para identificación de sistemas, diseño de control, simulación e implementación de controladores. El usuario puede aprovechar el desarrollo, análisis y visualización personalizados de algoritmos, así como integración con *hardware* de NI para generación rápida de prototipos de control.

3.3.4.2.1. Módulo NI *control design and simulation*

Con el módulo NI control design and simulation, el usuario tendrá la capacidad de analizar el comportamiento de modelos de lazo abierto, diseñar controladores de lazo cerrado, simular sistemas en y fuera de línea y realizar implementaciones físicas.

Además brinda la posibilidad de crear modelos desde el principio usando representaciones tipo función de transferencia, espacio de estado o cero-polos-ganancia. Con las herramientas de análisis de tiempo y frecuencia, como respuesta al escalón en tiempo o gráficas de Bode, podrá analizar de manera interactiva el comportamiento de lazos abiertos y cerrados.

Utilice las herramientas integradas para sistemas de entrada múltiple, salida múltiple (MIMO) y de una sola entrada, una sola salida (SISO) y aproveche las habilidades de simulación para verificar las dinámicas de sistemas lineales y no lineales. También puede utilizar las herramientas integradas para convertir sus modelos desarrollados en el *software* de *Mathworks*, *simulink* para trabajar con *LabVIEW*.

El usuario puede ampliar el uso del diseño de control y simulación de *LabVIEW* con otras herramientas de *software* de NI, por ejemplo puede usar *LabVIEW system identification toolkit*, el módulo de *LabVIEW statechart*, sobre los cuales se hablará a continuación, así como las herramientas de diseño embebido descritas anteriormente.

3.3.4.2.2. NI *LabVIEW* PID and *fuzzy logic toolkit*

El NI *LabVIEW* PID *and fuzzy logic toolkit* añade algoritmos de control sofisticados a su sistema de desarrollo de *software* de instrumentación. Al combinar las funciones de control PID y de lógica difusa en este *toolkit* con las funciones de matemática y lógica en el *software LabVIEW*, el usuario podrá rápidamente desarrollar programas para control automatizado.

También tendrá la capacidad de integrar estas herramientas de control con el *hardware* de adquisición de datos de NI y el módulo *LabVIEW real time* para crear sistemas de control robustos y determinísticos.

3.3.4.2.3. NI *LabVIEW simulation interface toolkit*

Este *toolkit* brinda a los ingenieros de diseño de control y pruebas de sistemas un enlace entre el entorno de desarrollo gráfico NI *LabVIEW* y el *software The Mathworks, Simulink*. Con el *simulation interface toolkit*, podrá construir fácilmente interfaces de usuario de *LabVIEW* personalizadas para ver y controlar su modelo durante el tiempo de ejecución.

Este *toolkit*, también brinda un complemento para uso con *The Mathworks, Real-Time Workshop*, para que el usuario pueda conectar, usando *LabVIEW*, su modelo desarrollado en el entorno *simulink* con el mundo real a través de una variedad de plataformas de E/S en tiempo real (requiere el módulo de *Real time*). Con estas habilidades, podrá fácilmente llevar sus módelos desde la verificación del *software* a la generación de prototipos en tiempo real y simulación de prueba de control en el ciclo (HIL).

3.3.4.2.4. NI *LabVIEW system identification toolkit*

El NI *LabVIEW system identification toolkit* combina herramientas de adquisición de datos con algoritmos para identificación de sistemas para modelado preciso de plantas. El usuario podrá sacar provecho de las herramientas intuitivas de adquisición de datos de *LabVIEW* como el DAQ *assistant* para simular y adquirir datos desde la planta y después automáticamente identificar un modelo de sistema dinámico. También podrá convertir los modelos para identificación de sistemas en formas de espacio de estado, función de transferencia o de ceros-polos-ganancia para análisis y diseño de sistemas de control.

El *toolkit* incluye funciones integradas para tareas comunes como procesamiento de datos, creación de modelos y análisis de sistemas. Haciendo uso de otras utilidades integradas, contará con la capacidad de imprimir el modelo con representación gráfica intuitiva así como guardar el modelo.

3.3.4.3. *Toolkits* y módulos para procesamiento de imágenes y señales

Con estas herramientas podrá incorporar cientos de funciones de procesamiento de imágenes y señales específicas para aplicaciones en sus aplicaciones de *LabVIEW*. Extienda *LabVIEW* con procesamiento específico por aplicaciones para sonido y vibración, visión artificial, comunicaciones de radio frecuencia (RF), análisis transitorios de señales, análisis de señales de corta duración y más.

3.3.4.3.1. Módulo *vison development* para *LabVIEW*

El módulo *vision development* de *National Instruments* es una colección de funciones de procesamiento de imágenes y visión artificial para varios lenguajes de programación, como NI *LabVIEW*, *Microsoft* C++, *visual basic* y *.NET*. Con estas funciones puede mejorar imágenes, verificar presencia, ubicar características, identificar objetos y medir partes. Junto con bibliotecas de programación, el módulo de visión también incluye en *NI vision assistant* y el *software* de NI *vision adquisition*.

Dicho módulo ofrece cientos de funciones para procesamiento de imágenes, incluyendo igualación de patrones e igualación geométrica, OCR, lectores de códigos de barras, clasificación de objetos, análisis de partículas, rápida generación de prototipos de aplicación y controladores para cámaras.

3.3.4.3.2. Módulo *LabVIEW Mathscript* RT

Con el módulo *mathscript* RT, el usuario puede integrar sus archivos .m personalizados al entorno gráfico de *LabVIEW*, combinar los beneficios de matemática basada en textos con programación gráfica representada en un nuevo enfoque híbrido de programación que ofrece la libertad de escoger la sintaxis más apropiada.

En el centro de *mathscript,* está un lenguaje de programación basado en texto de alto nivel con sintaxis y funcionalidad que abstrae la complejidad de las tareas de procesamiento de señales, análisis y matemáticas. Con más de 750 funciones integradas para este trabajo, usted también puede crear nuevas funciones definidas por el usuario.

LabVIEW mathscript proporciona dos metodologías de programación, una interfaz interactiva y una programática. Diseñado para el desarrollo de scripts, la ventana interactiva de *mathscript* ofrece una interfaz de líneas de comando en la cual puede cargar, almacenar, diseñar y ejecutar *scripts* de archivos .m, además el nodo *mathscript* es una característica embebida de *LabVIEW* que conecta las variables de E/S basadas en texto con las entradas y salidas de *LabVIEW*.

3.3.4.3.3. *Advanced signal processing toolkit*

El *toolkit* para procesado de señales avanzadas es un paquete de herramientas de *software*, programas de ejemplo y utilidades para el análisis de tiempo y frecuencia, análisis de series de tiempo y ondas cortas. También incluye una versión completa de NI *LabVIEW digital filter design*, del cual se hablará a continuación, y que también se encuentre disponible como un *toolkit* por separado.

3.3.4.3.4. *Digital filter design toolkit*

El *toolkit* para diseño de filtros digitales extiende *LabVIEW* con funciones (VIs de *LabVIEW* que el usuario instala en la paleta) y herramientas interactivas para el diseño, análisis e implementación de filtros digitales. Los nuevos usuarios de filtros digitales pueden explorar diseños clásicos con herramientas de diseño interactivas e integradas, mientras que los usuarios experimentados pueden encontrar una gran variedad de algoritmos, topologías de filtros y herramientas de análisis para filtros digitales de punto fijo y flotante.

Para filtros de punto fijo, puede crear efectos de cuantificación, optimizar representaciones/topologías numéricas y desplegar el diseño a un procesador de señal digital o arreglo de compuertas programables de campo (FPGA) usando *ANSI C* o código *LabVIEW* FPGA generado automáticamente.

3.3.4.3.5. *Adaptive filter toolkit*

El *toolkit* para filtros adaptivos de *LabVIEW* brinda herramientas para diseñar, analizar y simular filtros adaptivos, incluyendo de punto flotante y punto fijo, también puede utilizar estas herramientas para crear filtros adaptivo. El usuario puede aplicar estos filtros adaptivos creados para diferentes aplicaciones, como cancelación de ruido adaptivo, cancelación de eco e identificación de sistemas entre otros. Además puede crear e implementar filtros adaptivos de punto fijo en objetivos de arreglos de compuerta de campo programable (FPGA).

3.3.4.3.6. *Sound and vibration toolkit*

El *toolkit* para vibración y sonido extiende el desarrollo gráfico de *LabVIEW* con funciones e indicadores para manejar medidas de audio, análisis de fracciones de octava, análisis sinodal, medidas de nivel de sonido, análisis de frecuencia, medidas de respuesta en frecuencia, análisis transitorio y varias pantallas de sonido y vibración. También incluye el nuevo NI *sound and vibration assistant*, un *software* interactivo y autónomo para rápidamente adquirir, analizar y registrar datos acústicos, de ruido y vibración. Con una flexible biblioteca de medidas basada en configuración y análisis abierto, el *sound and vibration assistant* está diseñado para una rápida captura de datos a través de un enfoque de medidas único basado en *software* para crear aplicaciones personalizadas.

3.3.4.3.7. *Modulation toolkit*

El *toolkit* para modulación extiende la capacidad integrada de análisis de *LabVIEW* con funciones y herramientas para generación de señales, visualización y procesamiento de formatos estándares y personalizados de modulación digital y analógica. Con este *toolkit*, podrá desarrollar rápidamente aplicaciones personalizadas para investigación, diseño, caracterización, validación y pruebas de sistemas de comunicación y componentes que modulan señales. Las numerosas aplicaciones del *toolkit* de modulación incluyen formatos de modulación (AM, FM, PM, ASK, FSK, MSK, GMSK, PSK, QPSK, PAM Y QAM) que son la base de muchos estándares de comunicación digital encontrados en comunicaciones satelitales y equipos comerciales entre otros.

Para aplicaciones de RF (Radio Frecuencia), el *toolkit* de modulación complementa un *hardware* analizador de señales RF vectoriales así como un generador de señales RF vectoriales. Para operaciones de menor frecuencia (banda base o señales IF), el *toolkit* de modulación funciona con la plataforma de prueba de señales mixta de 100MHz con digitalizador, generador de formas de onda analógicas y productos de E/S de formas de onda digitales.

3.3.4.3.8. *Vision builder for automated inspection*

El NI *vision builder for automated inspection* (AI) es un entorno configurable, interactivo, guiado por menú y fácil de usar para construir, realizar pruebas comparativas y desplegar aplicaciones completas de visión artificial sin las complejidades de programación.

El usuario puede usar la interfaz de despliegue integrada para desplegar rápidamente sus aplicaciones de inspección, colocación e identificación. Con este *software* también puede instalar decisiones complejas de paso falla y comunicar resultados de inspección de E/S digital, serial o protocolos Ethernet. Además *vision builder* AI incluye funciones para diseñar variaciones del sistema que para que pueda estar seguro sobre el rendimiento del sistema aunque las condiciones reales como iluminación, movimiento, y posición de cámaras afecten su inspección visual.

3.3.4.3.9. *Math interface toolkit*

Este *toolkit* brinda a los desarrolladores de *LabVIEW* un enlace perfecto para distribuir sus aplicaciones de *LabVIEW* para uso en el entorno de análisis *MATLAB*. A través de un sistema intuitivo, podrá crear rápidamente el nombre de la función, organizar los parámetros en el prototipo de la función y personalizar automáticamente la ayuda generada para la función.

El archivo MEX final puede entonces ser distribuido para uso original en entorno *MATLAB*. Al extender el *software* de análisis *MATLAB* con *LabVIEW*, los usuarios de *MATLAB* pueden aprovechar fácilmente la amplia variedad de habilidades de E/S incluyendo adquisición de datos, control de instrumentos, movimiento, visión, interfaces de usuario intuitivas, protocolos de comunicación como TCP/IP y CAN y más de 450 funciones para análisis de medidas con *LabVIEW*.

3.3.4.4. Módulos y *toolkits* para monitoreo y control industrial

Con los toolkits y módulos que ofrece *National instruments* el usuario podrá desplegar labview hacia controladores de automatización programables (PAC) para crear sistemas distribuidos de monitoreo y control y conectarse a sus controladores de lógica programable (PLC) existentes y sistemas empresariales.

3.3.4.4.1. Módulo de *datalogging and supervisory control*

El módulo *LabVIEW datalogging and supervisory control* (DSC) de NI es la mejor manera de desarrollar interactivamente sus sistemas distribuidos de monitoreo y control. Con el módulo DSC de *LabVIEW*, el usuario cuenta con una gran cantidad de herramientas ideales y características que puede utilizar para extender las aplicaciones de *LabVIEW* ayudándolo a desarrollar fácilmente aplicaciones de registro de datos y alarmas de mucho canales sin programación.

Desarrollará su sistema de interfaz humano-máquina (HMI) y de control supervisorio y de adquisición de datos (SCADA) con características de *LabVIEW* DSC adicionales como configuración y administración de alarmas y eventos, visión de tendencias en tiempo real e históricas y configuraciones de seguridad en sus aplicaciones.

El módulo DSC de *LabVIEW* incluye herramientas como bases de datos en red compatibles con SQL para registro de datos distribuidos, alarmas y eventos basados en configuración, tendencias históricas y en tiempo real, arquitectura guiada por eventos con conectividad de cliente/servidor OPC, etiquetas ilimitadas, seguridad para aplicaciones a nivel de usuario y soporte para *Windows* XP y *Windows* XP embebido.

3.3.4.4.2. Módulo para *touch* panel

Con el módulo NI *LabVIEW touch* panel, podrá crear y desplegar rápidamente aplicaciones de interfaz humano-máquina (HMI) para comunicarse con plataformas embebidas de tiempo real como *compact field point*, *field point*, *compact* RIO y *compact vision system*. Estos *HMIs* brindan medios para mostrar datos y controlar sistemas autónomos en tiempo real. El módulo de *touch* panel incluye herramientas para desarrollar interfaces de usuario, así como funciones para comunicación, análisis de datos y almacenamiento de datos.

3.3.4.4.3. NI *Motion assistant*

Con el nuevo NI *motion assistant*, ahora podrá diseñar una aplicación de control de movimiento en un entorno interactivo en 3D y convertirlo a código C para cualquier compilador C o a instrumentos virtuales (*VIs*) de *LabVIEW*. *NI motion assistant* también expone su funcionalidad de contornos inteligentes de pendiente con un API que puede ser llamado por el usuario desde C, *visual basic* o *LabVIEW*, también puede importar perfiles de movimiento desde programas CAD y de dibujo, cuenta con la capacidad de controlar actuadores de motores paso a paso, servoactuadores y piezoactuadores y puede utilizar aplicaciones de un solo eje o múltiples ejes.

3.3.4.4.4. Módulo NI *LabVIEW softmotion*

El *software* NI *LabVIEW softmotion* y el módulo *softmotion* ofrecen desarrollo gráfico para aplicaciones personalizadas de control de movimiento. Con *softmotion* podrá utilizar un proyecto de *LabVIEW* para configurar todos sus ajustes de eje de movimiento, probar su configuración y afinar sus motores. Una vez que su configuración de *hardware* esté terminada, puede usar un panel de pruebas interactivo para validar su configuración y mover ejes individuales para verificar la configuración del *hardware*.

LabVIEW NI *softmotion* ofrece la comodidad de programar sus perfiles de movimiento con un bloque de función API de alto nivel basado en la biblioteca de control de movimiento definido por *PLCopen* incluyendo bloques de función para tipos de movimientos de línea recta, arcos y moldeados, asi como bloques de función para opciones avanzadas como engranajes y levas electrónico.

También tiene funciones avanzadas para diseño personalizado de aplicaciones de movimiento incluyendo generación de trayectorias, interpolación *spline*, control de posición y velocidad e implementación de codificador. Al combinar *LabVIEW NI softmotion* con los nuevos módulos de interfaz de controlador de la serie C para *compact* RIO, puede aprovechar la interfaz de escaneo RIO para el desarrollo simplificado de aplicaciones para movimiento coordinado de un solo eje o múltiples ejes.

Para aplicaciones que requieren funcionalidad o funciones que no están disponibles con los módulos de interfaz de controlador de la serie C, *softmotion* ofrece nodos de interfaz de ejes, los cuales pueden ser utilizados para crear aplicaciones de movimiento personalizadas que añaden E/S especializada o algoritmos de control personalizados.

Además, *LabVIEW* NI *softmotion* permite generación virtual de prototipos para aplicaciones de movimiento y diseño de máquinas al conectar la aplicación de diseño *Solidworks Premium* 3D *CAD*. Con NI *softmotion* para *Solidworks*, *National Instruments* ofrece una versión de *software* para simular sus diseños creados en *Solidworks* usando perfiles de movimiento desarrollados con bloques de función NI *softmotion*, por esta razón el usuario puede visualizar y optimizar el diseño y evaluar diferentes conceptos de diseño antes de gastar en prototipos físicos.

3.3.4.5. Módulos y *toolkits* para generación de reportes y almacenamiento de datos

Utilice estas herramientas de *LabVIEW* para exportar datos adquiridos y herramientas de *software* de terceros, como *Microsoft office* y bases de datos estándares de la industria y publicar la información en la *web*.

3.3.4.5.1. *Report generation toolkit*

El *toolkit* para generación de reportes para *Microsoft office* es una biblioteca de *VIs* flexibles y fáciles de usar para crear y editar de manera programática reportes de *Microsoft Word* y *Excel* desde NI *LabVIEW*. Ya sea que el usuario tenga necesidad de generar reportes para resumir resultados de pruebas de manufactura o compilar estadísticas de procesos para mejorar sus rendimientos de producción, el *LabVIEW report generation toolkit* acelera el desarrollo de reportes profesionales personalizados, de hecho podrá crear reportes personalizados en mucho menos tiempo usando *Microsoft office report express* VI.

3.3.4.5.2. *Internet toolkit*

El *internet* ofrece nuevas oportunidades y usos para *PCs* y estaciones de trabajo en cada industria y área de aplicación. Ingenieros e investigadores han descubierto que pueden realizar funciones importantes, como investigación, conclusiones de publicación, mostrar datos a través de la web, almacenar versiones de código fuente y programas para equipos de desarrollo de pruebas a través de *internet*. Además de aprovechar el *internet*, los instrumentos virtuales proporcionan aplicaciones remotas y distribuidas en mayor número. Al usar *LabVIEW internet toolkit* el usuario puede incorporar fácilmente una variedad de habilidades de comunicación electrónica como *XML,* CGI *y* FTP en sus aplicaciones de instrumentación virtual.

4. CARACTERISTICAS SOBRE LA CONEXIÓN ENTRE *LABVIEW* Y *SOLIDWORKS*

En los siguientes párrafos se trataran las herramientas específicas de cada software tratando, de la manera más sencilla posible, explicar cómo se hace un prototipo virtual y como se integran ambos programas dentro de la simulación de un prototipo virtual que se mueve en *Solidworks* y es controlado desde *LabVIEW.*

Primero se abordará *Solidworks*, donde se explicará de una forma sencilla cómo realizar un ensamblaje y qué características debe cumplir el ensamblaje, entre otras cosas, para permitir la comunicación bilateral de datos con *LabVIEW*.

Posteriormente será el turno de *LabVIEW* donde se hablará sobre las características del módulo utilizado para controlar la simulación del prototipo virtual y finalmente se explicará cómo se interconectan ambos programas entre sí, para lograr una controlar una simulación corriendo en *Solidworks* pero obedeciendo las órdenes desde *LabVIEW*.

4.1. Configurando un ensamblaje para simulación

Como se ha mencionado en más de una ocasión, *Solidworks* es un programa de diseño asistido por computadora (CAD) para crear desde piezas mecánicas simples hasta complicados y enredados sistemas mecánicos compuestos de muchas piezas con diseños que varían entre sí.

El primer paso para poder realizar la simulación de un mecanismo en *Solidworks*, es obviamente pensar en el mecanismo; lo cual lleva a pensar de forma lógica en lo siguiente: tener en mente la idea de lo que se pretende crear, crear las piezas y finalmente unirlas para materializar la idea; o bien para este caso, el adjetivo virtualizar la idea aplica, ya que lo que se hará es crear un prototipo virtual de algo.

Ahora, todos tienen la capacidad para imaginar piezas y mecanismos, pero no todos saben cómo trasladar esas ideas de la mente al monitor de una computadora, es por eso que a continuación se muestra de una forma rápida cómo crear una pieza en *Solidworks.* Lo primero es ejecutar *Solidworks* y atención en lo siguiente, para lograr combinar de una manera adecuada *Solidworks* y *LabVIEW*, el usuario debe tener *Solidworks Premium* 2009 *Service Pack* 2.1 o superior. *LabVIEW* también tiene sus requisitos pero al respecto se hablará más adelante.

Una vez que se tenga el programa corriendo, se deberá observar una pantalla como la que muestra la figura 41 que es la pantalla inicial, se busca el ícono universal para nuevo (que es la hoja en blanco) y se selecciona el campo donde dice pieza y hacemos clic en aceptar, este procedimiento se muestra en la figura 42.

Figura 41. **Pantalla de inicio de *Solidworks***

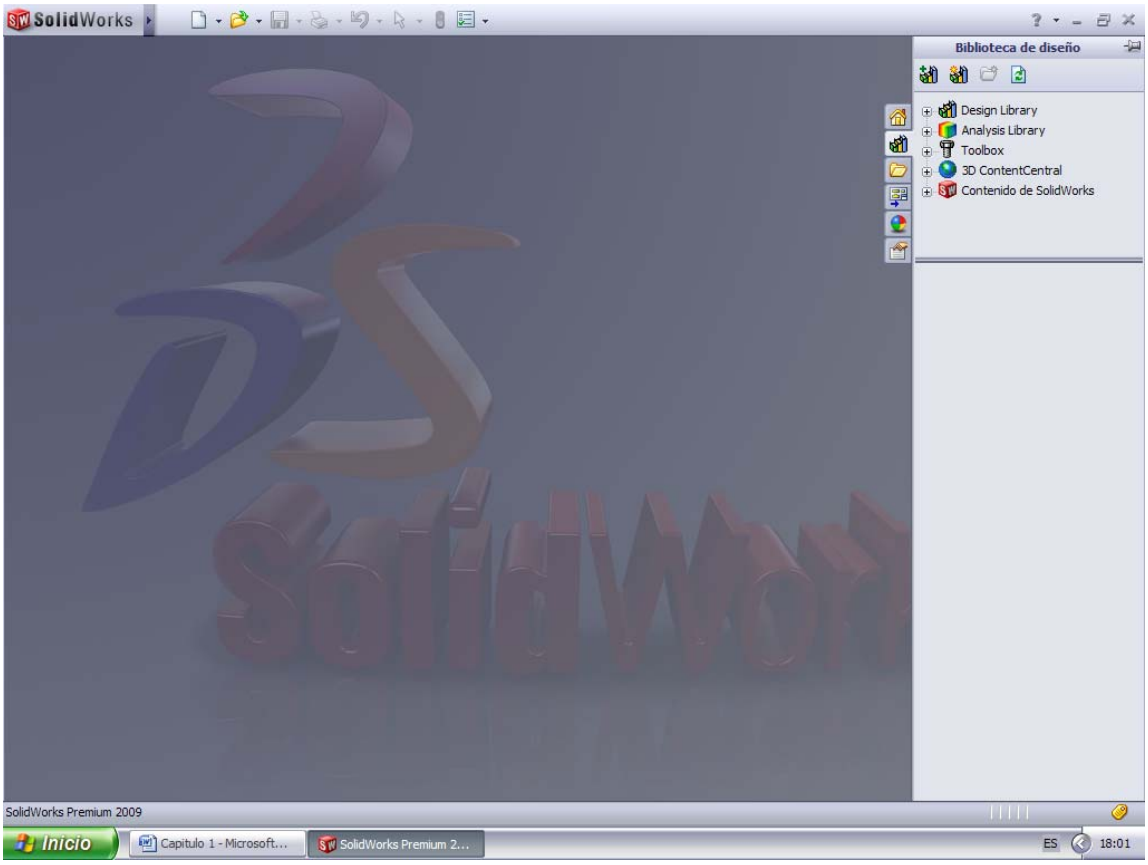
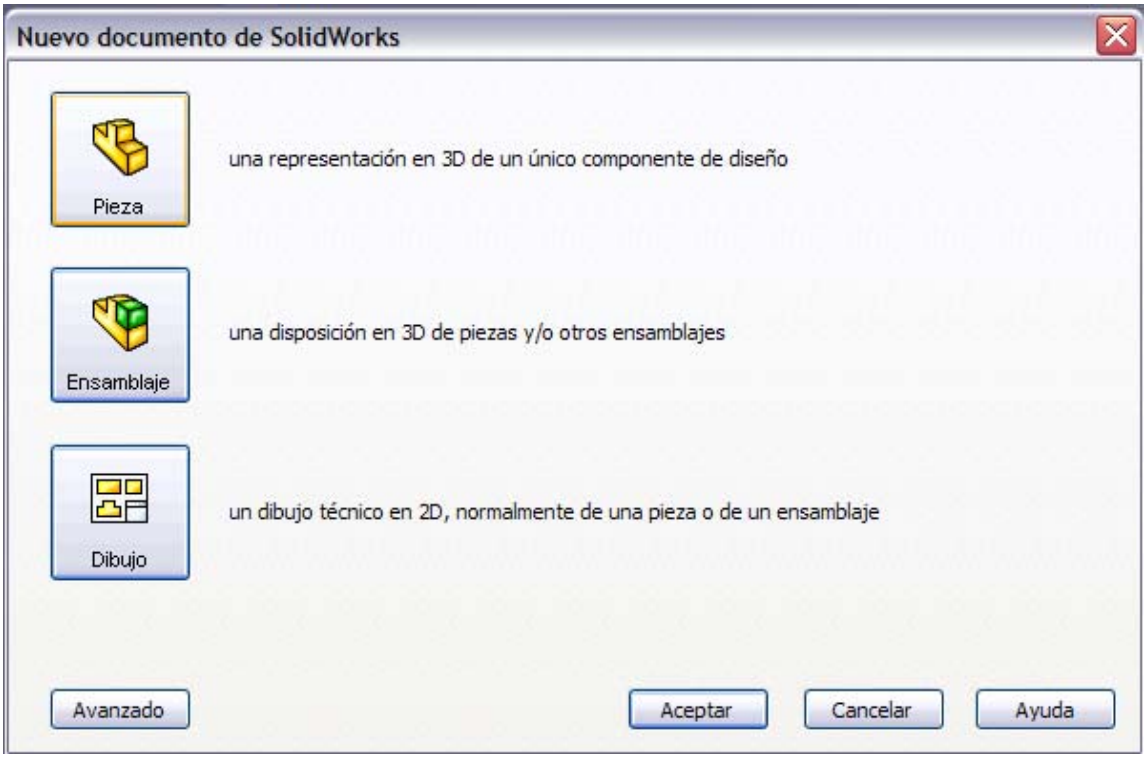
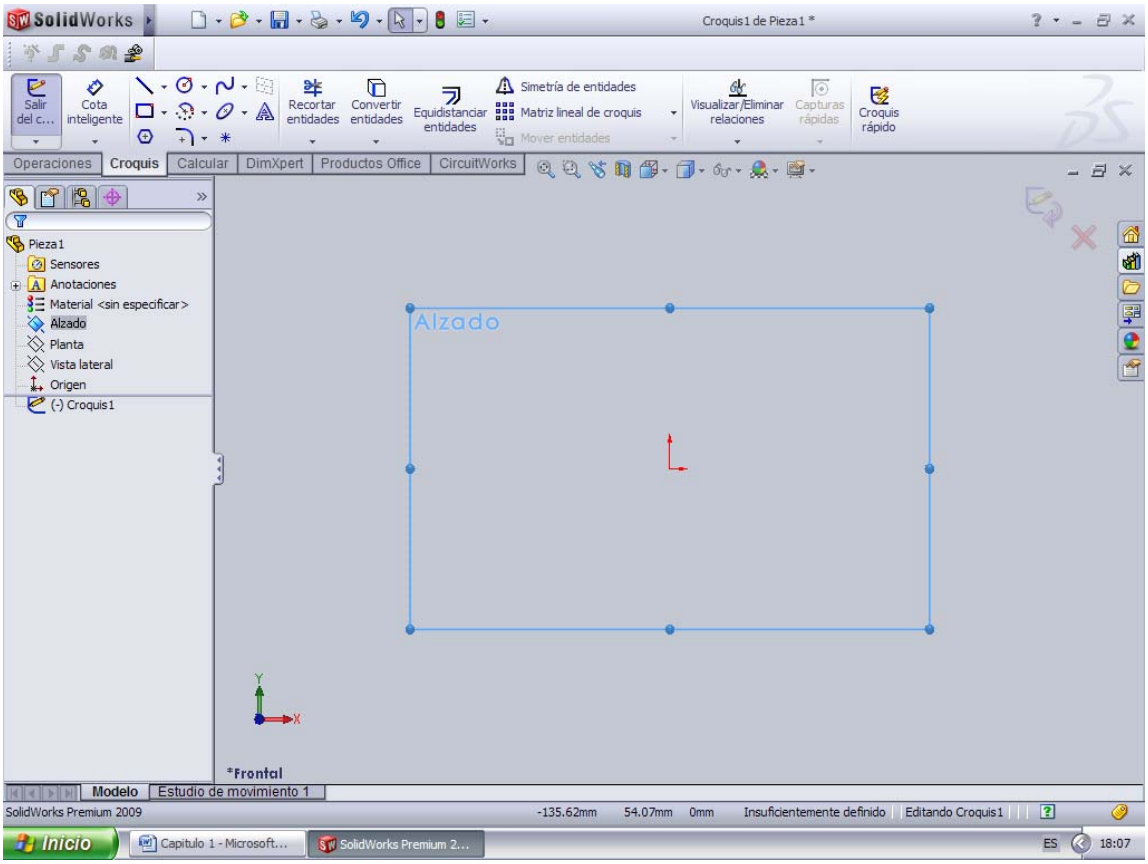


Figura 42. **Botón de nuevo y nueva pieza**



Una vez que se ha hecho lo anterior tendremos frente a nosotros el espacio para crear una pieza tan simple o tan complicada como se desee. El área de trabajo se muestra en la figura 43.

Figura 43. **Área de trabajo para creación de piezas**



Listo, ya se puede iniciar a plasmar todas las ideas en el monitor de una computadora. *Solidworks* es una herramienta de nivel profesional por lo cual no se explicará cómo utilizar algunas de las funciones para crear piezas, ya que desviaría el enfoque buscado en este trabajo de graduación; de hecho podría hacerse un trabajo de graduación solo de cómo crear piezas en *Solidworks* para que tenga idea de la cantidad de herramientas con que cuenta este *software.*

Una vez que se tengan por separado las piezas necesarias para nuestro mecanismo, será necesario unirlas para validar el diseño, y como el lector podrá intuir, *Solidworks* ayudará en esta tarea. Lo que se debe hacer es crear un nuevo archivo siguiendo el procedimiento que se mostró en la figura 42 con la diferencia que seleccionará ensamblaje en lugar de pieza y luego clic en aceptar y tendrá un espacio para crear relaciones de posición entre las distintas piezas y un estudio de movimiento, cosas que serán tratadas con un poco más de detalle a continuación.

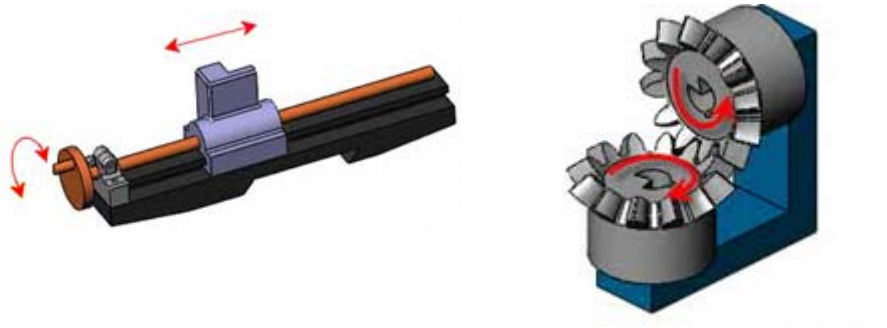
Al crear un nuevo ensamblaje, *Solidworks* proporciona un espacio en el cual el usuario podrá unir las distintas piezas que ha creado previamente. Este es un paso importantísimo dentro de la creación y simulación de un prototipo virtual, ya que si se hace un mecanismo deficiente la simulación se verá limitada y el diseñador se verá forzado a volver a trabajar para corregir los errores. Tratando de evitar esto, se hablará sobre los requerimientos generales para configurar un ensamblaje o mecanismo para simulación.

El primer paso es estar seguro que todas las relaciones entre las piezas fueron colocadas de manera apropiada. Existen distintas relaciones de posición en *Solidworks* y más adelante se explicarán algunas de las más importantes. El siguiente paso es definir el estudio de movimiento, el estudio de movimiento es el ambiente de simulación de *Solidworks* y es donde el usuario puede crear motores, fuerzas, configurar parámetros físicos del ensamblaje, propiedades de masa y fricción.

Lo más importante de un ensamblaje son las relaciones entre las piezas, de hecho la libertad de movimiento en ciertas partes del ensamblaje dependen de las relaciones de posición que use. Las relaciones de posición estándar son las relaciones que permiten unir las distintas piezas entre sí y mantenerlas en su lugar, entre éstas tenemos concéntricas, coincidentes, paralelas, perpendiculares, de distancia entre otras.

Por otro lado están las relaciones de posición mecánicas, un poco más avanzadas respecto de las estándar, son utilizadas para controlar el movimiento de uno o varios componentes con respecto al movimiento de otro componente, ejemplos de éstas son las relaciones de engranajes, de tornillo y de cremallera. Las relaciones de engranaje induce la rotación de un componente cuando otro componente también rota, la relación de tornillo por el otro lado es similar pero diferente, la relación de tornillo logra que cuando un componente rota el otro componente involucrado en la relación se desplace o traslade de manera lineal. La figura 44 muestra estas relaciones.

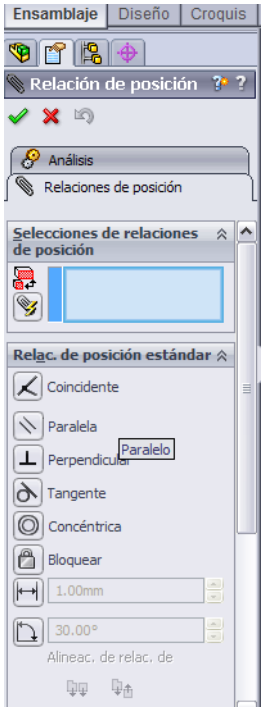
Figura 44. **Relaciones mecánicas de tornillo y de engranaje**



Fuente: http://zone.ni.com/wv/app/doc/p/id/wv-1417

Ahora se explicarán algunos detalles sobre las relaciones estándar y mecánicas para dar una idea menos abstracta de cómo funcionan las relaciones entre piezas, la figura 45 muestra la barra de relaciones estándar.

Figura 45. **Menú de relaciones de posición estándar**



Para asignar una relación a dos piezas, basta con seleccionar las caras de las piezas que el usuario desee relacionar y aplicar la relación deseada haciendo clic sobre ella, si la relación fuese imposible de realizar *Solidworks* le dará una advertencia, de lo contrario todo funcionará bien y en hora buena habrá agregado una relación de posición estándar entre dos piezas separadas.

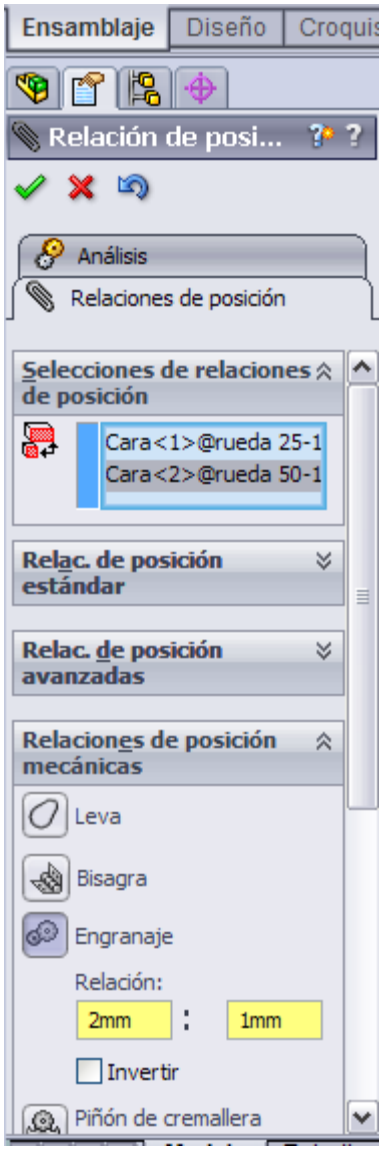
El caso de las relaciones mecánicas funciona de manera similar que las relaciones estándar, con la diferencia de que debido a la naturaleza avanzada de estas relaciones requieren parámetros adicionales a las caras de las piezas que se desean relacionar sobre los cuales se hablará en seguida.

Para una relación mecánica de engranajes es necesario saber lo siguiente, primero ambas piezas deben poder rotar de lo contrario nunca podrá aplicar la relación, la cara de la primera pieza que seleccione será la pieza maestra y la cara de la otra pieza (segunda pieza) será la pieza esclava.

Una vez seleccionadas las caras de las piezas maestra y esclava respectivamente, *Solidworks* automáticamente medirá los diámetros de ambas piezas y los colocará en un espacio dentro del menú con el nombre de relación. Es importante hacer saber al usuario que estos valores en realidad no importan, lo que realmente importa es la razón entre estos valores. Así que sin ningún problema el usuario puede dejar estos valores teniendo siempre en cuenta que lo realmente importante es la razón entre dichos valores o bien colocar una relación o razón deseada tal como 3:1; 1:2; 2:1, etc.

Este parámetro determina cuantas revoluciones dará la pieza esclava por cada revolución de la maestra. Por ejemplo una relación 2:1 significa que la esclava dará dos revoluciones por cada revolución de la maestra y una relación de vueltas 1:2 indica que por cada revolución de la pieza maestra la pieza esclava dará media revolución o bien que serán necesarias dos vueltas completas de la pieza maestra para completar una revolución de la pieza esclava, la figura 46 muestra este menú.

Figura 46. **Menú de relación de posición mecánica de engranajes**

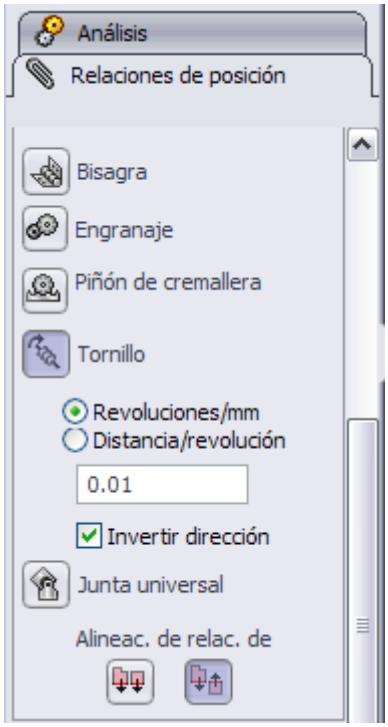


Ahora, se hablará sobre la relación mecánica de tornillo, que de hecho fue la que se utilizó para crear el prototipo presentado en este trabajo de graduación. La forma de operar es similar a la relación de engranajes, primero se deben tener las piezas adecuadas para poder aplicar esta relación; en el caso de la relación de engranaje ambas piezas eran ruedas, ahora ambas piezas deben ser cilíndricas y una de ellas debe de poder girar.

La primera cara a seleccionar debe ser la cara de la pieza que puede girar y la segunda cara seleccionada debe tener un agujero cilíndrico por el cual pase la primera pieza y así permitir el desplazamiento lineal de la segunda pieza cuando la primera pieza gire. Esta relación también requiere un parámetro extra que tiene dos opciones que permite escoger entre revoluciones por milímetro y distancia por revolución para controlar el desplazamiento lineal con relación al movimiento rotacional.

La relación revoluciones por milímetro permite determinar la fracción de revolución que tendrá la pieza giratoria con respecto a un milímetro, el número predeterminado es 0.01 lo cual quiere decir que la fracción de revolución 0.01 desplazará 1 milímetro la pieza móvil. La otra opción que relaciona distancia por revolución (la más usada) indica cuantos milímetros se desplazará la pieza móvil por cada revolución de la pieza rotativa, es decir si colocamos 10 en la casilla por cada revolución de la pieza rotativa la pieza móvil se trasladará 10 milímetros en forma lineal. La figura 47 ilustra el menú.

Figura 47. **Menú de relación de posición mecánica de tornillo**



De manera similar, cada una de las relaciones de posición mecánicas que se muestran en las figuras 46 y 47 y de las que no se dijo nada, como relación de piñón, de bisagra y relación de leva; requieren de la selección de dos caras de piezas distintas que estén adecuadas al tipo de relación que se desee aplicar además del establecimiento de un parámetro extra que tiene que ver con el tipo de movimiento que generaran las piezas una vez que hayan sido relacionadas de la manera correcta.

Ahora que se sabe cómo funcionan las relaciones mecánicas de engranajes y tornillo se revisarán los conceptos claves de cada relación. La relación mecánica de engranaje induce movimiento rotacional de una pieza (esclava) cuando la otra pieza relacionada (maestra) rota.

Los valores individuales de las circunferencias o diámetros de las piezas no son realmente importantes, lo que importa es la razón entre estos valores. La relación mecánica de tornillo funciona de tal manera que la rotación de una pieza induce movimiento de traslación lineal en la otra pieza perteneciente a la relación y podemos controlar cuanto se desplaza la pieza que se trasladará linealmente por cada revolución de la pieza que realiza el movimiento de rotación.

4.1.1. Estudio de movimiento

Ahora que se sabe cómo relacionar de manera adecuada las partes del ensamblaje es hora de hablar lo referente a la simulación, así que a continuación se hablará sobre los estudios de movimiento y su importancia para lograr la simulación de un ensamblaje o sistema mecánico bastante semejante a la realidad.

Un estudio de movimiento es el ambiente de simulación con que cuenta *Solidworks* para poner a prueba los ensamblajes creados, el estudio de movimiento utiliza las propiedades de determinado ensamblaje para simularlo, dentro de estas propiedades están la masa de los componentes del ensamblaje, cualquier relación entre piezas que se haya creado y así todas las propiedades importantes para correr una simulación.

Adicionalmente en el estudio de movimiento se pueden agregar componentes extras, que no forman parte del ensamblaje original, como motores, fuerzas y fricción entre otros que ayudan en el momento de poner a prueba lo que ha sido creado y verificar si funciona como se espera.

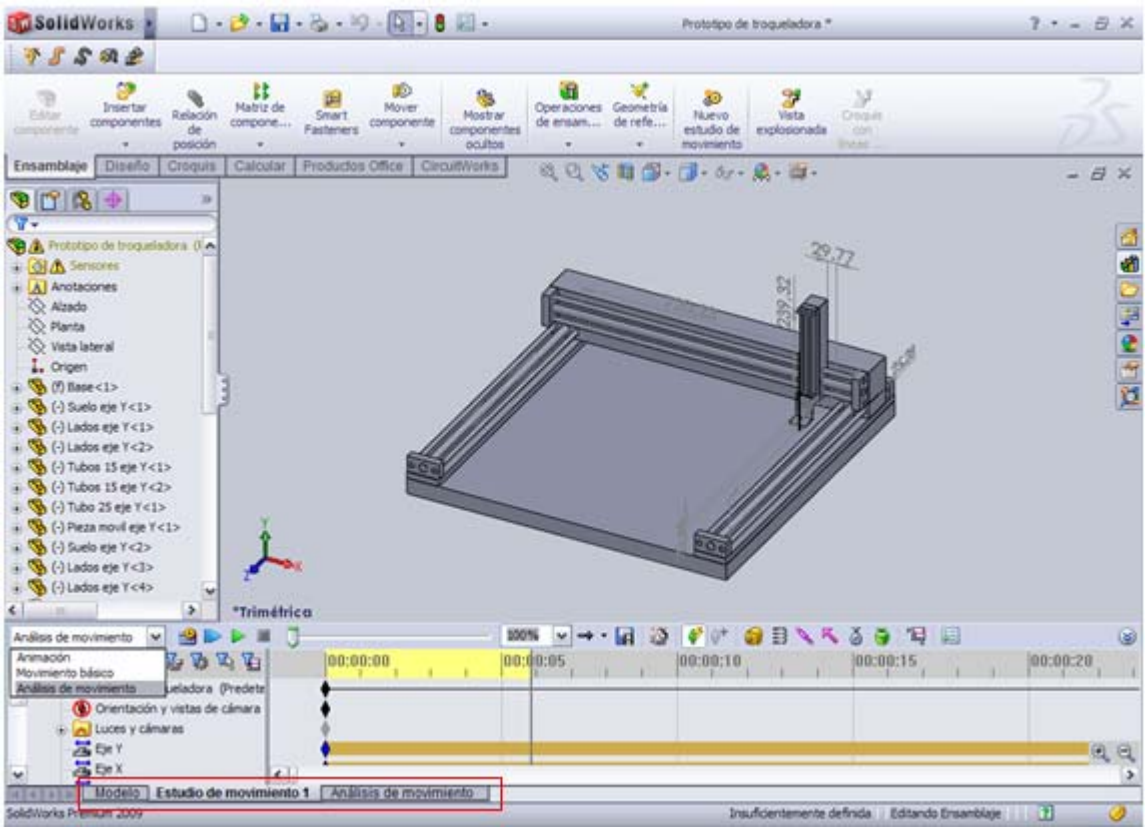
Existen tres tipos de estudio de movimiento disponibles en *Solidworks* y éstos son animación, movimiento básico y análisis de movimiento; este último es el único de los estudios de movimiento capaz de comunicarse con LabVIEW, de manera que cada vez que el usuario desee simular un mecanismo con *LabVIEW* debe asegurarse que seleccionó el tipo de estudio de movimiento análisis de movimiento, sobré el cual se darán algunos detalles a continuación.

El estudio de movimiento análisis de movimiento es un solucionador de cinemática que toma en cuenta todas las características físicas de los ensamblajes incluyendo fricción, masa de los componentes, de tal manera que cualquier fuerza y cualquier cosa que el usuario defina para un ensamblaje serán representadas matemáticamente dentro de este solucionador y por lo tanto tomado en cuenta a la hora de realizar una simulación.

También cuenta con un menú para resultados de manera que el usuario al final del estudio puede elegir entre una serie de herramientas que le permitirán observar datos numérico y gráficas de diversos parámetros que ayudarán a realizar un estudio mucho más detallado sobre el mecanismo puesto bajo análisis; un aspecto importante a tener en cuenta es que para poder ejecutar un estudio de análisis de movimiento debe contar con *Solidworks Premium* 2009.

La figura 48 ilustra como luce *Solidworks* cuando el usuario se encuentra en el ambiente del estudio de movimiento, si se observa la parte inferior de la ventana, se verán tres etiquetas con los nombres modelo, estudio de movimiento y análisis de movimiento. Si estuviese seleccionada la etiqueta de modelo significaría que el usuario se encuentra trabajando en el ambiente regular de diseño de *Solidworks* y las dos restantes, en efecto, llevan al usuario al ambiente de simulación de *Solidworks*.

Figura 48. **Ambiente de simulación de *Solidworks***

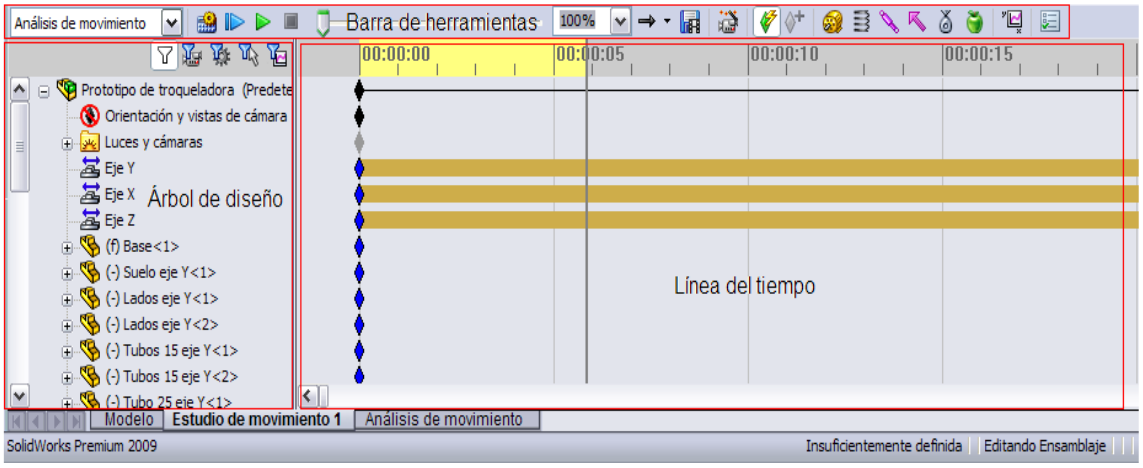


Motion manager, el *motion manager* tiene diferentes áreas que serán tratadas en las siguientes líneas. La primera es la barra de herramientas, que es donde el usuario puede seleccionar el tipo de estudio de movimiento, también contiene herramientas con las que puedo crear motores, resortes y otros componentes importantes para el estudio de movimiento. La siguiente parte del *motion manager* es el árbol de diseño, que es el espacio donde cualquier estudio de movimiento creado se podrá observar junto con todos los componentes del ensamblaje bajo análisis.

Permite, además, agregar y eliminar elementos del ensamblaje que sólo afectaran al estudio de movimiento actual, de manera que el usuario puede eliminar elementos innecesarios para el estudio de movimiento y si vuelve al espacio de diseño (etiqueta de modelo) los elementos eliminados anteriormente estarán disponibles.

La parte final del *motion manager* es la línea del tiempo que indica al usuario cuando se está corriendo una simulación y en que parte de la simulación se encuentra, además indica qué parte del ensamblaje se está moviendo y en qué tiempo se está ejecutando esta acción. De nuevo como en muchos pasajes anteriores se hará un paréntesis para hablar de *LabVIEW*, cuando se está ejecutando una simulación con *LabVIEW* y *Solidworks* la línea del tiempo no se utiliza mucho ya que el monitoreo de las acciones se hará desde *LabVIEW,* la figura 49 mostrará el *motion manager.*

Figura 49. ***Motion manager*** **y sus partes**

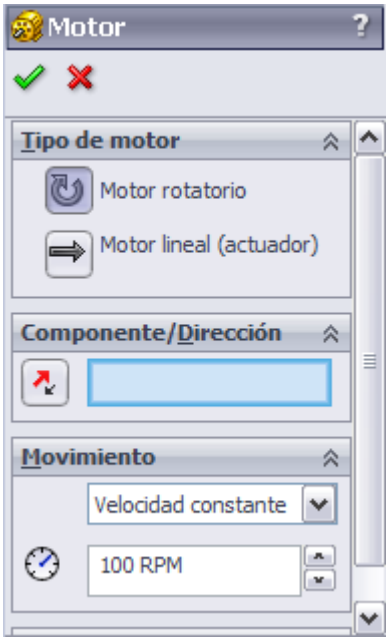


Motores, los motores son parte indiscutible de un estudio de movimiento y para los objetivos de esta tesis son una parte fundamental ya que estos motores son los que *LabVIEW* controlará a la hora de ejecutar la simulación final.

Los motores pueden ser adheridos a cualquier componente dentro del mecanismo y pueden mover dicho componente brindándole los grados de libertad con que el componente cuenta dentro del ensamblaje basado en las relaciones de posición y/o mecánicas que posee. Existen dos tipos de motores, el primero es el motor rotativo, el cual al funcionar induce un movimiento de rotación al componente al que está adherido y el segundo es el motor lineal o comúnmente conocido actuador que al funcionar desplaza linealmente el componente que gobierna.

Los pasos para agregar un motor al ensamblaje son dos, el primero es especificar en donde y sobre que componente el usuario desea colocar el motor, lo que también determinará la dirección de movimiento del motor y el segundo paso es seleccionar el tipo de motor. De tal manera que para colocar un motor rotativo, por ejemplo, se tendría que hacer lo siguiente, primero seleccionar el ícono del motor, que se encuentra en la barra de herramientas del *motion manager*, acción que mostrará un menú sobre las propiedades del motor como el que se muestra en la figura 50.

Figura 50. **Ícono del motor y propiedades del motor**



Con las propiedades a la vista el usuario debe seleccionar el tipo de motor (rotativo para este ejemplo), luego escoger la cara de la pieza sobre la cual desea colocar el motor y dependiendo de la cara que sea seleccionada y de las propiedades de dicha cara, será mostrada la dirección del movimiento por una flecha. Así para un motor rotativo el usuario deberá seleccionar cualquier pieza cilíndrica o una pieza con bordes circulares que sean capaces de girar.

A continuación se seleccionará el tipo de movimiento que el usuario desea realizar con el motor rotativo, existen varias opciones para este campo, pero en orden de realizar una configuración del estudio de movimiento para lograr conectarlo con *LabVIEW,* el usuario deberá seleccionar el tipo de movimiento de distancia. De seleccionar algún otro tipo de movimiento la comunicación entre *Solidworks* y *LabVIEW* no será establecida.

Los siguientes parámetros a ingresar, una vez seleccionado el tipo de movimiento distancia son el tiempo y precisamente la distancia que el motor moverá el componente al cual fue ligado. Es importante resaltar que en el momento de realizar la simulación de un mecanismo de *Solidworks* con *LabVIEW*, éste último ignorará por completo los parámetros de tiempo y distancia porque no los necesita, pero si son necesarios para definir de manera adecuada que acción realizará el motor dentro de *Solidworks*.

Ahora, el hecho que el control de la simulación se haga con *LabVIEW* no significa que *Solidworks* no pueda ejecutar algún control sobre las simulaciones por sí solo, de hecho los parámetros de distancia y tiempo son utilizados por *Solidworks* para controlar la ejecución de dicha simulación.

Y es precisamente acá donde reside la razón de que *LabVIEW* ignore completamente los parámetros de control utilizados por *Solidworks* para controlar las simulaciones, ya que una simulación no puede hacerle caso a dos cosas a la vez. Además *LabVIEW* cuenta con características que brindan mayor control sobre la simulación, pero de esto se hablará más adelante.

Volviendo a los motores y sus características, el caso del motor lineal sigue la misma secuencia lógica de pasos que para el motor rotativo, las diferencias que se deben tener en cuenta son que para este motor la cara a seleccionar debe ser de una pieza con bordes rectos y planos ya que el movimiento que generará será de traslación y no de rotación como en el caso anterior y obviamente seleccionar el tipo de motor lineal y no rotativo. Las figuras 51 y 52 muestran ambos casos.

Figura 51. **Aplicación del motor rotativo en un mecanismo**

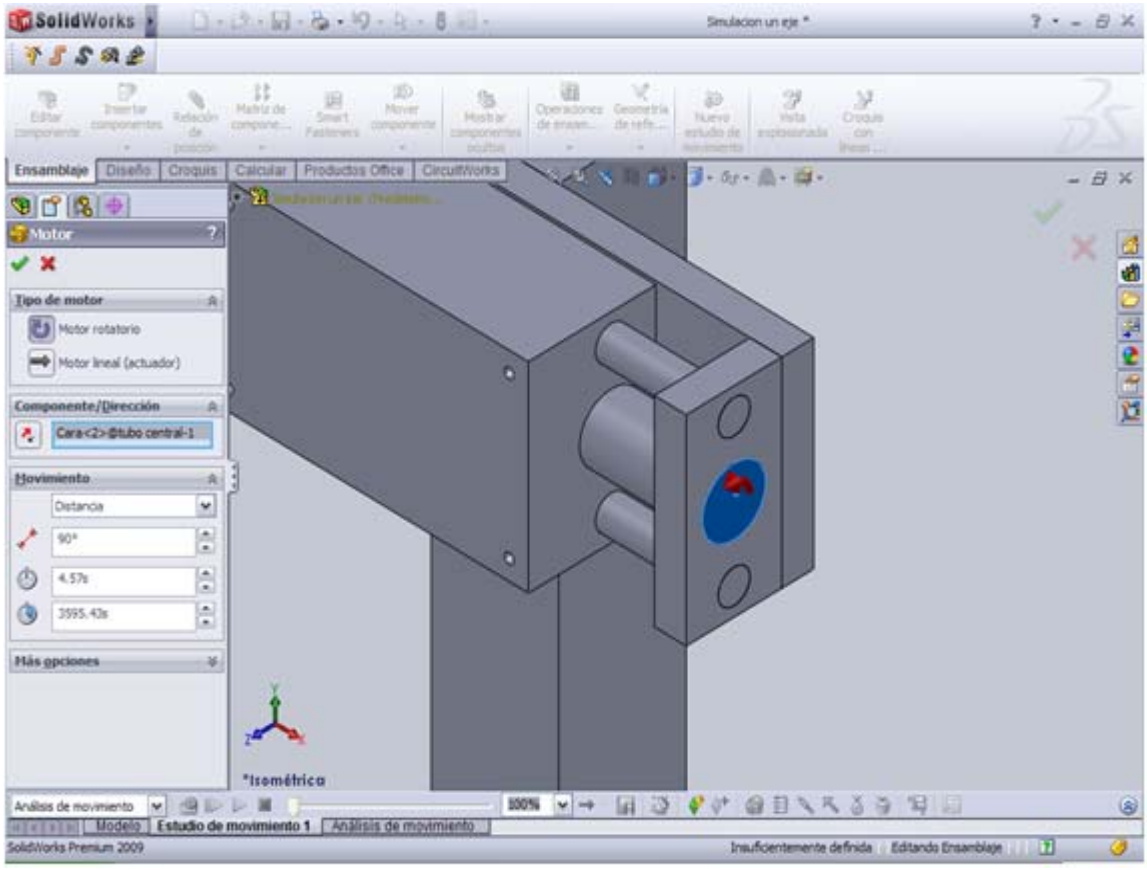
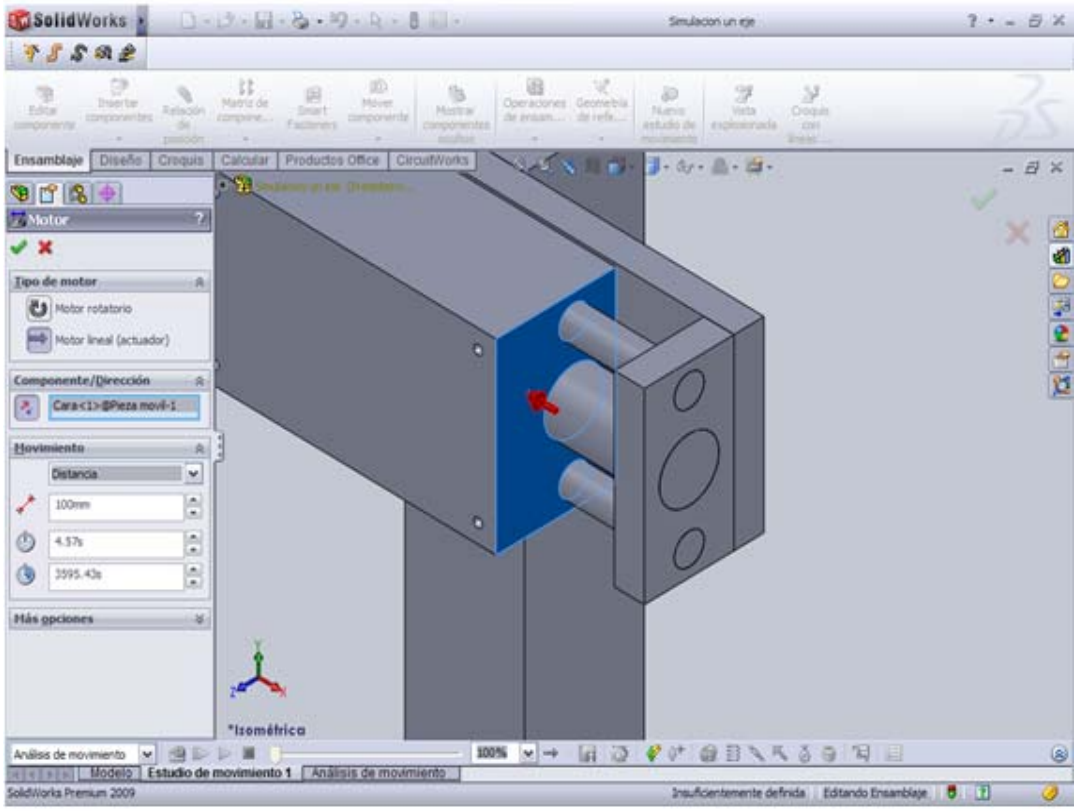


Figura 52. **Aplicación del motor lineal en un mecanismo**



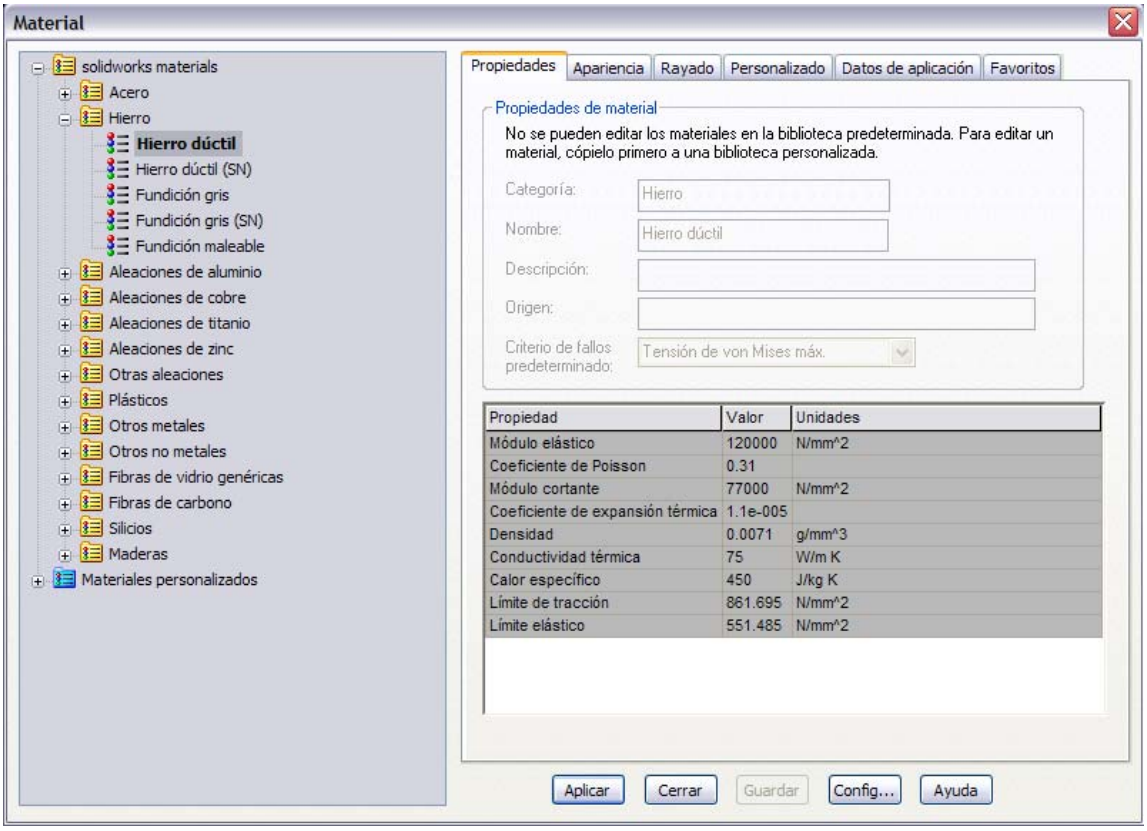
Como se logra observar en las figuras, se aplicó distinto tipo de motor al mismo mecanismo, pero en distintos componentes. El mecanismo utilizado se conoce como fleje y consta de una pieza móvil que se desplaza en forma lineal conforme un tornillo sin fin rota. La pieza que simula el tornillo sin fin es el cilindro de mayor diámetro y al cual fue aplicado el motor rotativo y la pieza móvil que se desplaza conforme la rotación, se le aplicó el motor lineal, si observa las flechas que aparecen en el mecanismo notará que indican en qué dirección se moverá el componente.

Ahora se enfocará la atención hacia otros parámetros físicos que son importantes a la hora de realizar una simulación, como se mencionó unos párrafos atrás el solucionador de cinemática de *Solidworks* tomará en cuenta todas las distintas propiedades físicas del ensamblaje, estas incluyen masa y fricción entre los componentes. Estas propiedades pueden no ser calculadas automáticamente por *Solidworks*, por lo que el usuario puede colocarlas para mejorar los resultados de la simulación.

Especificar las propiedades de inercia y masa de los componentes puede hacerse de dos maneras, la primera es especificar el material del que estarán hechos los componentes, ya que *Solidworks* cuenta con una base de datos con todas las propiedades para cada material, una de estas propiedades es la densidad, la cual ayuda a calcular dicha masa, y la otra forma es especificar arbitrariamente la masa de un componente introduciendo un valor desde el teclado. La inercia en el caso de componentes que rotan, requiere del cálculo de fuerzas dada la aceleración que lo afecta, es calculada automáticamente basándose en la geometría de la pieza.

Para llevar a cabo la especificación de la masa de un componente en *Solidworks*, lo primero es tener un componente creado en *Solidworks*, luego seleccionará el componente y hará clic derecho sobre él y verá un menú con las propiedades del componente, buscará y seleccionará material y luego editar material. Al hacer clic en editar material aparecerá una ventana como la mostrada en la figura 53.

Figura 53. **Ventana para editar materiales en *Solidworks***



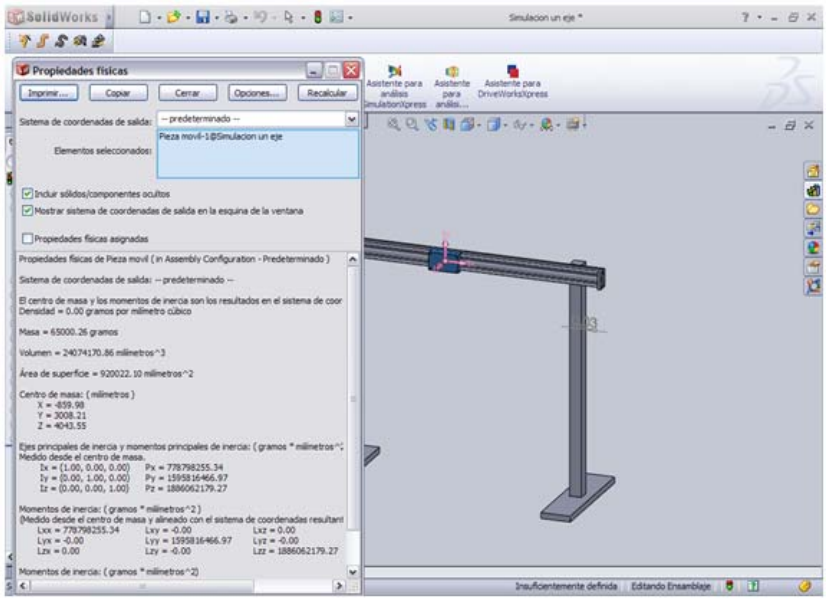
Como se observa hay una gran variedad de materiales para escoger y cada uno cuenta con sus propiedades físicas. Una vez escogido el material damos clic en aceptar y listo habrá aplicado un material específico con características físicas propias que *Solidworks* tomará de la base de datos para dicho material y servirán para calcular distintos parámetros dentro de la simulación.

Para verificar las propiedades como masa, centro de masa, volumen y otras características físicas de la pieza se usa la etiqueta que se encuentra en la parte superior de la ventana de *Solidworks* y que está identificada como calcular, y luego clic en el botón de propiedades físicas, la figura 54 ilustra.

Figura 54. **Etiqueta calcular y botón de propiedades físicas**

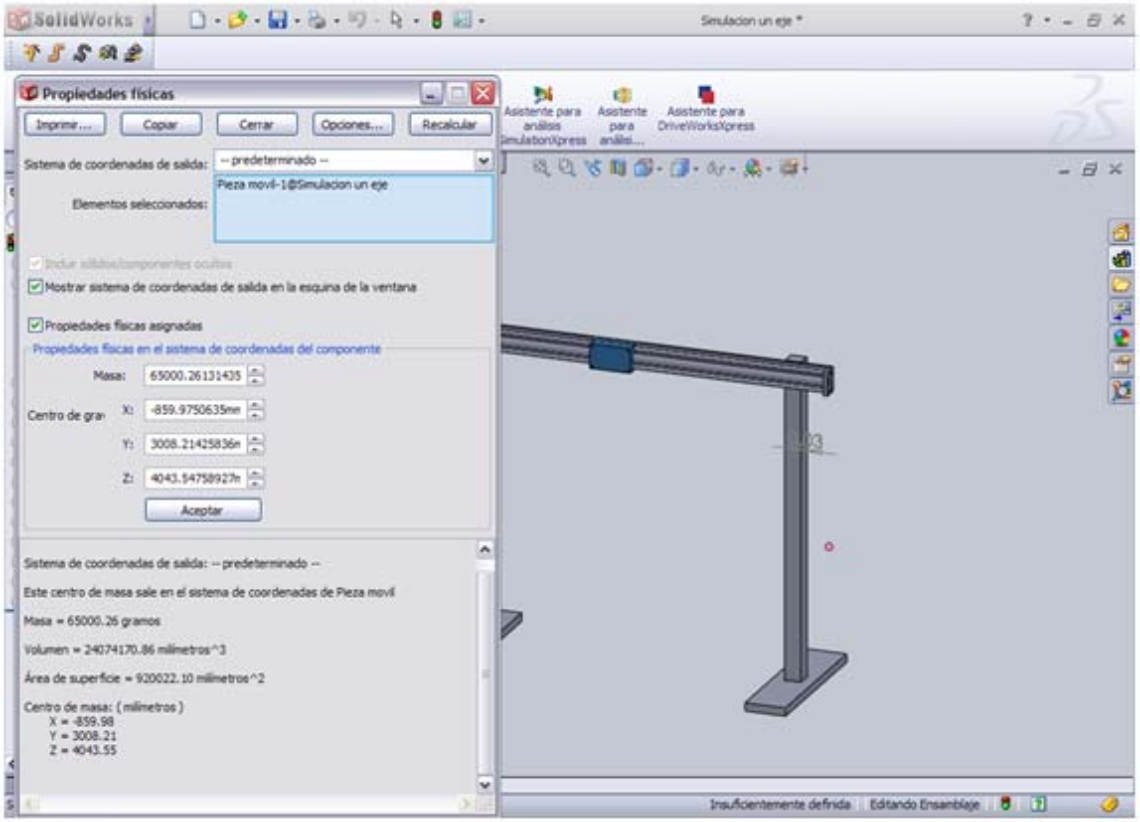
Al hacer clic en este botón el usuario verá una ventana que contiene las propiedades físicas de la pieza seleccionada con los valores de masa correspondientes al material seleccionado para la pieza además de otros datos de interés como centro de masa, volumen, área superficial de la pieza, etc. la figura 55 muestra la ventana.

Figura 55. **Ventana de propiedades físicas de una pieza**



Esta ventana tiene una opción llamada propiedades físicas asignadas que se encuentra desactivada, esta opción es precisamente la segunda forma de asignar la masa a un componente, ya que al momento de activarla el usuario tiene la capacidad de indicar el valor que él considere adecuado para la masa del componente e incluso puede también colocar coordenadas para especificar el centro de gravedad, la figura 56 muestra como varia la ventana de propiedades físicas al habilitar la opción mencionada.

Figura 56. **Ventana de propiedades físicas con el campo propiedades físicas asignadas activado**



Otra característica física muy importante es la fricción y en el siguiente párrafo se explicará cómo asignar fricción a dos componentes y cómo afecta a la simulación el hecho de agregar fricción entre los componentes. La fricción es una fuerza que siempre está presente entre dos superficies; y para *Solidworks,* a menos que sea especificada por el usuario, no será tomada en cuenta para ninguna simulación.

Si existen dos superficies cuya interacción posee un valor significativo de fricción, el cual pueda influir de una manera u otra en los resultados finales de la simulación de un mecanismo, es muy posible que el usuario tenga el deseo de incluir esta fuerza entre ambas superficies, es importante que el usuario tenga en cuenta que el hecho de agregar fricción entre dos superficies prolongará significativamente el tiempo total de ejecución de dicha simulación.

Para agregar fricción entre las superficies de dos piezas distintas primero es necesario agregar un contacto entre ambas superficies. Para lograrlo se debe hacer clic en el ícono llamado contactar que se encuentra en la barra de herramientas del *motion manager,* la figura 57 ilustra este ícono.

Figura 57. **Botón contactar que forma parte del *motion manager***

Una vez hecho el clic sobre el ícono el usuario tendrá en pantalla el menú para configurar las propiedades del contacto entre las superficies seleccionadas, lo primero es seleccionar las caras de ambas piezas sobre las cuales el usuario desea simular el contacto, una vez seleccionadas, ambas superficies serán resaltadas y el usuario podrá especificar los coeficientes de fricción estáticos y dinámicos de dos maneras.

La primera implica utilizar parámetros establecidos de acuerdo a los materiales seleccionados en el menú de propiedades de fricción, se debe aclarar que los materiales para determinar la fricción entre las superficies son distintos a los materiales para determinar la masa de una pieza, dentro de los materiales para determinar la masa de una pieza, el usuario tiene cientos de materiales y aleaciones para seleccionar, pero en el menú de fricción sólo cuenta con unos cuantos materiales simples y comunes así que es importante no confundir los materiales usados para determinar masa con los materiales usados para determinar fricción ya que no son los mismos.

Entonces, cuando se selecciona un material de las opciones disponibles, los espacios para fricción estática y dinámica están deshabilitados ya que dichos coeficientes están preestablecidos de acuerdo al material, esto corresponde a la primera forma de especificar dichos parámetros, pero el usuario también puede especificar sus propios coeficientes; para lograrlo sólo debe desactivar la casilla de especificar material y automáticamente podrá variar las características de la fricción estática y dinámica a su gusto. Las figuras 58 y 59 ilustran ambos casos.

Figura 58. **Menú de propiedades de fricción especificada por el material seleccionado para ambas superficies**

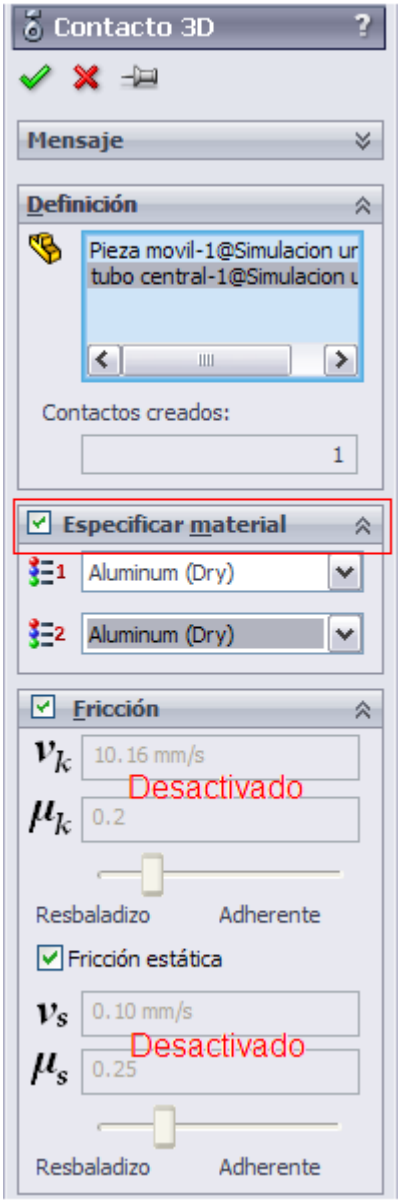
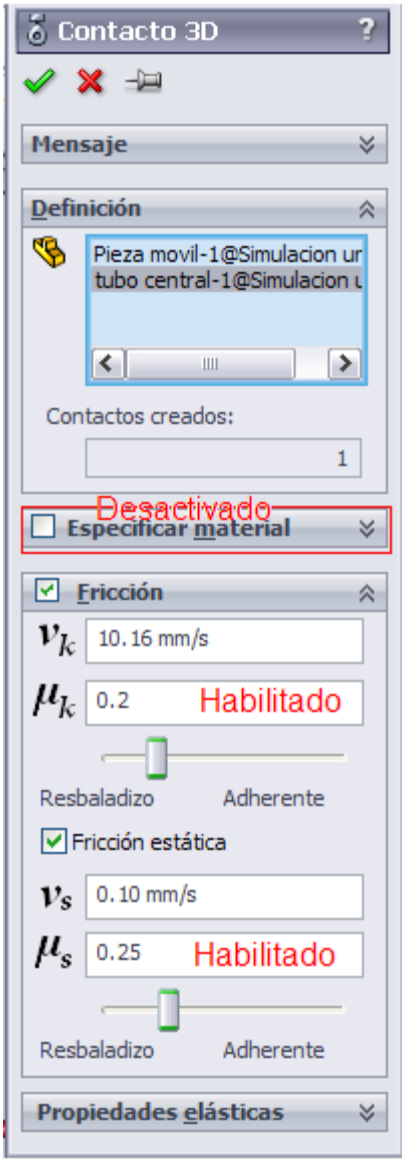


Figura 59. **Menú de propiedades de fricción especificadas por el usuario**



Sobre la fricción es importante mencionar los siguientes aspectos, primero debido a que es una característica del estudio de movimiento los contactos y las fricciones pueden ser agregadas y eliminadas sin modificar el ensamblaje original, basta con colocarse sobre el contacto creado en el árbol de diseño hacer clic sobre él y seleccionar suprimir.

Segundo es importante aclarar, que el hecho de que agregar fricción a una simulación, hará que ésta sea más lenta y más larga debido a que el modelo matemático para un mecanismo con fricción es mucho más complicado que uno sin fricción, por lo que los cálculos para este modelo más complejo llevan más tiempo, y finalmente la fricción permitirá obtener datos más precisos y semejantes a los de un mecanismo real.

Ahora, antes de pasar al siguiente punto, es vital aclarar los conceptos clave acerca de un estudio de movimiento, lo primero es recordar que existen tres tipos de estudio de movimiento y que para los objetivos de esta tesis se utilizará el más completo que es el análisis de movimiento, que está compuesto por un solucionador cinemático que toma en cuenta todas las características físicas del ensamblaje. También se habló sobre motores, uno de los elementos del estudio de movimiento que el usuario debe crear. Lo importante acerca de los motores es que existen dos tipos de motores los rotativos y los lineales y que el tipo de movimiento a seleccionar es el de distancia debido a que es el único que se comunica con *LabVIEW*.

Otro tema que se tocó fueron las propiedades de masa de los componentes, esto es importante ya que entre más exacto y detallista sea el usuario a la hora de definir el ensamblaje, con el mismo nivel de detalle y exactitud obtendrá los datos resultantes de la simulación de su ensamblaje, otro aspecto que aporta para la causa de la exactitud en la simulación del ensamblaje es la fricción, que aporta un toque de realismo a las propiedades y datos obtenido de la simulación; finalmente vale la pena recordar que tanto las propiedades de masa y fricción pueden ser determinadas por dos vías, la primera seleccionando materiales con datos pre configurados y la segunda ingresándolos por medio del teclado.

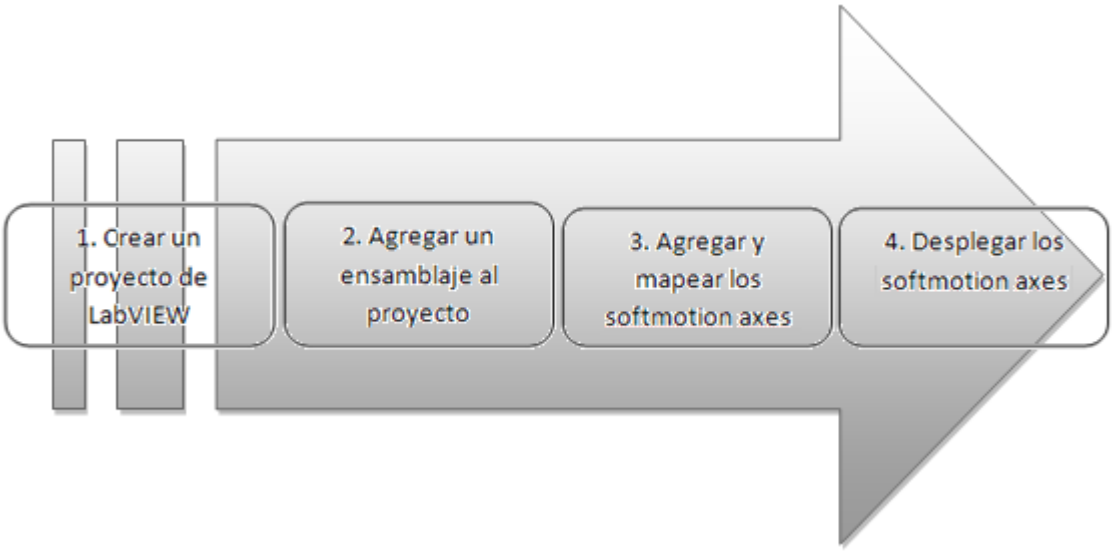
4.2. Conexión básica de *Solidworks* con *LabVIEW*

En esta parte se explicará el tipo de conexión más básica entre *Solidworks* y *LabVIEW,* es tan básica que para realizar la simulación en *Solidworks* controlada desde *LabVIEW* no se necesitará de programación alguna por parte de *LabVIEW*. Una mejor manera de explicar lo que se hará con estos dos programas, es utilizar un panel interactivo propio de *LabVIEW* para realizar pruebas preliminares sobre el mecanismo creado en *Solidworks*

Antes de entrar en detalle con la simulación se dará una breve descripción sobre los requisitos que debe cumplir *LabVIEW* en orden de lograr una simulación exitosa, el usuario necesitará *LabVIEW* 2009, cualquier otra versión del *software* no funcionará, además deberá contar con módulo NI *Softmotion for Solidworks* que está incluido dentro del módulo NI *Softmotion module* 2009.

En orden de construir un vínculo entre ambos programas, observando las cosas desde la perspectiva de LabVIEW, lo primero que se necesita es crear un proyecto de LabVIEW, una vez que el proyecto ha sido creado, podrá moverse al siguiente paso, que es agregar un ensamblaje creado en *Solidworks* al proyecto, con el ensamblaje añadido al proyecto se podrán mapear y agregar ejes de *softmotion*, de acá en adelante llamados *softmotion axes* y finalmente desplegar estos datos. La acción de desplegar los datos permitirá al usuario controlar el ensamblaje de *Solidworks* desde *LabVIEW,* la figura 60 ilustra los pasos.

Figura 60. **Pasos para conectar *Solidworks* con *LabVIEW***



Antes de explicar el primer paso es importante hablar sobre el explorador de proyectos de *LabVIEW* que es una ventana que se muestra cada vez que el usuario crea un proyecto nuevo en *LabVIEW* o cuando se abre un proyecto ya existente.

Para las personas que no están familiarizadas con *LabVIEW* o que nunca han usado *LabVIEW*, el explorador de proyectos permite tener una visión desde la perspectiva de un sistema operativo de una aplicación, contiene todas las partes que componen una aplicación, los diferentes archivos y cualquier cosa que forme parte de un programa será encontrado en el explorador de proyectos de *LabVIEW*, que también ayuda al usuario a controlar cuándo iniciar la simulación del prototipo virtual en *Solidworks* y *LabVIEW*.

Para agregar un nuevo proyecto, lo primero es ejecutar *LabVIEW* 2009, una vez corriendo el programa lo siguiente que se observará es una ventana como la que se muestra en la figura 61, que es la ventana de inicio de *LabVIEW y* dentro de la cual deberá seleccionar la opción *Empty Project*, luego el usuario podrá observar como automáticamente se le presenta una ventana con algunos íconos dentro de ella, esta es la ventana del explorador de proyectos de la que se habló en el párrafo anterior y que se muestra en la figura 62.

Figura 61. **Ventana de inicio de *LabVIEW 2009***

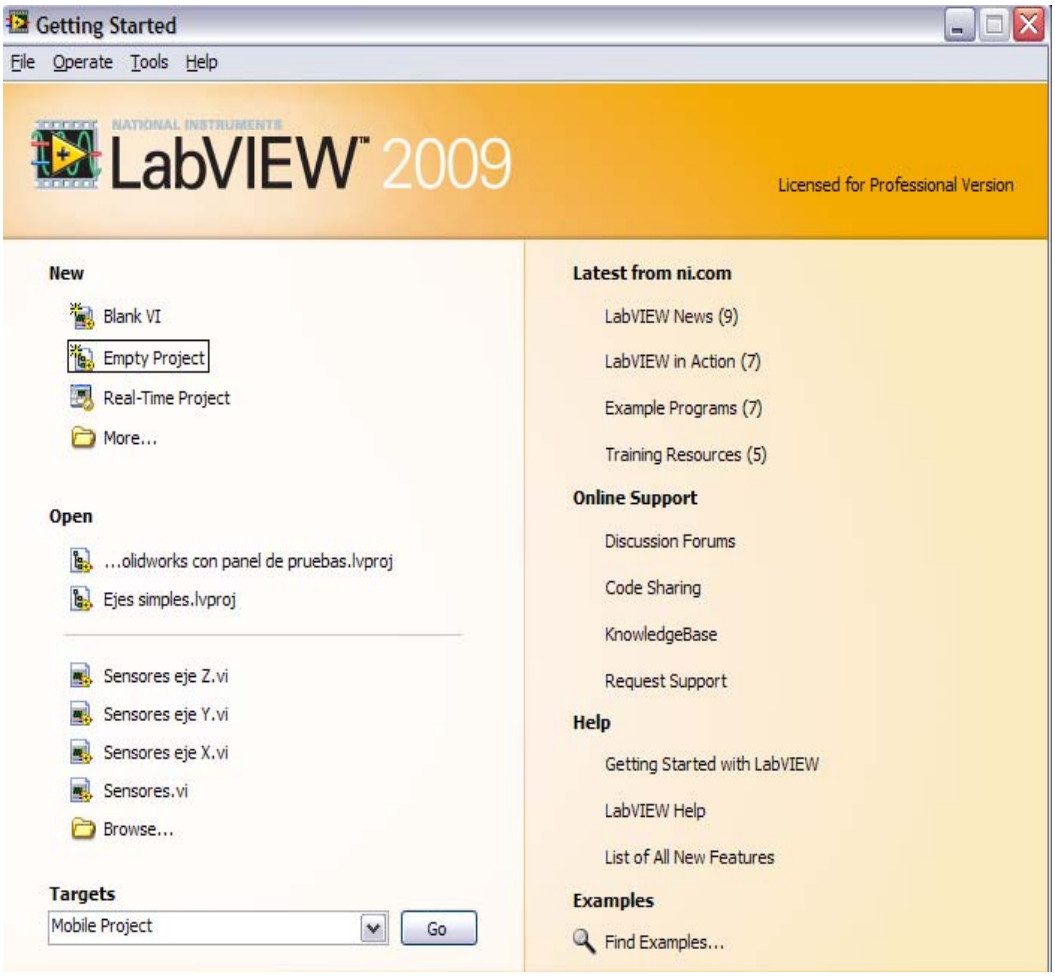
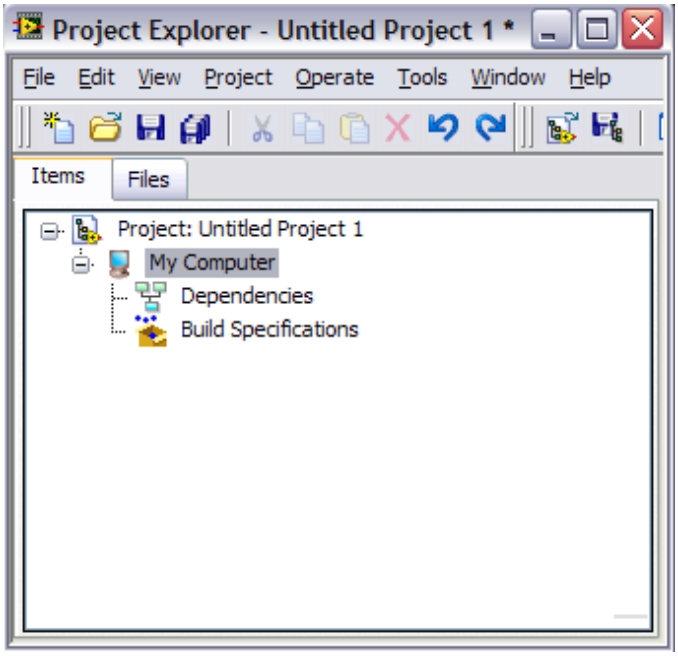
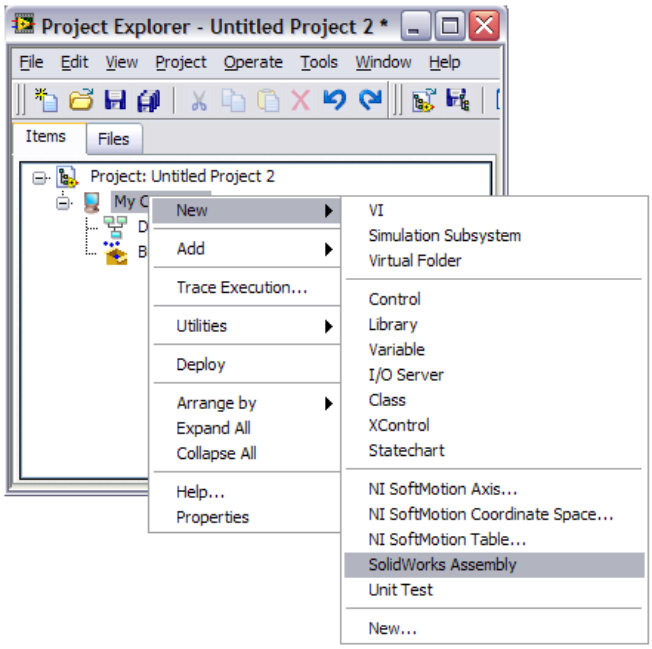


Figura 62. **Explorador de proyectos de *LabVIEW***



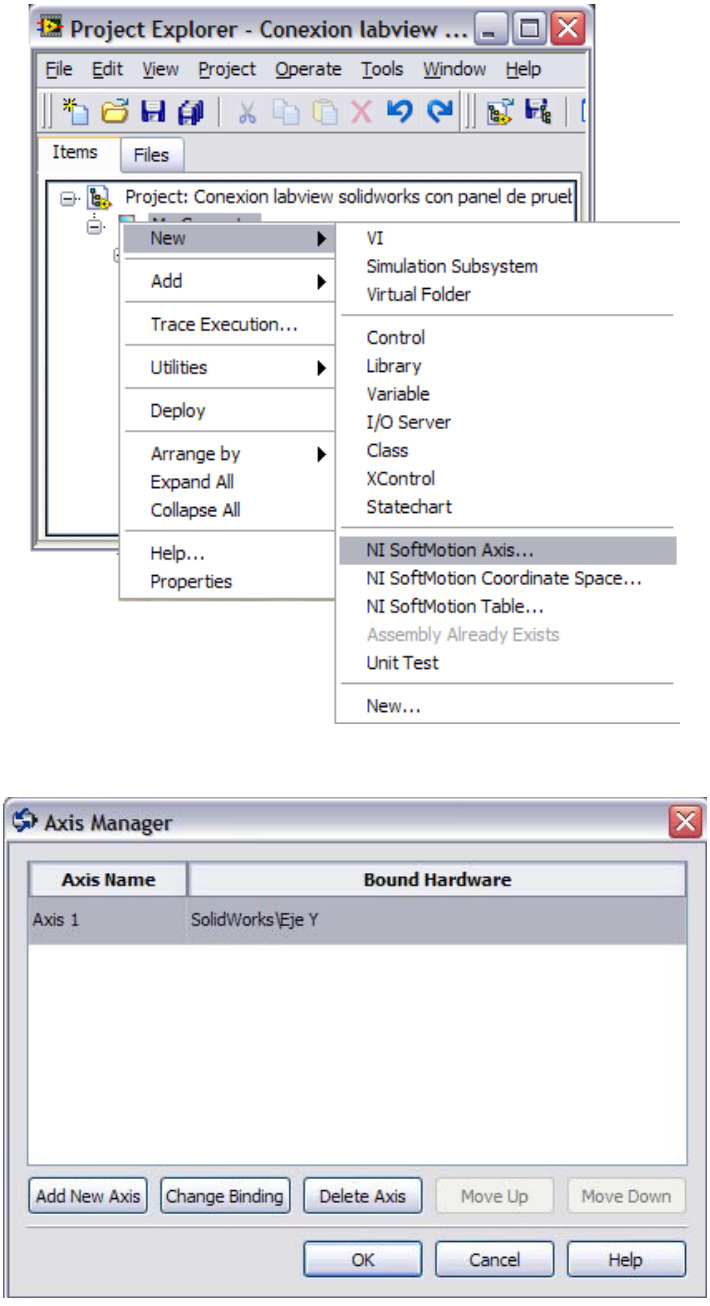
Para agregar un ensamblaje creado en *Solidworks* al nuevo proyecto dentro de la ventana del explorador de proyectos de *LabVIEW* el usuario deberá situarse sobre el ícono de *My computer* dentro del proyecto, hacer clic derecho, seleccionar *New* y buscar el campo *solidworks assembly*; la figura 63 ilustra el proceso. Si el usuario tiene abierto un ensamblaje en *Solidworks* al momento de seleccionar el campo *solidworks assembly,* éste agregará dicho ensamblaje al proyecto de forma automática, de lo contrario pedirá especificar una ruta para ubicar la localización del ensamblaje y podrá importarlo al nuevo proyecto de *LabVIEW*.

Figura 63. **Agregar un ensamblaje de *Solidworks* al proyecto**



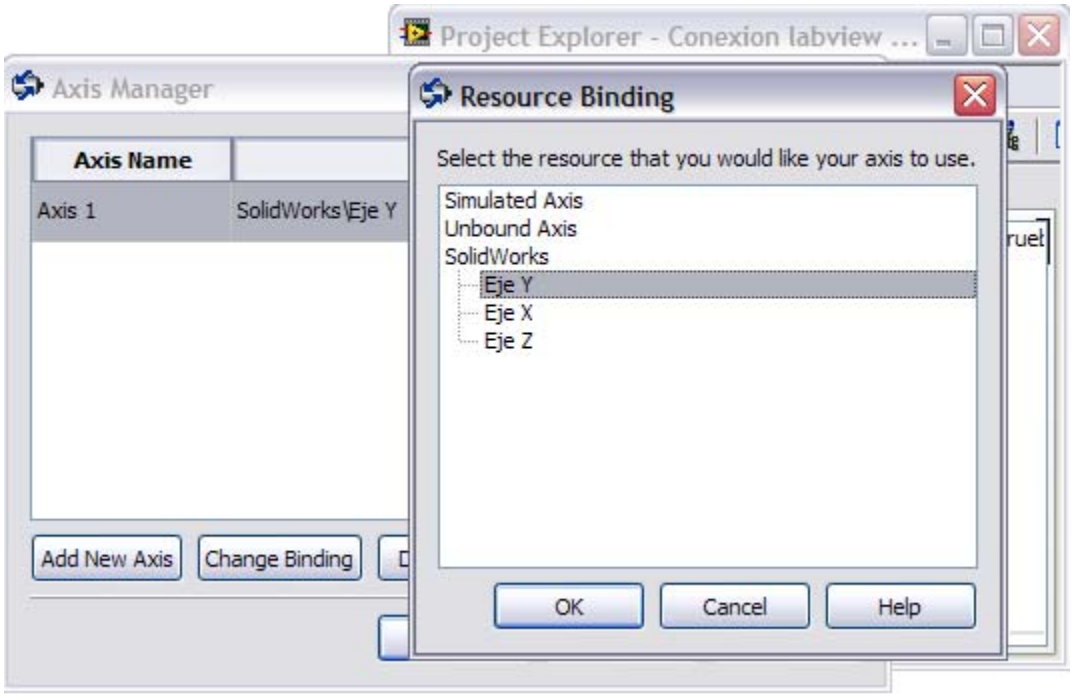
El siguiente paso será agregar el *NI softmotion a*xis al proyecto y el procedimiento es muy similar al anterior, clic derecho sobre *My computer*, luego se sitúa sobre *New*, clic sbore NI *softmotion axis,* por último clic en *Add New Axis* y listo se habrá agregado un eje de *Solidworks* en *LabVIEW*. Una vez que se ha agregado el *softmotion axis* es necesario conectarlo al eje que se desea controlar, para este ejemplo será el eje Y, entonces el explorador de proyectos de *LabVIEW* desplegará una ventana llamada axis manager que indica el eje al que se encuentra ligado el motor en *Solidworks* y al que quedará ligado para controlar desde *LabVIEW*, la figura 64 ilustra los detalles.

Figura 64. **Agregar *softmotion axis* y *axis manager***



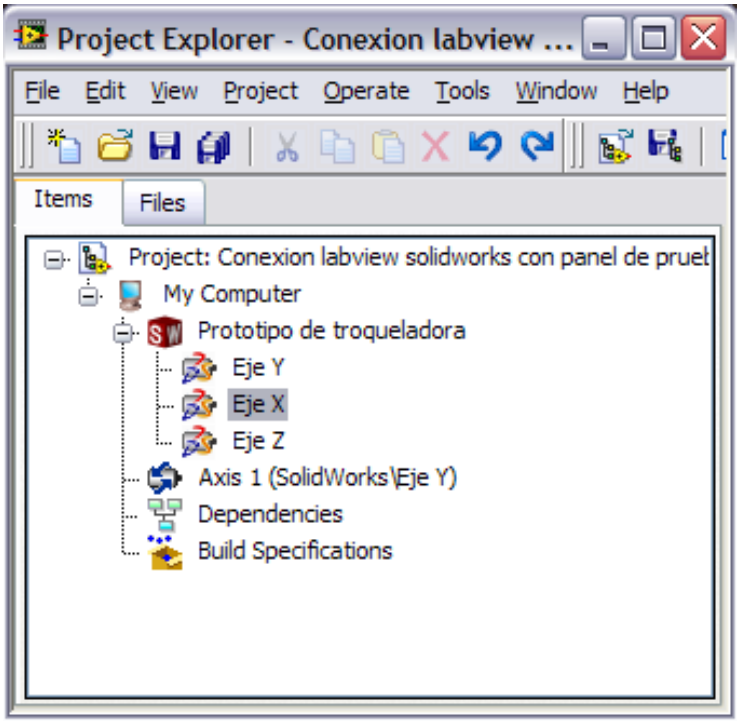
Si por alguna razón el usuario desea ligar el *softmotion axis, Axis* 1 a otro *axis* de movimiento del ensamblaje en *Solidworks*, por ejemplo el eje X, lo único que debe hacer es clic sobre el botón *Change Binding* en la ventana *axis manager* y podrá seleccionar de un listado de los ejes disponibles el eje X y de esta forma podrá controlar con el *Axis* 1 de *LabVIEW* el eje X en *Solidworks,* la figura 65 muestra lo descrito.

Figura 65 **Menú *Resource Binding* del *axis manager***



La figura 66 muestra el proyecto que se ha configurado y que de hecho se encuentra casi listo para realizar una simulación básica con el panel interactivo, dentro del proyecto se observa el ensamblaje llamado prototipo de troqueladora, que por cierto es el ensamblaje que se ha diseñado para simular el control final del prototipo virtual. Además se observan los motores o ejes de *Solidworks* y el *softmotion axis* llamado *Axis* 1.

Figura 66. **Ensamblaje configurado para iniciar simulación en *Solidworks* controlada desde *LabVIEW***



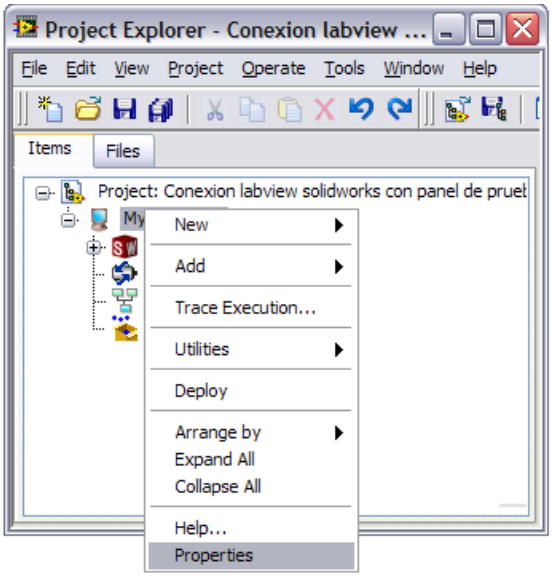
Como se ve en la figura 66, el ensamblaje tiene ligados tres motores o ejes, esto debido a que el ensamblaje fue creado con tres ejes de movimiento, si tuviera sólo uno observaríamos sólo un motor y si tuviera diez ejes de movimiento veríamos diez motores.

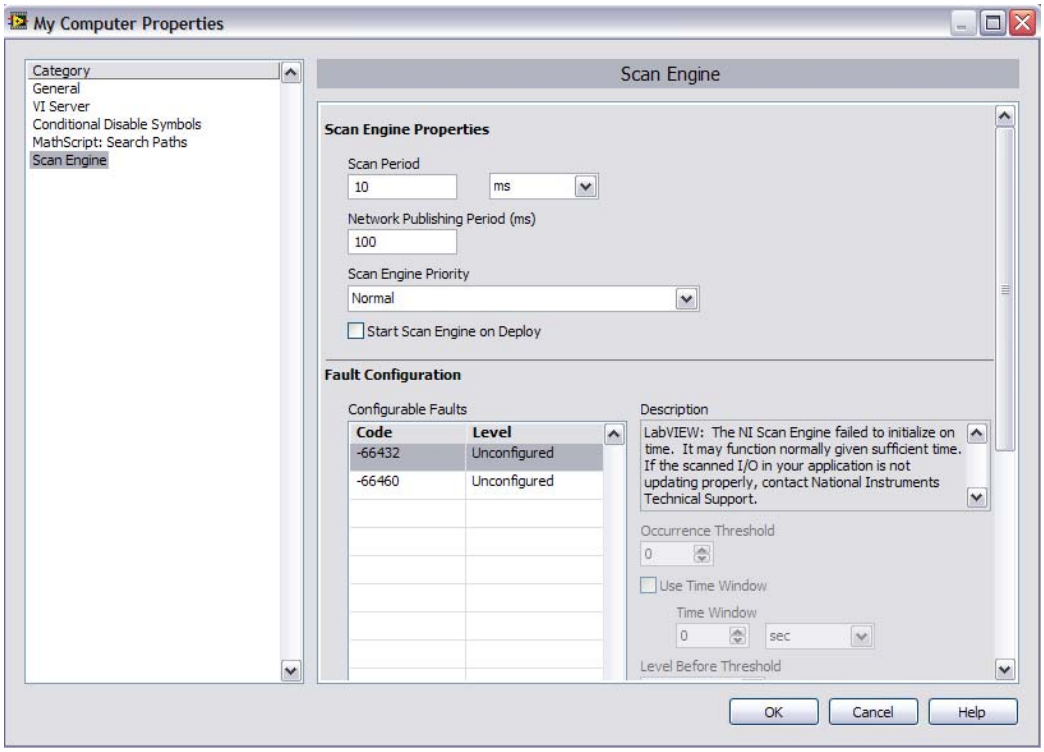
Los motores son importados automáticamente una vez que hemos agregado el ensamblaje al proyecto, luego de los ejes del ensamblaje se puede ver el *softmotion axis, Axis* 1, que se encuentra ligado al motor que representa el eje Y en Solidworks y que como se ha mencionado antes es el recurso que permite a *LabVIEW* poder controlar el eje Y dentro del ambiente de simulación de *Solidworks.*

El último paso para completar la configuración de un proyecto de *LabVIEW* capaz de controlar una simulación en *Solidworks* es habilitar el *scan engine* a la hora de realizar el despliegue de datos. El *scan engine* es la parte que termina de cerrar la conexión entre *Solidworks* y *LabVIEW*, permite cerrar la conexión que se inició añadiendo el ensamblaje al proyecto, luego agregando el *softmotion axis* y ligándolo a un eje específico del ensamblaje. Si no se habilita el *scan engine* no se podrá realizar el control del ensamblaje desde *LabVIEW*.

En orden de habilitar el *escan engine* cuando se desplieguen los datos, el usuario deberá situarse sobre el ícono de *My computer*, hacer clic derecho y seleccionar el campo *Properties* (propiedades), una vez hecho esto el usuario observará la ventana de propiedades de *My computer*, en la cual buscará el campo *scan engine* y hará clic sobre él, lo que desplegará del lado derecho de la ventana las propiedades del *scan engine*, la figura 67 muestra los pasos.

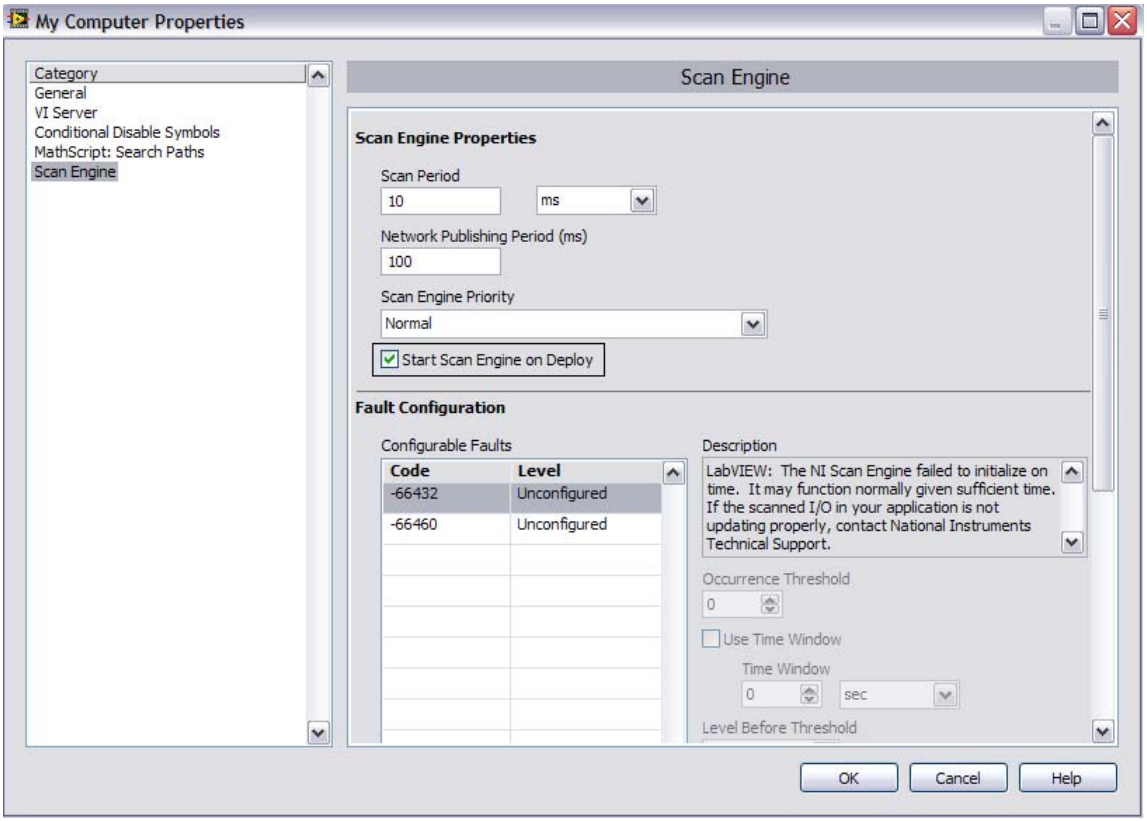
Figura 67. **Ventana de propiedades de *My computer***





Una vez que el usuario tiene a la vista las propiedades del *scan engine*, lo que debe hacer es buscar el campo llamado *start scan engine on deploy* y habilitarlo haciendo clic en el espacio en blanco a la izquierda del mismo y haciendo clic sobre el botón *OK*, el procedimiento lo ilustra la figura 68. En este punto el proyecto creado se encuentra completamente configurado para iniciar la simulación conjunta entre *Solidworks* y *LabVIEW.*

Figura 68. **Campo *start scan engine on deploy* habilitado**



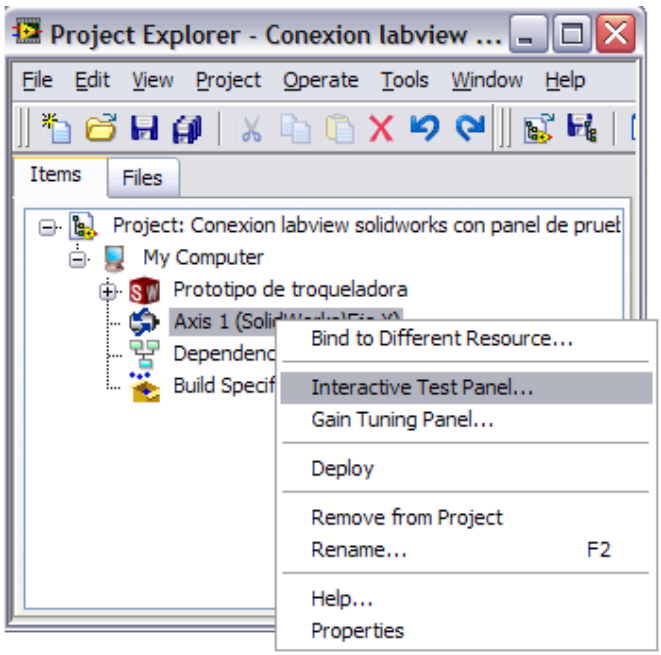
A continuación se mostrará el panel de pruebas interactivo (*Interactive Test Panel*), que es una utilidad de *LabVIEW* para controlar el prototipo creado en *Solidworks* desde *LabVIEW* y con la cual no hay necesidad alguna de programar en el lenguaje gráfico de *LabVIEW*.

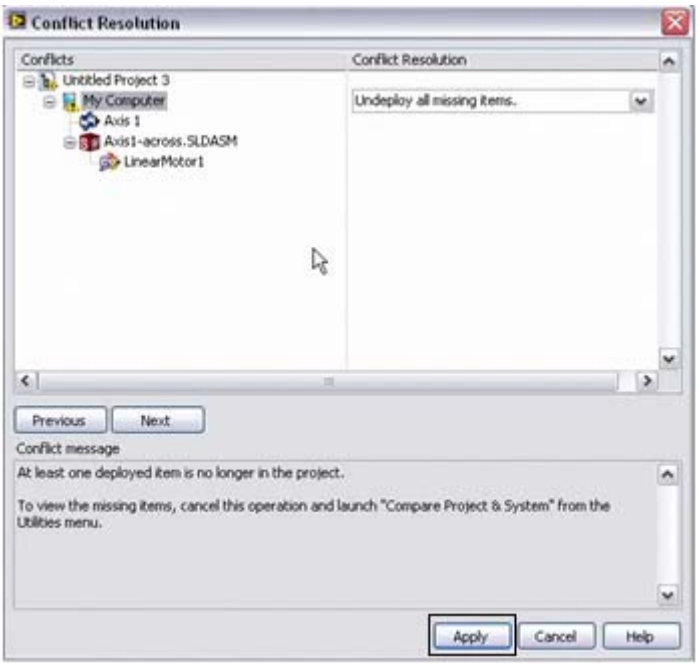
Lo que se intenta hacer con el panel interactivo es mostrar al usuario cómo controlar el movimiento básico de un eje del ensamblaje con *LabVIEW* y para alcanzar este objetivo se debe hacer lo siguiente: primero, dentro del proyecto nos situamos sobre el *Axis* 1, damos clic derecho sobre él y seleccionamos el campo *interactive test panel.*

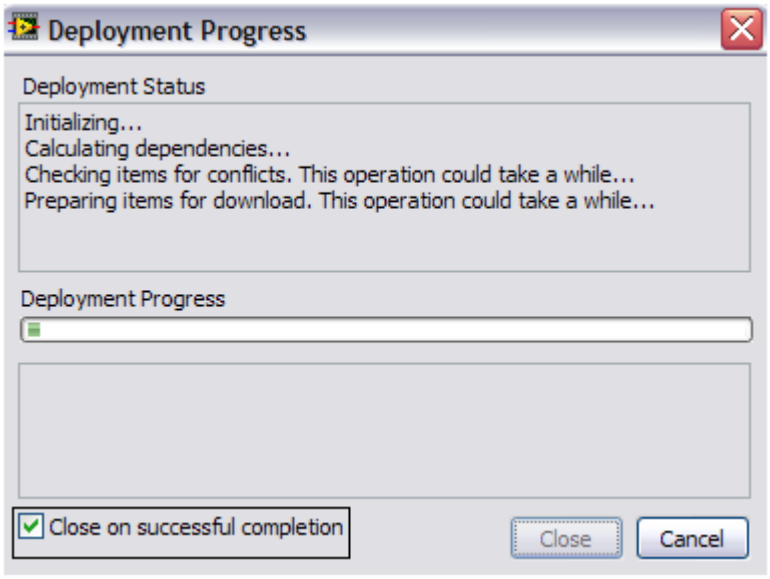
Cuando el usuario haya realizado lo indicado *LabVIEW* mostrará la ventana del progreso del despliegue (*deploy progress*) y una ventana con una adevertencia, la cual se corrige haciendo clic en el botón *Apply*, una vez que el proceso de despliegue ha concluido el usuario verá en su monitor el *interactive test panel*.

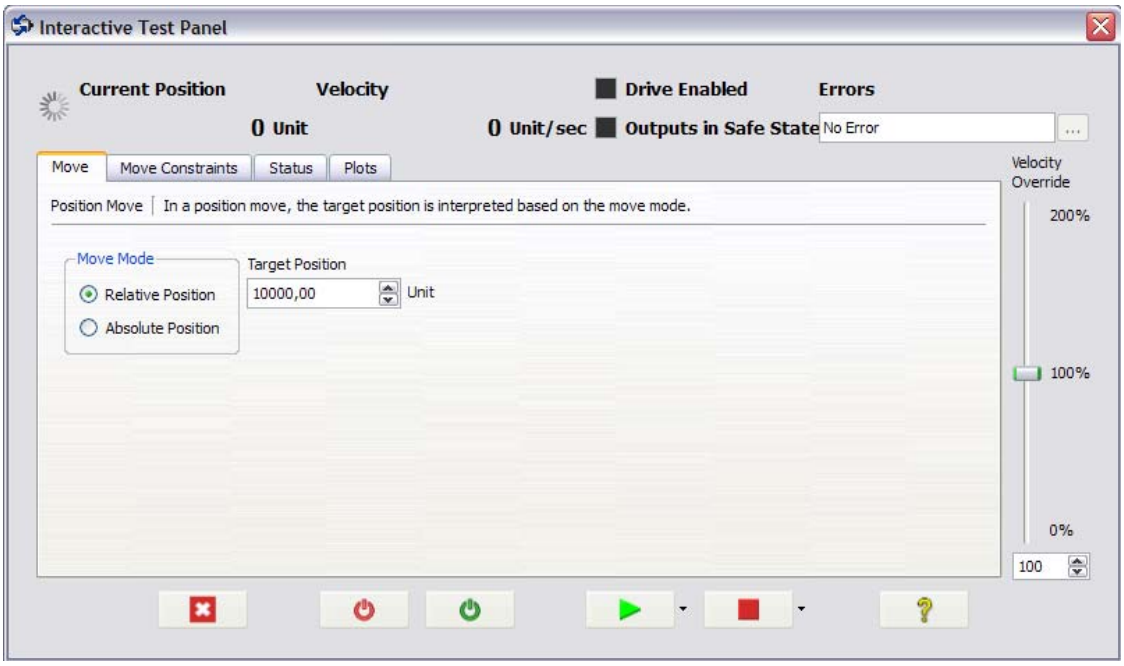
Una función importante presente en la ventana del *deploy progress* es la que se encuentra en la parte inferior izquierda de la ventana, dicha función se identifica como *close on successful completion*, que cierra esta ventana automáticamente siempre y cuando el despliegue de datos se realizó de una forma satisfactoria, una forma de verificar que todo se realizó sin problema alguno; para activarla basta con activar la casilla en blanco que se muestra en la figura 69 y si no se encuentra habilitada dicha casilla el usuario tendrá que cerrar la ventana manualmente y le será más difícil detectar cualquier error.

Figura 69. **Procedimiento para activar la utilidad *Interactive Test Panel***




Conflict Resolution
Conflicts
Conflict Resolution
Untitled Project 3
My Computer
Axis 1
Axis1-across.SLDASM
LinearMotor1
Undeploy all missing items.
Previous
Next
Conflict message
At least one deployed item is no longer in the project.
To view the missing items, cancel this operation and launch "Compare Project & System" from the Utilities menu.
Apply
Cancel
Help


Deployment Progress
Deployment Status
Initializing...
Calculating dependencies...
Checking items for conflicts. This operation could take a while...
Preparing items for download. This operation could take a while...
Deployment Progress
Close on successful completion
Close
Cancel

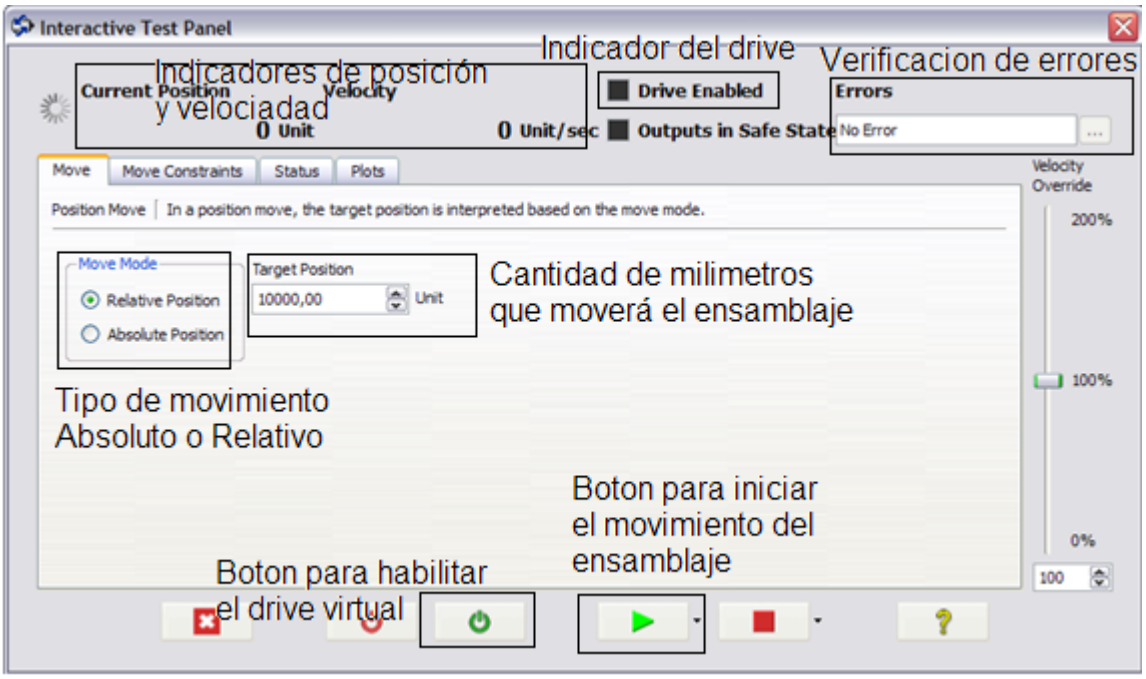


Es importante dejar claro lo siguiente, para lograr controlar una simulación en *Solidworks* desde *LabVIEW*, naturalmente *Solidworks* debe de estar en ejecución, ya que de otro modo el usuario se topará con una serie de errores y advertencias que no le permitirán hacer absolutamente nada. La importancia de que *Solidworks* se esté ejecutando al mismo tiempo que *LabVIEW,* es porque al momento que desde *LabVIEW* se ordene la ejecución de un movimiento cualquiera, dicha acción se verá reflejada moviendo ejes dentro de *Solidworks*.

El panel interactivo asistirá al usuario para controlar el movimiento del eje al que se encuentre ligado el *Axis* 1 dentro del proyecto. Cada vez que el usuario ordene la ejecución de un movimiento, dicho movimiento podrá ser observado en el estudio de movimiento que se encuentra corriendo en *Solidworks*, *LabVIEW* no presenta ningún tipo de simulación y se concentra únicamente en presentar en pantalla el control para el ensamblaje al que ha sido conectado.

Ahora se explicará cómo utilizar el *Interactive Test Panel*, una vez que el usuario lo tiene en pantalla, observará que cuenta con varios botones e indicadores, éstos ayudarán al usuario a controlar el prototipo virtual que se encuentra abierto y corriendo es *Solidworks*. El primer paso es verificar que no hay ningún tipo de error, seguido el usuario deberá habilitar el drive virtual (*Axis* 1) para que éste pueda controlar el movimiento del motor y que posee su indicador, a continuación el usuario debe seleccionar el tipo de movimiento absoluto o relativo, de lo que se darán más detalles posteriormente, ingresar la cantidad de milímetros que desea mover el ensamblaje y finalmente presionar le botón de inicio, la figura 70 muestra las partes de dicho panel.

Figura 70. **Partes del *Interactive Test Panel***



Suponga, por ejemplo que se quiere desplazar el eje Y del ensamblaje 200 milímetros, hacia la izquierda (posición positiva del eje Y), deberá de hacer lo siguiente. Primero verificar que no hay errores, en seguida clic sobre el botón para habilitar el drive, escoger tipo de movimiento, para este ejemplo funciona el movimiento relativo, en el campo *taget position* ingresar 200 y finalmente presionar el botón de *play*. Si ahora el usuario se coloca sobre el ensamblaje de *Solidworks* (prototipo de troqueladora) agregado al proyecto verá como el eje Y, que mueve horizontalmente el punzón, se desplaza 200 milimetros a la derecha.

Además el usuario podrá monitorear la velocidad y distancia recorridas por el eje bajo control, ya que el panel cuenta con indicadores para estos parámetros en la parte superior del mismo; también podrá realizar movimientos en la dirección negativa del eje Y (hacia la derecha), bastará con que el usuario en el campo de *target position* ingrese la cantidad de -200 milímetros en lugar de 200 y al ordenar la ejecución del movimiento y situarse sobre el prototipo de la troqueladora verá como regresa sobre el primer recorrido de los 200 milímetros.

El panel cuenta con otras características como control sobre la velocidad, aceleración y desaceleración de la simulación antes de ejecutarla y un espacio que presenta una gráfica distancia vs tiempo en la que se puede observar, en un espacio de dos dimensiones, la trayectoria que ha seguido el eje controlado.

4.3. Conexión entre *LabVIEW* y *Solidworks* utilizando NI *softmotion*

Hasta el momento se ha explicado cómo preparar un ensamblaje en *Solidworks* de manera adecuada y cómo realizar la conexión entre *Solidworks* y *LabVIEW* de la forma más sencilla posible y sin realizar programación alguna dentro de *LabVIEW*. Ahora se busca brindar una herramienta más al usuario, explicando las características más importantes del módulo NI *softmotion*, que ayudaran al usuario a programar aplicaciones en *LabVIEW* para controlar, casi cualquier tipo de movimiento que pase por la mente del usuario a la hora de realizar la simulación de mecanismos en *Solidworks.*

Como se explicó en el capítulo 3 el módulo NI *softmotion* es una herramienta para desarrollo gráfico en aplicaciones personalizadas de control de movimiento, además el módulo cuenta con una gran cantidad de utilidades como funciones para crear distintos tipos de movimientos como líneas rectas, arcos y contornos, también posee bloques para funciones avanzadas como engranaje y levas electrónicas. Pero a pesar que posee muchas características de gran relevancia se expondrán, únicamente, los aspectos útiles para cumplir los objetivos de este trabajo de graduación.

El principal aspecto, es la capacidad con que cuenta este módulo para la generación de prototipos virtuales en aplicaciones de control de movimiento y diseño de máquinas, que permite enlazar las funciones propias del módulo con *Solidworks,* permitiendo ofrecer aplicaciones que, operando de manera conjunta, puedan simular los diseños creados en *Solidworks* usando trayectorias de movimiento desarrollados con bloques de función NI *softmotion* y permitiendo al usuario visualizar, optimizar y evaluar diversos aspectos del diseño antes de gastar en prototipos físicos.

4.3.1. Ciclo temporizado (*Timed loop*)

Antes que nada es importante refrescar lo que ya se ha explicado, primero el explorador de proyectos es la vista de un proyecto desde la perspectiva de un sistema operativo, acá es donde se encuentran todos los componentes de la aplicación que el usuario desea programar en *LabVIEW*.

Para el caso de una simulación entre *LabVIEW* y *Solidworks* encontraremos el ensamblaje de *Solidworks*, seguido de los motores parte de dicho ensamblaje y que son agregados automáticamente al agregar el ensamblaje, luego los *softmotion axes* que deben ser ligados a los motores de *Solidworks* de manera que puedan ser controlados desde *LabVIEW* y el último detalle importante sobre la ventana del *project explorer* es que desde ella, el usuario puede controlar cuando iniciar la simulación en *Solidwork.* Ésta última característica tomará sentido más adelante cuando explique cómo cambiar el estado de la simulación entre el estado activo y el de configuración.

Se explicará a continuación una parte importantísima del trabajo de graduación, programar movimiento con *LabVIEW,* y en orden de hacerlo de la forma más clara posible, lo primero que se hará es dar una breve introducción sobre el ambiente de *LabVIEW*.

LabVIEW es un programa de alto rendimiento, que cuenta con un lenguaje de programación gráfico con utilidades configuradas específicamente para mediciones y aplicaciones industriales. Con *LabVIEW* el usuario rápidamente puede ligar sus aplicaciones a equipos, analizar datos y desplegar resultados, además ofrece cientos de funciones interconstruidas para entrada y salida de datos, visión, movimiento y control para implementar de manera rápida aplicaciones personalizadas, las cuales pueden ser descargadas a sistemas embebidos para un funcionamiento más rápido y efectivo.

Las funciones creadas en *LabVIEW* se llaman *VIs* (*Virtual instruments*) y están compuestos por dos partes el panel frontal (*front panel*) y el diagrama de bloques (*block diagram*). El *front panel* será la interfaz de usuario, es lo que el usuario verá en pantalla y es por medio del cual podrá interactuar con la aplicación. El *front panel* contiene controles e indicadores, los controles son las entradas del programa y los indicadores son las salidas del programa.

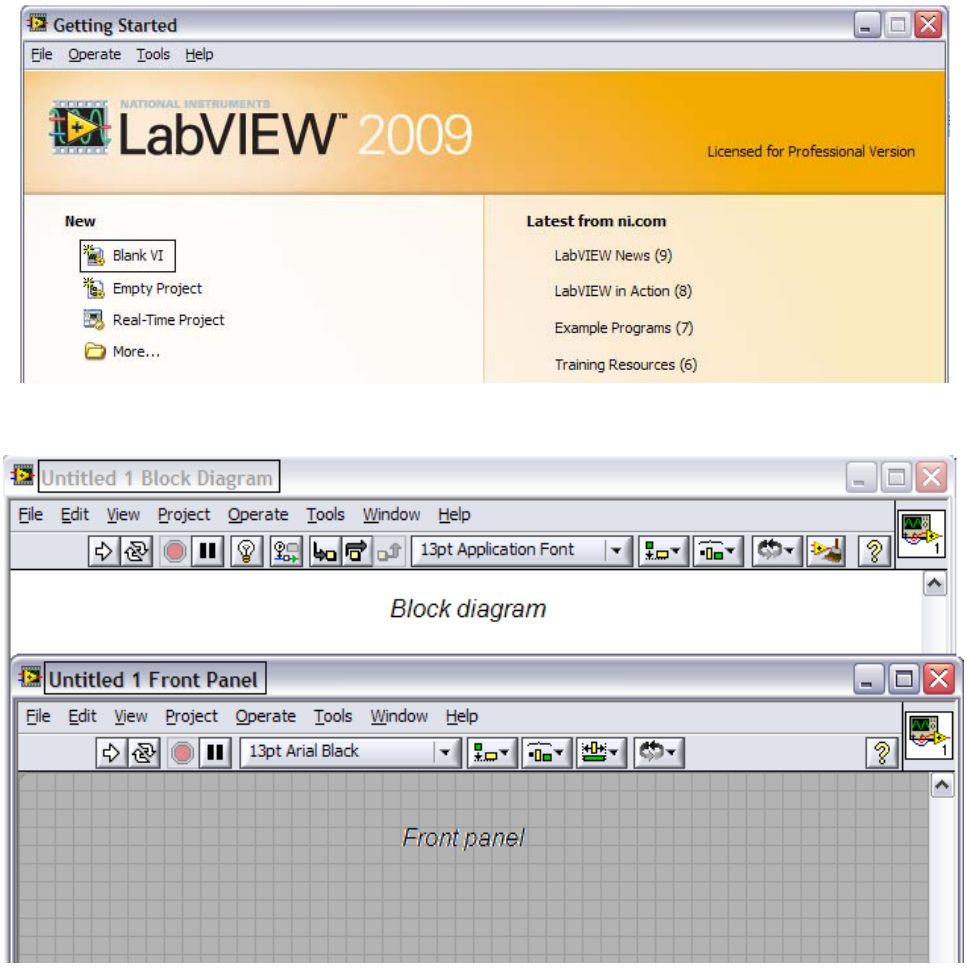
El *block diagram* es el segundo elemento del VI, el *block diagram* es el código, es donde se desarrolla la programación gráfica. Dentro del *block diagram* existen terminales para los controles e indicadores y es por medio de estas terminales, cómo se realiza el intercambio de datos entre este bloque y el *front panel.*

Es importante que el usuario tenga en cuenta que a pesar de que se exponen los conceptos básicos con respecto a la programación de movimiento en *LabVIEW* es importante estar familiarizado con el software, así que se exhorta a todo aquel que tenga deseos a investigar más sobre qué es y cómo se usa *LabVIEW* que lo haga ya que es un programa muy extenso y es difícil que exista un “todologo” que sepa utilizar todas las funciones de dicho *software*.

Una vez claros los puntos anteriores se pasa al siguiente nivel, la programación de aplicaciones de movimiento con *LabVIEW.* Lo primero es mostrar los ciclos de tiempo (*timed loops*). Un *timed loop* es como un ciclo mientras (*while loop*) que se utiliza para ejecutar un programa de manera continua. Para el caso particular del *timed loop*, este ciclo aparte de ejecutar un programa de manera continua, cuenta con características internas de sincronización que son las que utilizará el usuario para comunicarlo con *Solidworks* por medio del *scan engine*.

Es importante que el usuario tenga en cuenta que ya que se encuentra programando una aplicación en *LabVIEW*, debe de estar trabajando sobre el *block diagram* de su VI. Para crear un nuevo VI el usuario debe ejecutar *LabVIEW* y en la pantalla de inicio debe escoger el campo *Blank* VI para luego poder observar en pantalla dos ventanas, una correspondiente al *front panel* y otra correspondiente al *block diagram* del nuevo VI, la figura 71 ilustra

Figura 71. **Creación de un nuevo VI**



Para colocar un *timed loop* en el *block diagram* el usuario deberá hacer clic derecho dentro del *block diagram*, buscar el campo *programming*, *structures*, *timed structures* y por ultimo seleccionar *timed loop*, como lo ilustra la figura 72. Luego de haber realizado el proceso anterior y haber creado un cuadro con el botón izquierdo del ratón presionado; verá en su *block diagram* un marco azul con unos campos a la izquierda como el que se muestra en la figura 73.

Figura 72. **Localización del *timed loop***

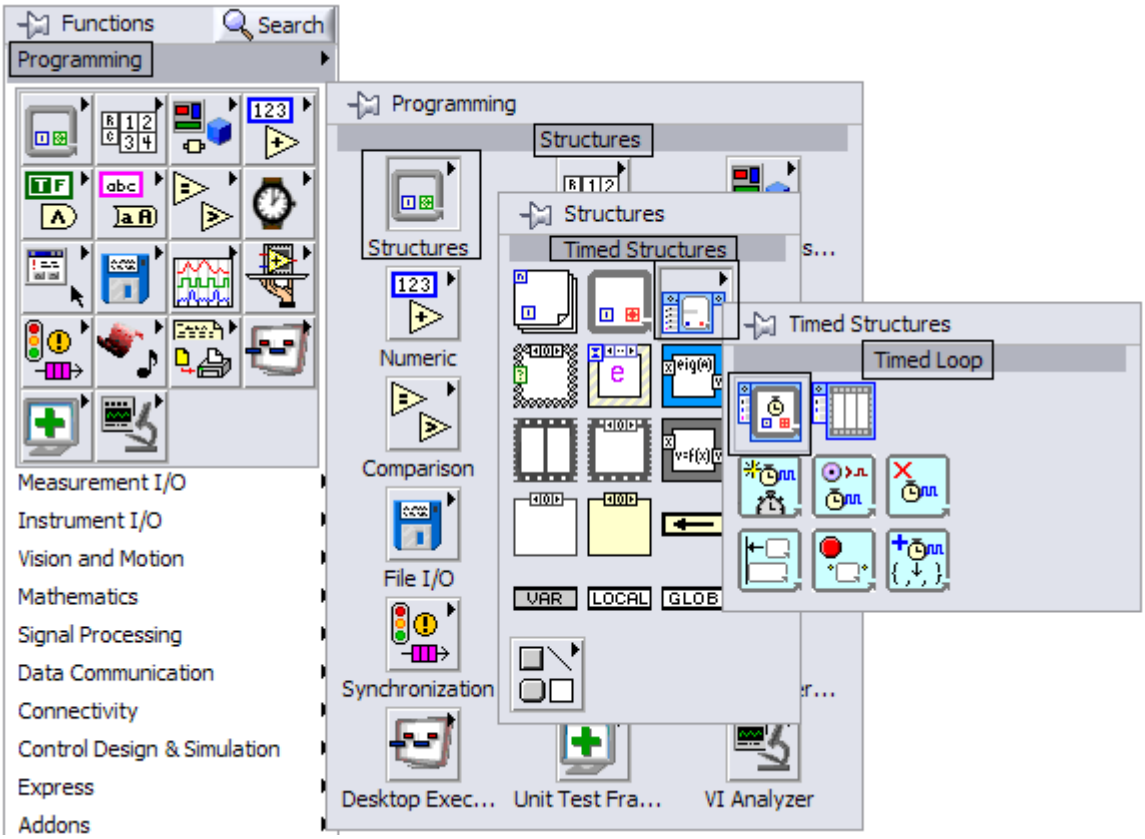
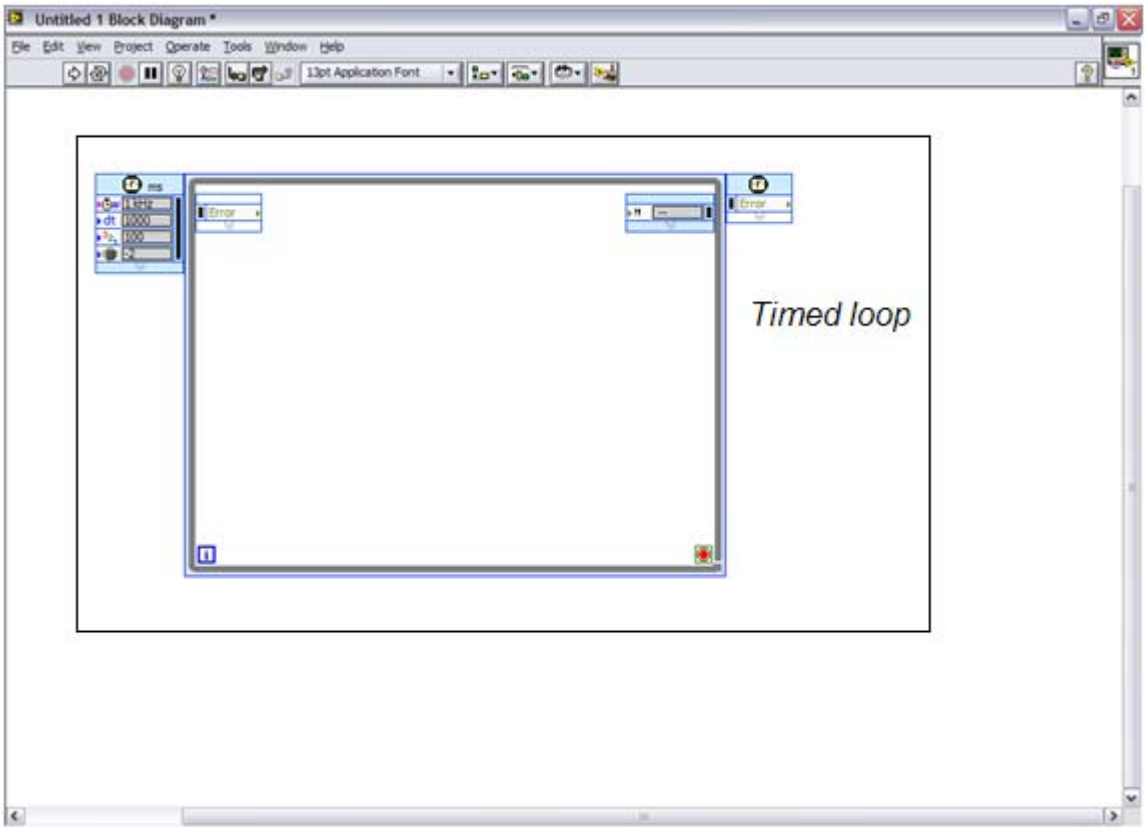


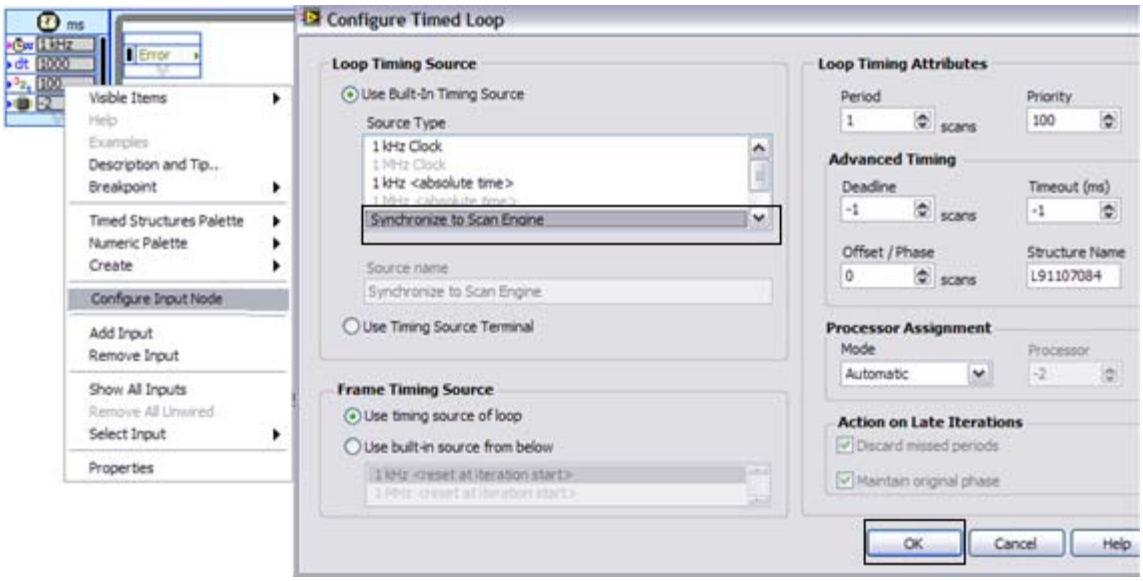
Figura 73. ***Timed loop* dentro del *block diagram***



Ahora, es dentro del *timed loop* donde el usuario deberá programar su aplicación, ya que esto garantizará que su código se ejecutará de manera continua hasta que por algún motivo en especial o simplemente porque así se programó la función de paro con que cuenta este ciclo cambie su estado de falso a verdadero y este suceso detenga la ejecución del código dentro del *timed loop*.

Para poder ejecutar el código dentro de un *timed loop,* primero se debe configurar el ciclo para que pueda comunicarse con *Solidworks*, lo que se logra sincronizándolo con el *scan engine*. La sincronización del *timed loop* con el *scan engine* se hace configurando el nodo de entrada (*input node*), este nodo es la parte que se encuentra del lado izquierdo del ciclo, y se configura haciendo clic derecho sobre él, seleccionando el campo *configure input node* y por último seleccionando el campo *syncronize to scan engine* de la lista *source type* y haciendo clic en *OK* para guardar los cambios tal y como lo muestra la figura 74, esto asegura que cada vez que se ejecute un ciclo del *timed loop* se encuentre sincronizado con cada ciclo de ejecución del *scan engine*.

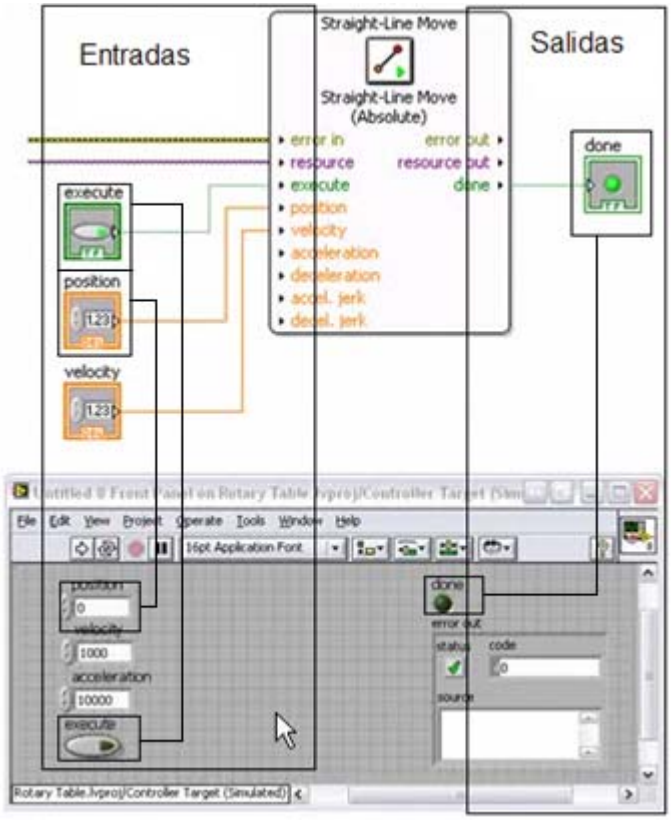
Figura 74. **Configuración del *input node***



4.3.2. Funciones de movimiento del módulo *softmotion*

LabVIEW cuenta con varias funciones de movimiento interconstruidas, cada función de movimiento está diseñada para realizar una tarea de movimiento específica. Todas las funciones de movimiento cuentan con entradas y salidas, como puede observar en la figura 75.

Figura 75. **Entradas y salidas de un bloque de movimiento**



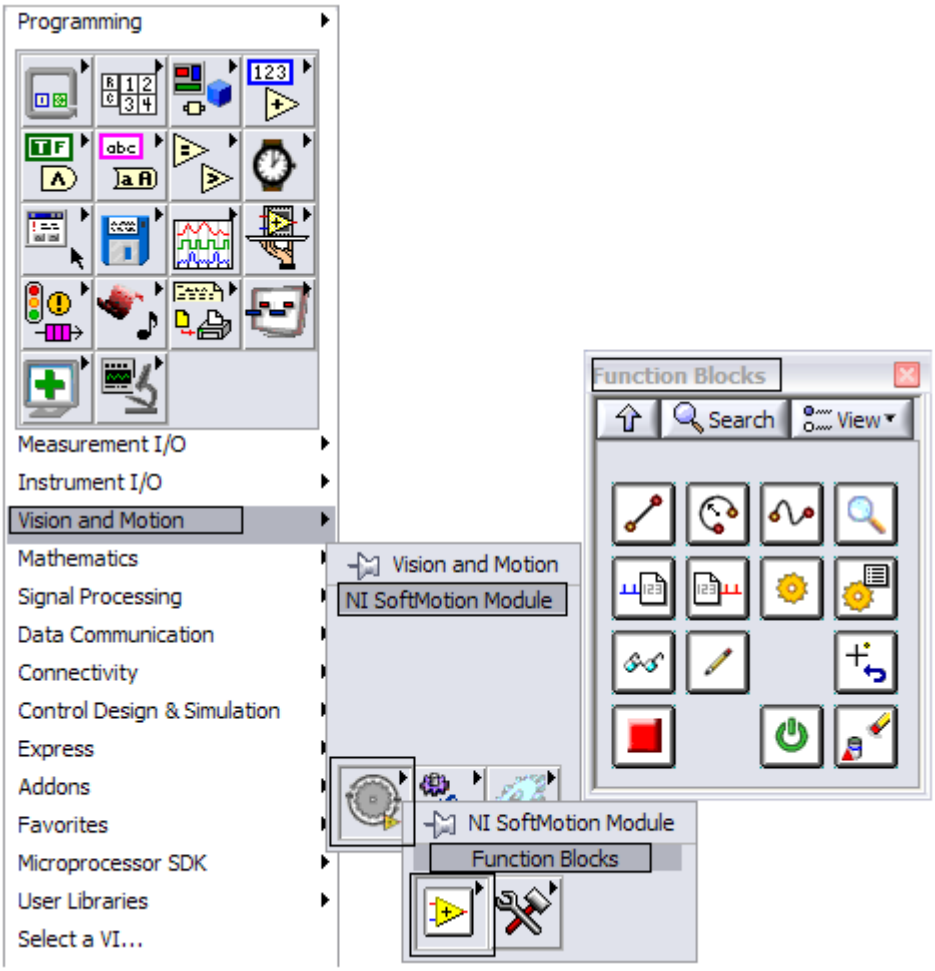
Las entradas se encuentran del lado izquierdo de la función y a ellas pueden ser ligadas controles o bien valores constantes. Los controles permitirán al usuario modificar los parámetros de la función, desde el *front panel* de *LabVIEW*, ya que los controles se ven reflejados en dicho espacio.

Por ejemplo en la figura anterior se puede observar un botón con la etiqueta *execute* en el *front panel*, que corresponde al control *execute* ligado a la función en el *block diagram*. Por el otro lado si utiliza constantes para especificar los parámetros requeridos por la función, el usuario no tendrá la capacidad de controlarlos desde el *front panel* debido a que las constantes no se ven reflejadas en dicho espacio.

Las salidas que se encuentran del lado derecho de las funciones de movimiento se comportan de manera similar que los controles, sus diferencias radican en que a las salidas sólo pueden ser ligados indicadores, nunca controles, y dichos indicadores también se ven reflejados en el *front panel,* utilizando de nuevo el recurso de la imagen 75 se puede ver cómo la salida con la etiqueta *done* en el *block diagram* está ligada a un indicador *booleano*, que se representa como un led indicador con la misma etiqueta, *done*, el *front panel* de *LabVIEW*.

Las funciones de movimiento pueden ser encontradas, estando en el *block diagram*, siguiendo los siguientes pasos, clic derecho sobre el block diagram, clic sobre *vision and motion*, NI *softmotion module* y finalmente *function blocks*, que es la paleta que se muestra más hacia la derecha en la figura 76.

Figura 76. ***NI softmotion function blocks***

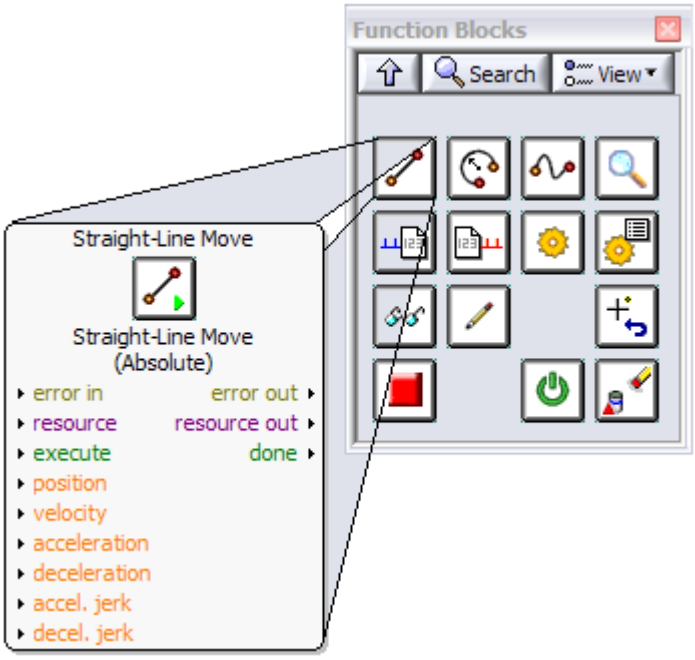


Como se mencionó, cada función de movimiento realiza una tarea específica, pero también se pueden conectar, como se verá más adelante, múltiples bloques de movimiento de diferentes maneras para crear trayectorias de movimiento complejas.

4.3.2.1. Función para trazar una línea recta

La primera función en detallar será la función para una línea recta (*straight line move function*), dicha función realiza una línea recta sobre un eje o bien sobre un recurso coordinado (sistema de ejes). Un eje o *axis* como ya se sabe es simplemente un motor ligado al ensamblaje de *Solidworks* al momento de agregarlo al proyecto, la figura 77 ilustra la función.

Figura 77. **Función para hacer una línea recta**



Como también se explicó existen motores rotativos y motores lineales; ésta función se comporta casi de la misma manera interactuando con un motor lineal que con uno rotativo, la única diferencia radica en las unidades que el usuario provee a la función, para un motor rotativo la terminal con el nombre *position*, indicará a la función cuantos grados debe rotar, por ejemplo si el usuario desea que un motor rotativo ejecute un cuarto de vuelta, el valor ingresado en la terminal de posición será 90, que equivale a los 90 grados de un cuarto de vuelta. La terminal velocidad para el caso del motor rotativo indicará el número de grados por segundo.

En el caso de los motores lineales todo lucirá exactamente igual, pero la terminal *position* ya no hará referencia a grados, ahora indicará los milímetros que el usuario desea que se desplace el eje. Por ejemplo si el usuario desea ejecutar un movimiento en línea recta de 10 centímetros, deberá ingresar el valor 100 que representa 100 milímetros en el espacio *position* en el *front panel* correspondiente a la terminal *position* presente en el *block diagram*.

En la figura 77, el usuario observará varias entradas y salidas, empecemos con la primera de las entradas *error in*, esta es una entrada para manejar errores durante la ejecución de la función, cualquier error que ocurra lo mostrará por medio del indicador *error out*, luego tenemos *resource*, a esta entrada se debe ligar el *softmotion axes* del eje que se desea controlar, por ejemplo si es el eje Y, el usuario se dirige al *Project explorer*, ubica el *Axis* 1, lo arrastra hasta el *block diagram* de la aplicación y lo suelta, automáticamente observará un campo color morado llamado *Axis* 1 que es el que debe ligar a la entrada *resource*.

A continuación se ve la entrada *execute* que es la encargada de indicar a la función cuando iniciar la ejecución del movimiento, un detalle importante de esta terminal, es que representa un control *booleano*. Un control *booleano* funciona enviando flancos positivos a la terminal conectada cuando éste cambia su estado de falso a verdadero y hay dos formas de obtener el flanco positivo, una es activando el control *booleano* y la otra es que dicho flanco positivo provenga de una función previa, acá se vuelve importante entender cómo funciona el indicador *done,* que entrega un flanco positivo al momento en que la función ha terminado de ejecutar su labor y que de hecho es una forma de conectar dos funciones en secuencia.

También están las entradas para posición, velocidad, aceleración y desaceleración que se conectan a controles que permiten ingresar los datos necesarios para que la función realice un trazo personalizado. La única entrada, de estas cuatro, que no puede quedar sin estar conectadas a un control es la de posición, las tres restantes si no se conectan a algún control poseen valores predeterminados para velocidad, aceleración y desaceleración.

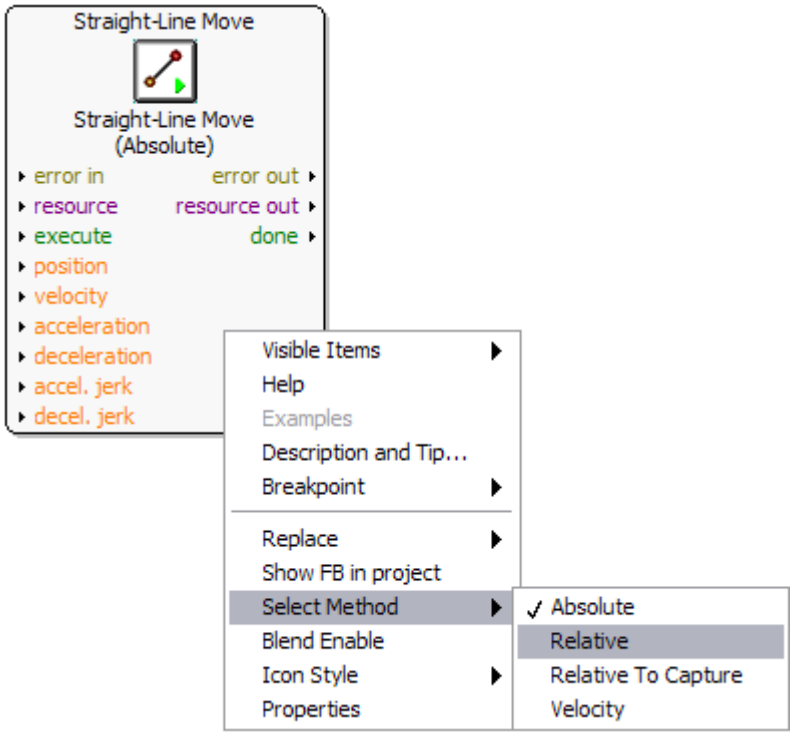
Un detalle importante acerca de la entrada de posición, es que la entrada cambia automáticamente para adaptarse al tipo de datos que le serán ingresados ya que pueden ser datos simples (control de un solo eje) o bien un arreglo de datos tan grande como el usuario desee (control de múltiples ejes), este tema será tocado nuevamente con un poco más de detalle posteriormente; y por último tenemos las salidas que sirven para visualizar datos, como el caso de la salida *done* y *error out*. La salida *done* sirve para conectar distintas funciones de movimiento en forma secuencial.

La función de línea recta, cuenta con diferentes métodos, entre ellos el absoluto y el relativo. El método absoluto iniciará el movimiento con respecto a la posición que el ensamblaje tiene como referencia de cero cuando fue creado, por ejemplo si en método absoluto desplaza el eje a la posición 100 y luego desea ir a la posición -100, el eje primero regresará a la posición cero y entonces iniciará el conteo de las 100 unidades en el sentido opuesto, ó sea en la dirección -100.

Para el método relativo la cosa es un poco diferente, en este método la referencia no por fuerza deberá ser la posición cero definida en el diseño del ensamblaje, de hecho la referencia del cero se moverá hasta donde terminó el último movimiento realizado por el ensamblaje. Si, por ejemplo, el mecanismo desplaza el eje a la posición 100 milímetros y luego ejecuta un movimiento a la posición -100. Lo que sucederá ahora es que cuando terminé el primer movimiento (100), en esa posición se establecerá la nueva referencia de cero de manera que cuando ejecute el segundo movimiento (-100) recorrerá los mismos 100 milímetros en dirección opuesta y al final del movimiento habrá regresado a la posición inicial.

Para seleccionar el método en la función de línea recta, basta con hacer clic derecho sobre la función, seleccionar el campo *Select Method* y finalmente habilitar el método deseado tal y como lo muestra la figura 78. Es importante que el usuario sepa que tanto las entradas como las salidas de la función no cambiaran para cualquiera de los métodos que sea seleccionado.

Figura 78. **Métodos absoluto y relativo de la función de línea recta**



Existe una variante para la función de línea recta y esta variante es la línea recta en espacio coordinado. Un espacio coordinado, esencialmente, es la agrupación de múltiples ejes y lo interesante de esto es que los ejes agrupados pueden ser movidos simultáneamente.

Para poder hacer esto el primer requisito es crear un *NI softmotion coordinate space*, que es un recurso que utilizan las funciones de movimiento para controlar ejes al mismo tiempo y lo que hace es agrupar distintos *softmotion axes* y convertirlos de recursos individuales de movimiento en un nuevo recurso multieje de movimiento.

El *coordinate space* se utiliza para especificar sobre que plano, el usuario desea ejecutar el movimiento en línea recta, por ejemplo se puede ejecutar una línea recta entre los planos XY, YZ, ZX e incluso realizar un trazo en tres dimensiones agrupando los ejes XYZ. Es importante señalar que para observar de manera adecuada el trazo de la línea recta ya sea en dos o tres dimensiones, eventualmente el usuario deberá mover la vista del ensamblaje dentro de *Solidworks* para contar con la perspectiva adecuada del trazo.

Ahora, se explicará el procedimiento para crear un *coordinate space,* lo primero es crear en el proyecto todos los *softmotion axes* que sean necesarios, como recordatorio los *softmotion axes* están ligados a los motores presentes en el ensamblaje de *Solidworks*; una vez creados los *softmotion axes* y ligados a sus respectivos motores, hará clic derecho sobre *My computer* en el *Project explorer*, se ubica el campo *New* y luego en NI *softmotion coordinate space*.

Una vez que se ha hecho clic sobre *coordinate space* aparecerá una ventana llamada *configure coordinate space.* Del lado izquierdo de la ventana se tiene una lista con los *softmotion axes* disponibles para crear el *coordinate space*, se seleccionan y luego se hace clic en la flecha para trasladarlos al espacio en blanco del lado derecho, una vez seleccionados los *axes* necesarios se da clic en *OK* y en el *Project explorer* aparecerá un nuevo campo con el nombre *coordinate space* 1. Las figuras 79 y 80 ilustran lo recientemente comentado.

Figura 79. **Creación de un NI *softmotion coordinate space***

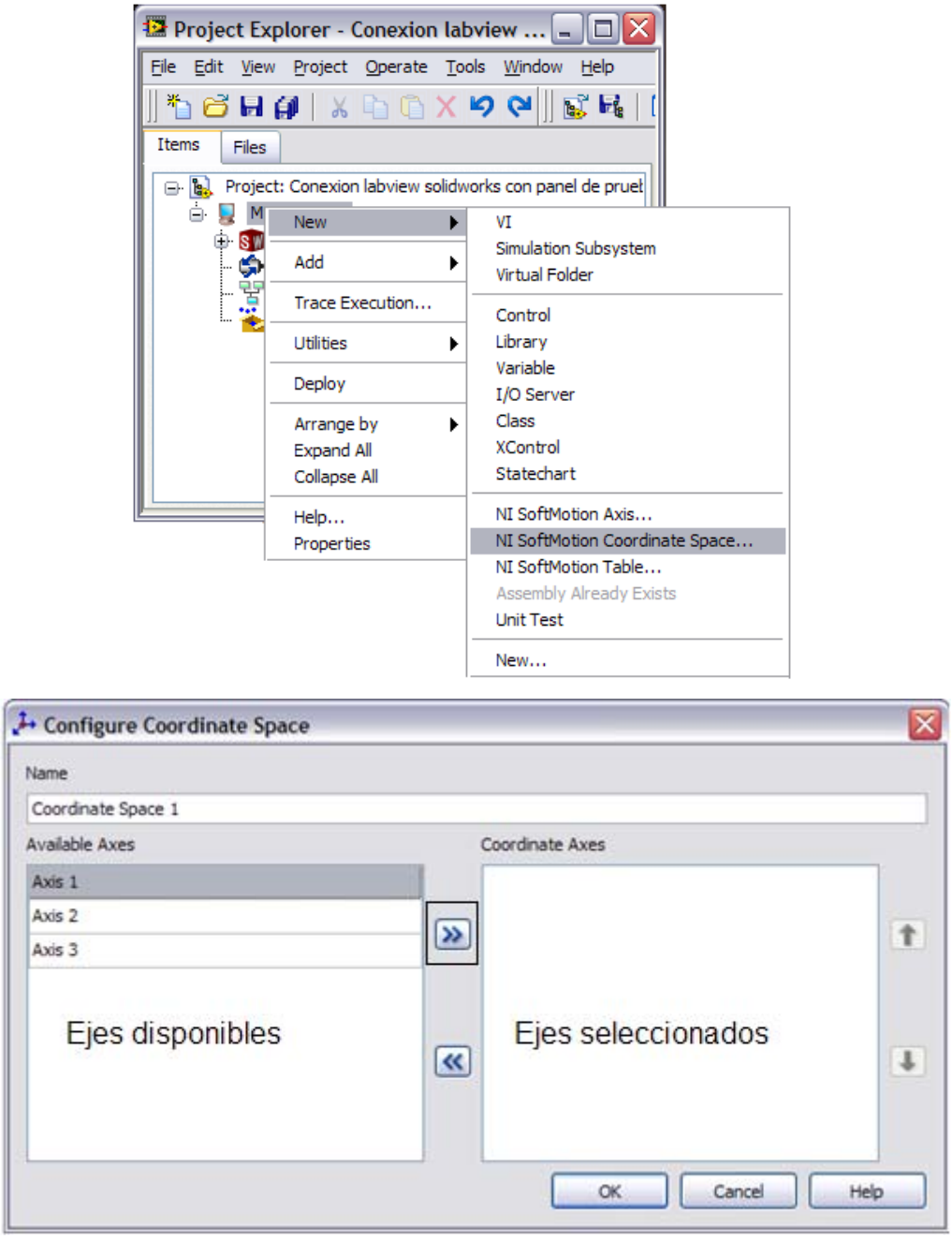
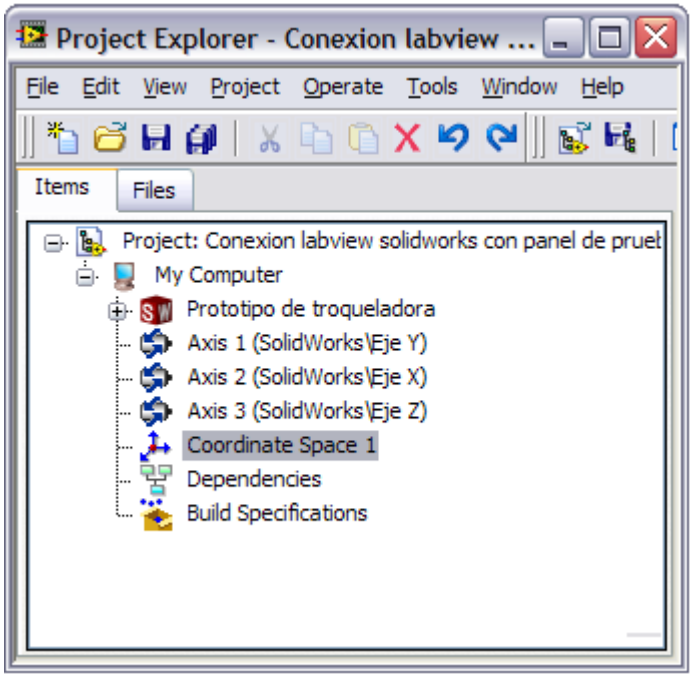


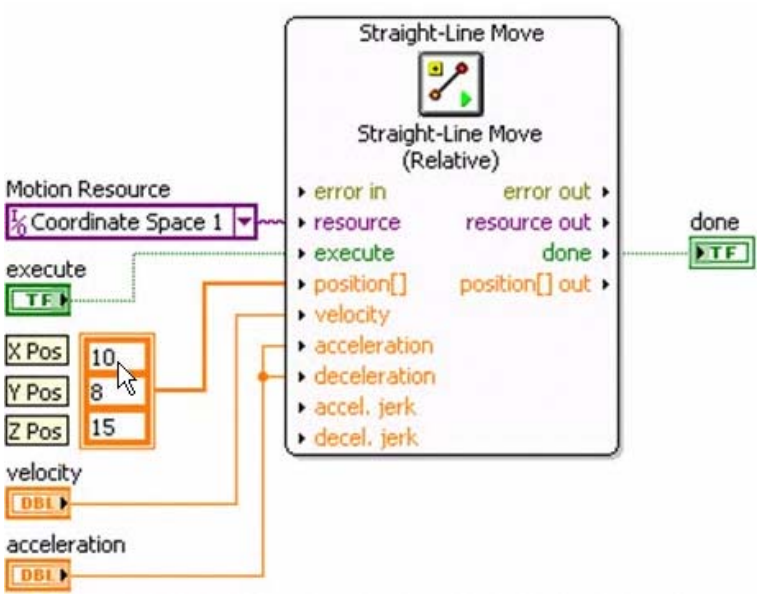
Figura 80. **Ventana del *project explorer* con el *coordinate space* agregado**



Ya creado el *coordinate space* se mostrará, de manera general, cómo programar un movimiento en línea recta controlando múltiples ejes a la vez, para iniciar el usuario podrá notar que la figura 81 muestra la misma función para línea recta que se presento al inicio de este apartado, lo que significa que la función es útil por igual utilizando un eje simple o bien un conjunto de ejes.

Es importante resaltar que desde el momento en que la entrada *resource* es ligada a un *coordinate space* la entrada de posición cambiará de tal forma que en lugar de esperar un solo dato, ahora espera recibir un arreglo de tres datos (ejes X, Y, Z) razón por la cual la entrada de posición cambia de "*position*" a "*position* [ ]" que se puede comprobar comparando la imagen de la figura 77 con la de la figura 81.

Figura 81. **Función de línea recta programada para realizar un trazo en un espacio tridimensional**



Además se puede observar que la línea o alambre, como se conoce en *LabVIEW,* que une el arreglo de datos a la entrada *position* [ ] de hecho es un poco más gruesa con respecto al alambre de un dato individual como el que une el control velocidad con su respectiva entrada en la función. Otro detalle que se muestra en la figura 83 es que la entrada *position* [ ] está ligada a un arreglo de constantes y no a un arreglo de controles, lo que implica que en el *front panel*, el usuario no podrá cambiar estos valores ya que no se verán reflejados y para modificarlos deberá parar la aplicación, ingresar en el *block diagram* y cambiarlos, cosa que no sería necesaria si utilizara controles.

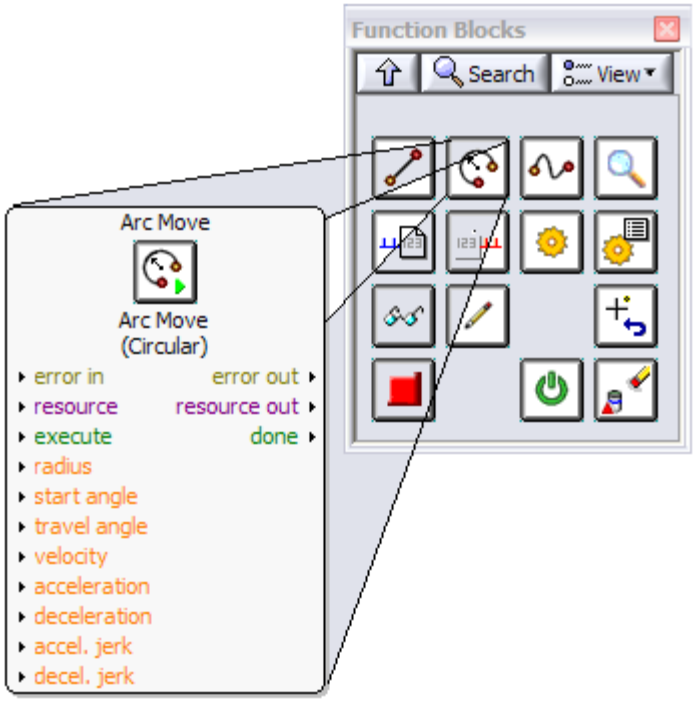
Entonces rápidamente, lo que hará este bloque será realizar el trazo de una línea recta en tres dimensiones moviendo el eje X 10 milímetros en la posición positiva, el eje Y 8 milímetros en la posición positiva y el eje Z 15 milímetros también en dirección positiva, todos al mismo tiempo, es decir un *coordinate space* se caracteriza por realizar un trazo de n dimensiones garantizando que todos los ejes involucrados en dicho trazo se detendrán prácticamente al mismo tiempo sin importar si un eje se mueve unos cuantos milímetro y otro tenga que ejecutar un movimiento de cientos de milímetros; con respecto a la velocidad y aceleración, ambos parámetros aplican de igual manera a los tres ejes ya que la misma función está controlándolos por medio del *coordinate space*.

4.3.2.2. Función para trazar movimiento circular

La siguiente función es para realizar movimientos circulares (*Arc move function*), esta función controla dos ejes de manera simultánea para formar una trayectoria circular. Los requisitos para que esta función pueda generar una trayectoria circular son, poseer dentro del proyecto un NI *softmotion coordinate space*, que implica poseer como mínimo dos NI *softmotion axes* ligados a un motor de *Solidworks.*

La figura 82 ilustra la función para trazar trayectorias circulares, dicha función se encuentra en la misma paleta de funciones donde se extrajo la función para trazar líneas. Como puede observar posee, al igual que todas las funciones de la paleta, entradas y salidas, pero las entradas en esta ocasión son un poco diferentes en comparación con las entradas de la función para el trazado de líneas.

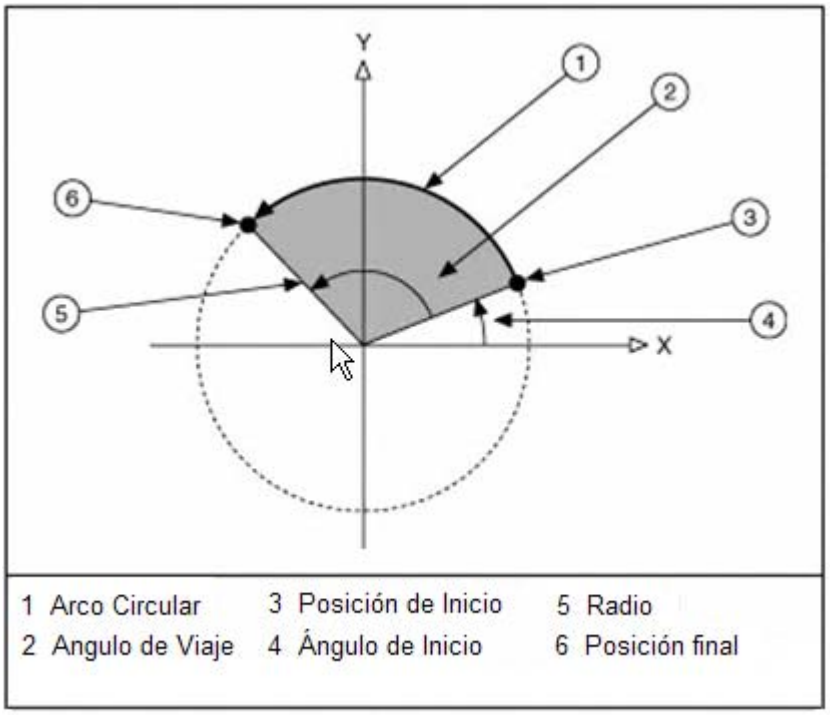
Figura 82. **Función para trayectorias circulares (*Arc move function*)**



En las entradas especializadas para esta función encontramos las de radio, ángulo de inicio y ángulo de viaje, que en la función aparecen etiquetadas como *radius, start angle y travel angle* respectivamente.

La entrada *radius*, ésta entrada servirá para especificar el radio de la trayectoria circular que será trazada y puede observarse bajo el número 5 en la figura 83, la siguiente es *start angle* que indicará a la función en qué punto, el arco o la trayectoria circular, empezará a ejecutarse y está acotada bajo el número 4. Por ejemplo si el usuario desea iniciar el trazo en el punto medio del plano XY en el primer cuadrante, el valor correcto sería 45 grados.

Figura 83. **Parámetros de un trazo circular**

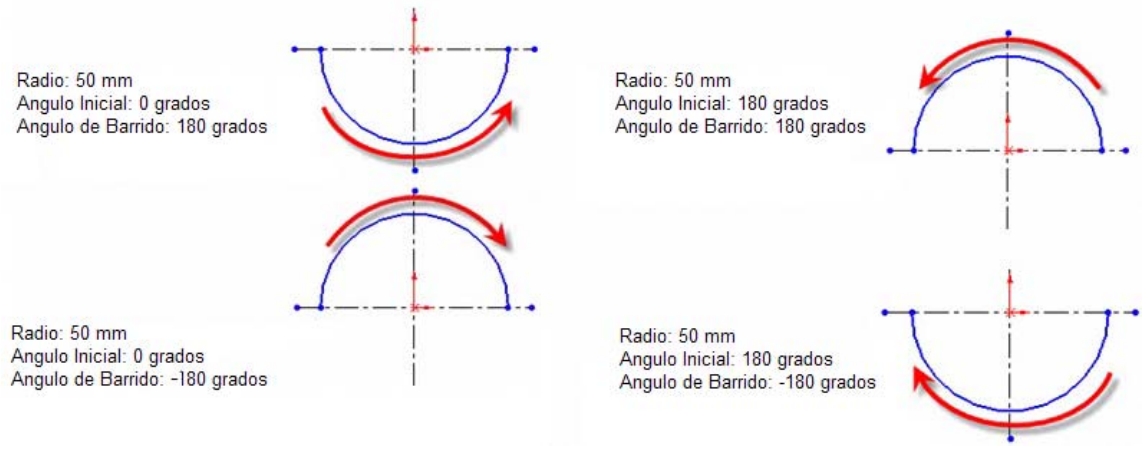


Travel angle, que indica cuantos grados barrerá la trayectoria circular o cuantos grados abarcará dicha trayectoria, por ejemplo si el usuario desea realizar medio círculo el valor adecuado será 180, correspondiente a los 180 grados que crean un semicírculo. La velocidad y aceleración funcionaran de la misma manera que para la función *straight line move* e indicaran la velocidad a la que será ejecutado el trazo y que tan rápido cambiará dicha velocidad respectivamente.

La siguiente figura, figura 84, mostrará distintas configuraciones para movimientos circulares. En la parte superior izquierda podrá observar una figura con 50 milímetros de radio, 0 grados como ángulo de inicio y 180 grados como ángulo de viaje o ángulo de barrido. Así este trazo iniciará del lado izquierdo sobre el eje X, abarcará 100 milímetros en su recorrido, finalizará del lado derecho sobre el eje X y todo esto lo hará en sentido contrario a las manecillas del reloj.

Ayudado, de nuevo, por la figura 84, en la parte inferior izquierda vemos otro trazo, en esta ocasión el ángulo de inicio es de nuevo 0 grados por lo que dicho trazo iniciará sobre el lado izquierdo del eje X, pero ahora el ángulo de barrido es de -180 grados. Este cambio de signo en el parámetro indica que el trazo al igual que el anterior finalizará, de nuevo, sobre el lado derecho del eje X, pero la variante es que ahora el trazo se hará en el sentido de las manecillas del reloj.

Figura 84. **Ejemplos de trazos circulares**



En la parte superior derecha de la figura anterior, podrá ver un trazo de radio 50, ángulo inicial de 180 y ángulo de barrido de 180. Estos parámetros indican que ahora el trazo iniciará en la parte derecha del eje X, terminará en la parte izquierda del mismo eje y se ejecutará en sentido contrario a las manecillas del reloj.

Finalmente en la parte inferior derecha de la figura 84, observará un trazo circular con los siguientes parámetros: radio de 50 milímetros, ángulo inicial 180 grados y ángulo de barrido de -180 grados. Como es evidente, este trazo iniciará en la parte derecha del eje X, abarcará 100 milímetros sobre este eje, finalmente se detendrá en la parte izquierda y todo esto moviéndose en el sentido de las manecillas del reloj.

Se debe aclarar que éstos, son solo algunos ejemplos que muestran cómo los parámetros de un trazo circular afectan dicha ejecución, obviamente todos estos parámetros pueden variar y ser ajustados de la manera que resulte más útil al usuario y de esta forma poder personalizar y/o ajustar su aplicación.

4.3.2.3. Función para trazar contornos

Ahora se explicarán los detalles de la función para contornos (c*ontour move function*). Un movimiento de contorno es aquel que está formado por una serie de posiciones, que al ser extrapoladas dan como resultado un trazo suave guiado por dichas posiciones. Estas posiciones se encuentran almacenadas como puntos dentro de una tabla y todos los puntos dentro de la tabla son relativos al punto de inicio del trazo.

Un contorno, en *LabVIEW*, puede ser creado para ambos, un eje simple (*softmotion axis*) o múltiples ejes (*softmotion coordinate space*), además aparece un parámetro extra llamado intervalo del contorno e indica el cambio de tiempo entre puntos, que puede ser interpretado como la velocidad del movimiento de contorno.

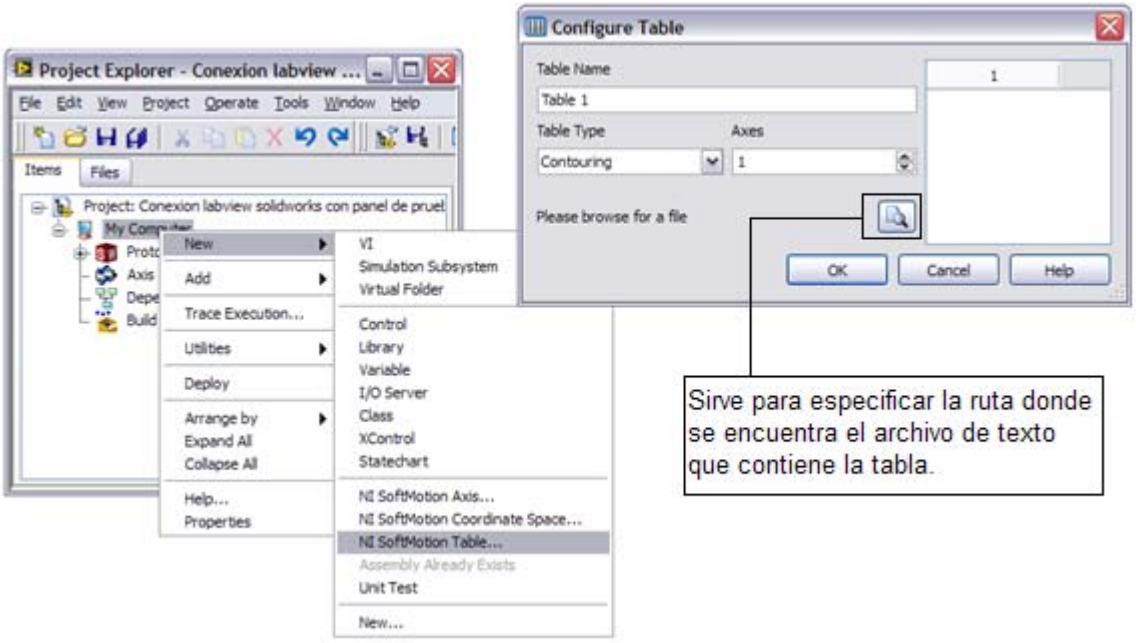
Los pasos preliminares o los requisitos para poder programar esta función son, primero haber creado los NI *softmotion axes*, así sea uno o varios ya que esta función trabaja de igual forma contornos de eje simple y de múltiples ejes.

Una vez agregados al proyecto los *softmotion axes* y el *coordinate space*, debemos agregar al proyecto de *LabVIEW* un campo nuevo conocido como *NI softmotion table,* del que se hablará a continuación y finalmente con la tabla creada y agregada al proyecto el usuario está listo para programar la función de contorno en el *block diagram* de su aplicación.

Para crear una NI *softmotion table*, de nuevo, el usuario deberá hacer clic derecho sobre el ícono *My computer* en el *Project explorer,* seleccionar *New* y luego seleccionar NI *softmotion table*, lo cual desplegará una ventana como la de la figura 85 llamada *configure table.* Dentro de esta ventana el usuario podrá configurar el nombre de la tabla en el espacio *table name,* en el espacio *table type* el usuario debe ser precavido y seleccionar *contouring* de las dos opciones disponibles, ya que lo que hará es un movimiento de contorno.

Si el usuario seleccionase la opción *camming*, que es la otra opción, la definición del tipo de tabla no sería la correcta, aunque de igual manera funcionaria. Pero la tabla para *camming* es utilizada por otra función que en español se conoce como leva electrónica.

Figura 85. **Creación de una NI *softmotion table***



Luego deberá seleccionar el número de ejes sobre los cuales dará información la tabla, puede ser uno, para un eje simple y si se está utilizando un *coordinate space* deberá ser de dos o más dependiendo del tipo de movimiento. Y finalmente el usuario debe especificar la ruta en donde se encuentra el archivo de texto con la tabla que contiene los datos por medio del botón resaltado en la figura anterior.

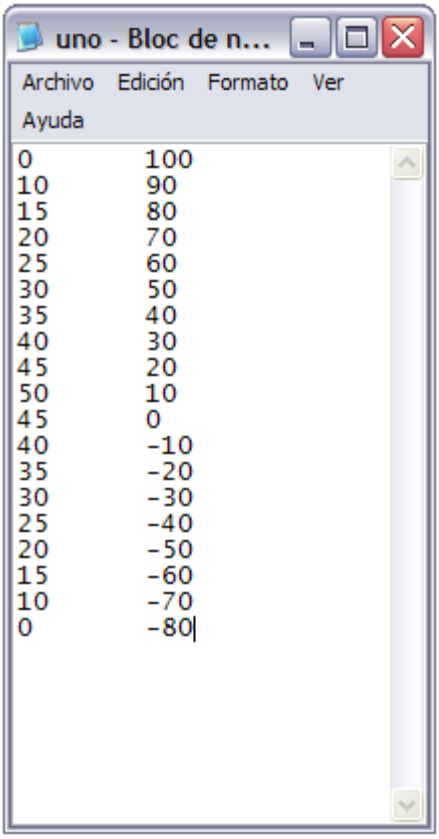
La tabla puede ser creada desde un procesador de textos básico como *wordpad*, luego la tabla debe ser creada teniendo en cuenta que si es para un eje la tabla tendrá una columna, si es para dos ejes la tabla tendrá dos columnas y si es para tres ejes la tabla deberá contar con tres columnas.

Los datos para cada eje no deberán tener espacios entre ellos, ya que de existir un espacio *LabVIEW* lo interpretará como un valor no existente y no aceptará la tabla, además, el usuario, debe de cuidar no dejar espacios en blanco luego del último dato de la tabla, ya que esto también es interpretado como valor no existente y causará error.

Los valores para los distintos ejes pueden estar separados por una tabulación, teniendo cuidado en no dejar espacios en blanco entre los datos y al final del último dato y finalmente siempre revisar que el número de ejes de la tabla coincida con el número de ejes seleccionado en la ventana para configurar la NI *softmotion table*. La figura 86 muestra la imagen de una tabla de dos ejes creada en *wordpad*.

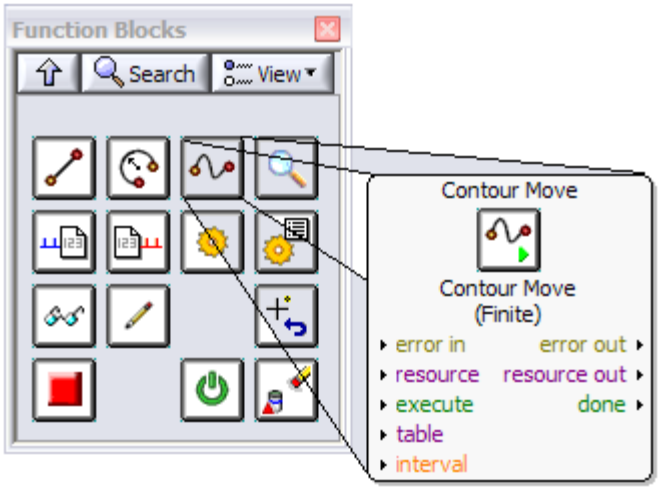
Un detalle importante acerca de la tabla es que si se observa el número 80 negativo al final de la segunda columna, se puede ver cómo el cursor está al lado del último número y no por debajo, si apareciera debajo del último número y guardara el archivo con ese espacio en blanco, seguramente, dará un error al especificar la ruta para el archivo de la tabla.

Figura 86. **Tabla de dos ejes para un *contour move***



A continuación se mostrará donde se puede encontrar la función para trazar contornos dentro de la paleta de funciones del NI *softmotion module* y se hablará un poco sobre los detalles más importantes referentes a dicha función. Para esto se usará, la imagen mostrada en la figura 87.

Figura 87. **Función para crear contornos (*Contour move function*)**



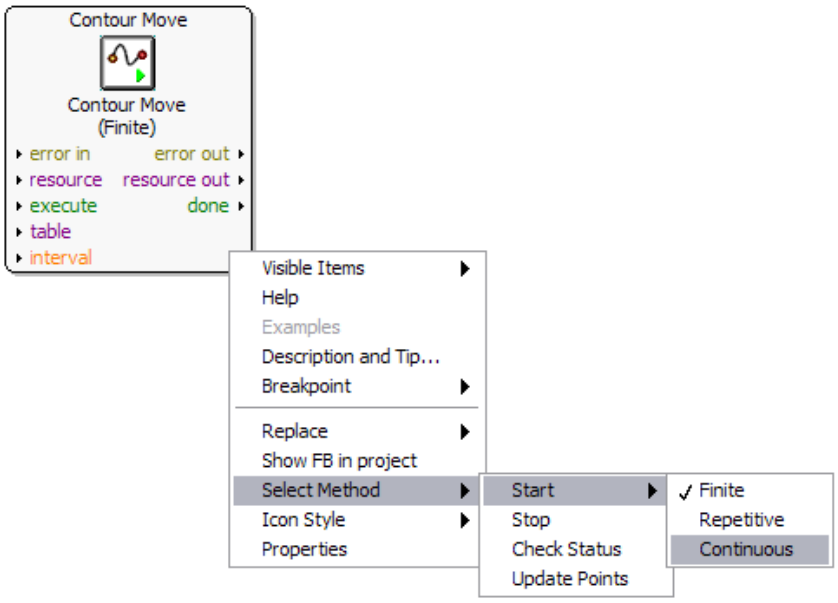
Como se puede observar en la figura, está la entrada *error in* para identificar errores, la entrada *resource* que debe estar ligada a un *softmotion axis* si es movimiento para un eje o bien a un *coordinate space* si en movimiento de dos o más ejes.

La siguiente es la de *execute* que indica cuando debe iniciar la ejecución del movimiento y que es *booleana*, luego se ve la entrada *table* que debe estar ligada a un control, el cual debe contener la ruta del archivo desde donde se importaran los datos y finalmente *interval* que determina el tiempo que le llevará a la función ir desde un punto al siguiente y que, prácticamente, es la velocidad a la que se ejecutará el trazo; entre mayor es el número más tiempo tardará y viceversa, el valor pre configurado si no se conecta ningún valor es 0.01.

Lo referente a las salidas es mínimo y como se ha mencionado antes las salidas son *error out* que muestra si algún error ocurrió durante la ejecución del programa e indica el número del error, *done* que es el indicador que se activa o cambia de flanco al momento de que la función ha terminado de realizar el trazo y *resource out* cuyos resultados no son relevantes por lo que no serán tomados en cuenta.

Otra característica importante sobre esta función son sus métodos, los cuales pueden ser cambiados haciendo clic derecho sobre la función, seleccionando el campo *select method*, luego *start* y finalmente aparecerán los tres métodos finito, repetitivo y continuo. El método finito realiza la función una única vez y se detiene, en el repetitivo es necesario especificar el número de repeticiones y el continuo que ejecutará el trazo una y otra vez mientras la aplicación permanezca en estado activo, la figura 88 ilustra.

Figura 88. **Métodos de operación de la función para contornos**



4.3.3. Comunicación entre *Solidworks* y aplicaciones programadas en *LabVIEW*

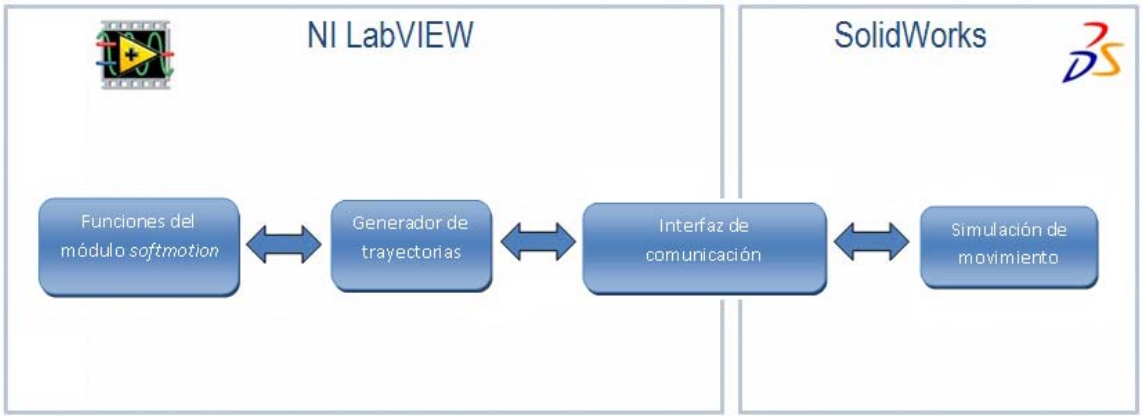
Como recordará, ya se explicó cómo conectar *Solidworks* y *LabVIEW* sin realizar programación alguna, por medio del panel interactivo de pruebas, pero luego de haber expuesto los detalles sobre las herramientas para programar trayectorias de movimiento, el usuario ya conoce cómo programar una aplicación en *LabVIEW*, pero aún desconoce el procedimiento para comunicar dicha aplicación con *Solidworks*.

Para empezar se explicará sobre la arquitectura que permite la comunicación entre *Solidworks* y *LabVIEW,* es decir cómo es que se lleva a cabo este proceso ya que ayudará al usuario a entender el proceso del despliegue de datos de una forma fácil. Luego se abordará cómo debe realizarse la configuración en orden de lograr desplegar los datos y correr la simulación en *Solidworks*.

Por parte de *LabVIEW*, para comunicarse con *Solidworks*, existen varios componentes que juegan una parte vital dentro de todo este proceso. En el nivel más alto tenemos las funciones del módulo *softmotion* que son las que el usuario utilizará para programar la aplicación, cuando ésta es ejecutada, las funciones hablan con el generador de trayectorias, que se encarga de generar todos los puntos involucrados en cada movimiento, información que luego pasa a la interfaz de comunicación.

La interfaz de comunicación, aún forma parte de *LabVIEW*, pero es quien comparte la información generada con *Solidworks*. Luego *Solidworks* tiene la habilidad de enviar información de vuelta hacia *LabVIEW*, información sobre desplazamientos, velocidades, motores, etc. que *LabVIEW* puede utilizar para procesarla y presentar resultados. Un esquema general de todo este proceso se muestra en la figura 89.

Figura 89. **Diagrama de bloques del proceso de comunicación entre *Solidworks* y *LabVIEW***



La interfaz de comunicación se conoce como *scan engine*, dicha interfaz se encarga de permitir o interrumpir la transferencia de datos entre *Solidworks* y *LabVIEW* al momento de correr la aplicación y mientras la aplicación esté en ejecución toda la información es intercambiada bilateralmente en lapsos iguales de tiempo y de forma simultánea, dicho lapso es llamado *scan period*.

El *scan period* es la velocidad a la cual está trabajando el *scan engine*. Continuando con el scan engine, éste cuenta con dos modos de operación el modo activo (*active mode*) y el modo de configuración (*configuration mode*).

El modo de configuración, es el modo en el que debe permanecer el *scan engine* cuando no está siendo ejecutada la aplicación, o bien cambiando el *scan engine* a este modo es la forma en que se detiene la simulación si se está ejecutando, además, con el *scan engine* en este modo, es cuando el usuario puede realizar modificaciones a la aplicación.

Por otro lado está el modo activo, que inicia la simulación; de modo que una vez que el usuario ha terminado con la configuración, ha desplegado todos los datos de manera adecuada y ha terminado con la programación de su aplicación en *LabVIEW*, podrá iniciar con la simulación cambiando el modo del *scan engine* de *configuration mode* a *active mode*.

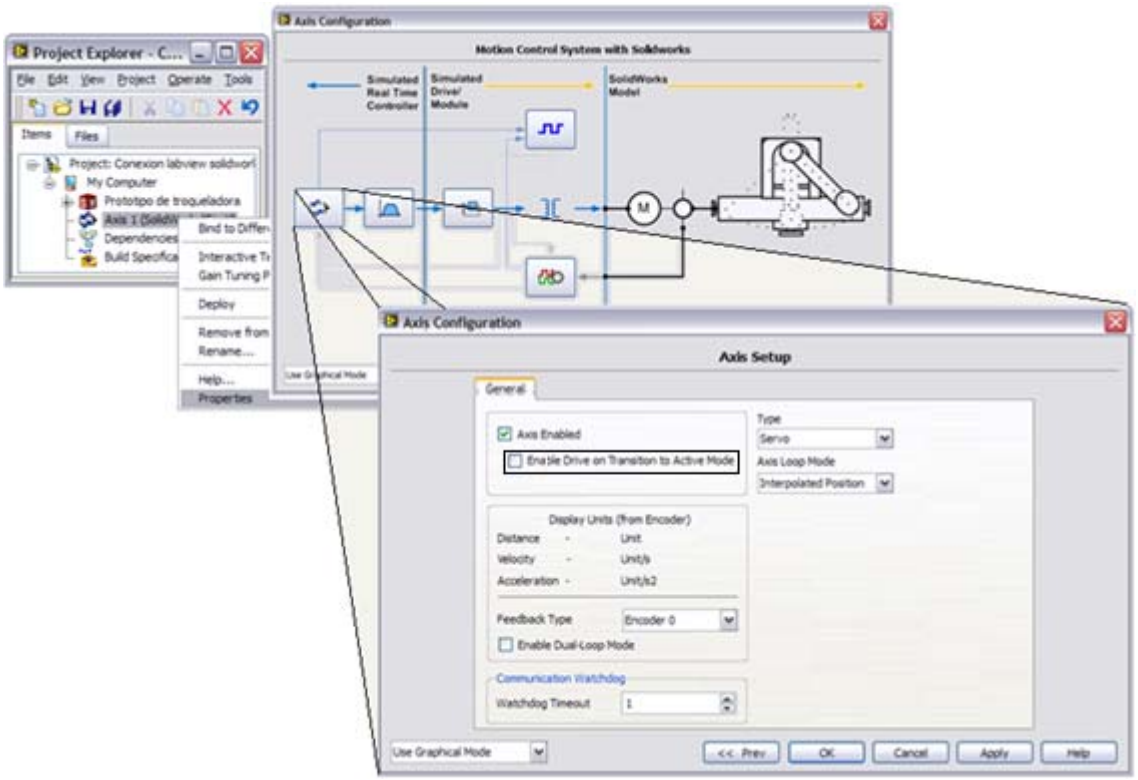
Ahora que se tiene una idea más clara de cómo se lleva a cabo la comunicación entre *Solidworks* y *LabVIEW*, el proceso para correr una simulación es el siguiente, primero se debe configurar el proyecto, luego deben ser desplegados todos los componentes del módulo NI *softmotion*, seguido se debe cambiar el estado del *scan engine* al modo activo, que iniciará la simulación y finalmente correr el VI, la aplicación que ha sido programada para controlar la simulación.

La configuración del proyecto se realiza de la siguiente forma, lo primero es configurar el *scan engine*, que es quien controla el intercambio de datos entre *Solidworks* y *LabVIEW*. Para configurarlo el usuario debe dirigirse al *Project explorer*, dar clic derecho sobre *My computer*, seleccionar el campo propiedades, luego seleccionar el campo *scan engine* y finalmente activar la casilla *start engine on deploy*, este procedimiento fue el mismo que se explicó para conectar el panel interactivo a la simulación. La realización de estos pasos asegura que el *scan engine* arranque para cuando el usuario desee ejecutar la aplicación.

El siguiente paso en la configuración del proyecto de *LabVIEW* es configurar los *softmotion axes.* La configuración que se explica a continuación debe realizarse a cada *axis* que haya sido agregado al proyecto, es decir si el proyecto sólo tiene un *softmotion axis*, sólo se hará una vez; pero si en el proyecto hay cinco *softmotion axes*, la configuración deberá realizarse cinco veces.

Para llevar a cabo la configuración se iniciará con clic derecho sobre el *softmotion axis* que se desea configurar, se selecciona el campo *properties* que desplegará una ventana llamada *axis configuration*, dentro de esa ventana el usuario debe hacer clic sobre el primer ícono a la izquierda que desplegará otra ventana llamada *axis setup,* y lo único que debe hacer, dentro de esta ventana, el usuario es habilitar el campo *Enable drive on transition to active mode* y clic en *OK*. Vuelvo a recordar que este procedimiento debe hacerse con cada *softmotion axis* presente en el proyecto, la figura 90 ilustra.

Figura 90. **Configuración de los *softmotion axes***



Una vez configurados todos los *softmotion axes*, el usuario puede moverse al siguiente paso, que es el despliegue de todos los componentes que tengan relación, directa o indirecta, con el módulo NI *softmotion*.

Los que poseen una relación directa son los NI *softmotion axes*, los *coordinate space* y las NI *softmotion table*. Los que tienen relación indirecta son el ensamblaje de *Solidworks* y *My computer*. Puede que cruce por la mente del usuario que los *VIs* también tienen una relación indirecta con dicho módulo, lo cual es correcto pero no se deben desplegar, ya que no son dependencias necesarias.

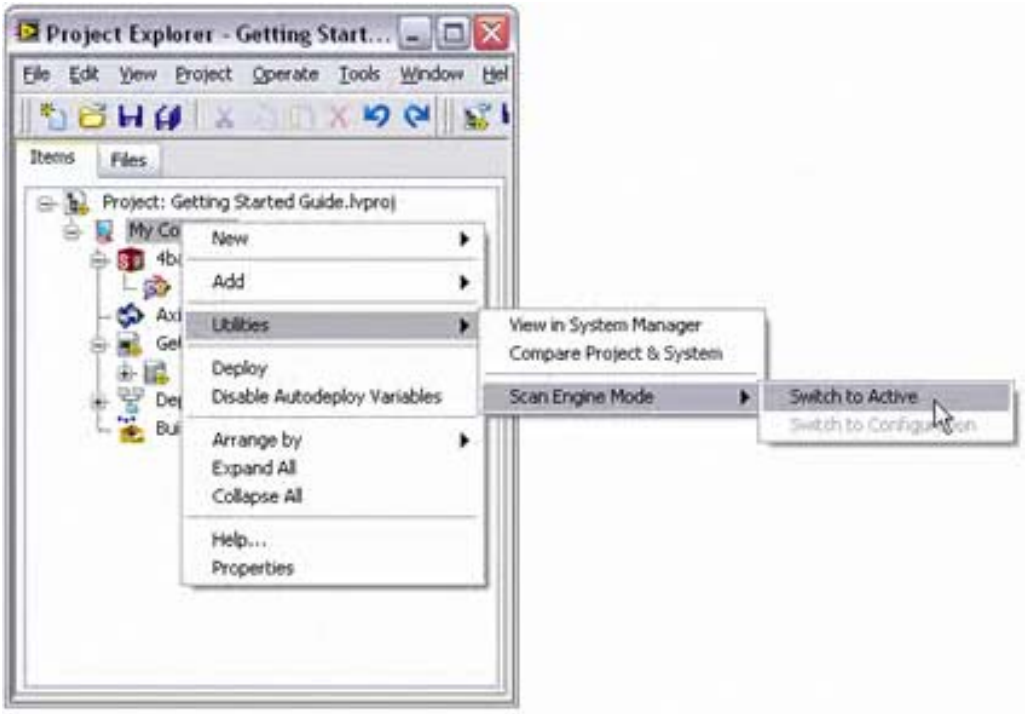
Este proceso asegura que todas las dependencias necesarias, se encuentren disponibles a la hora de desplegar los datos. Para realizar el despliegue de datos el usuario deberá seleccionar todos los componentes mencionados anteriormente, exceptuando los *VIs*, realizar clic derecho sobre cualquiera de los componentes seleccionados y hacer clic sobre el campo *deploy*, como se observa en la figura 91.

Figura 91. **Proceso de despliegue de datos desde el *Project explorer***

Una vez los datos han sido desplegados, si es la primera vez que el usuario despliega los datos del proyecto, el *scan engine* cambiará al modo activo de forma automática, y si no es la primera vez, entonces el usuario deberá de hacerlo de modo manual.

Recuerde que este paso es necesario para iniciar la simulación, para hacerlo de modo manual deberá hacer clic derecho sobre *My computer*, luego se dirige a *Utilities*, *Scan engine mode* y finalemente *Switch to active*, tal y como se ve en la imagen de la figura 92.

Figura 92. **Cambio del modo de operación del *scan engine***



Una vez que el usuario ha activado el *scan engine* y la simulación se está ejecutando, es momento de correr el *VI* donde se ha programado algún tipo de movimiento para el mecanismo y tener la oportunidad de comprobar cómo funciona la aplicación y es así, cómo a grades rasgos, se logra una simulación.

Luego que el usuario ha terminado de ejecutar la aplicación, debe cambiar el modo del *scan engine* de vuelta al modo de configuración, que como se mencionó sirve para detener de manera correcta la simulación y cuyo procedimiento es el mismo que para activarlo, clic derecho sobre *My computer*, *Utilities*, se dirige a *scan engine mode* y por ultimo *switch to configuration*.

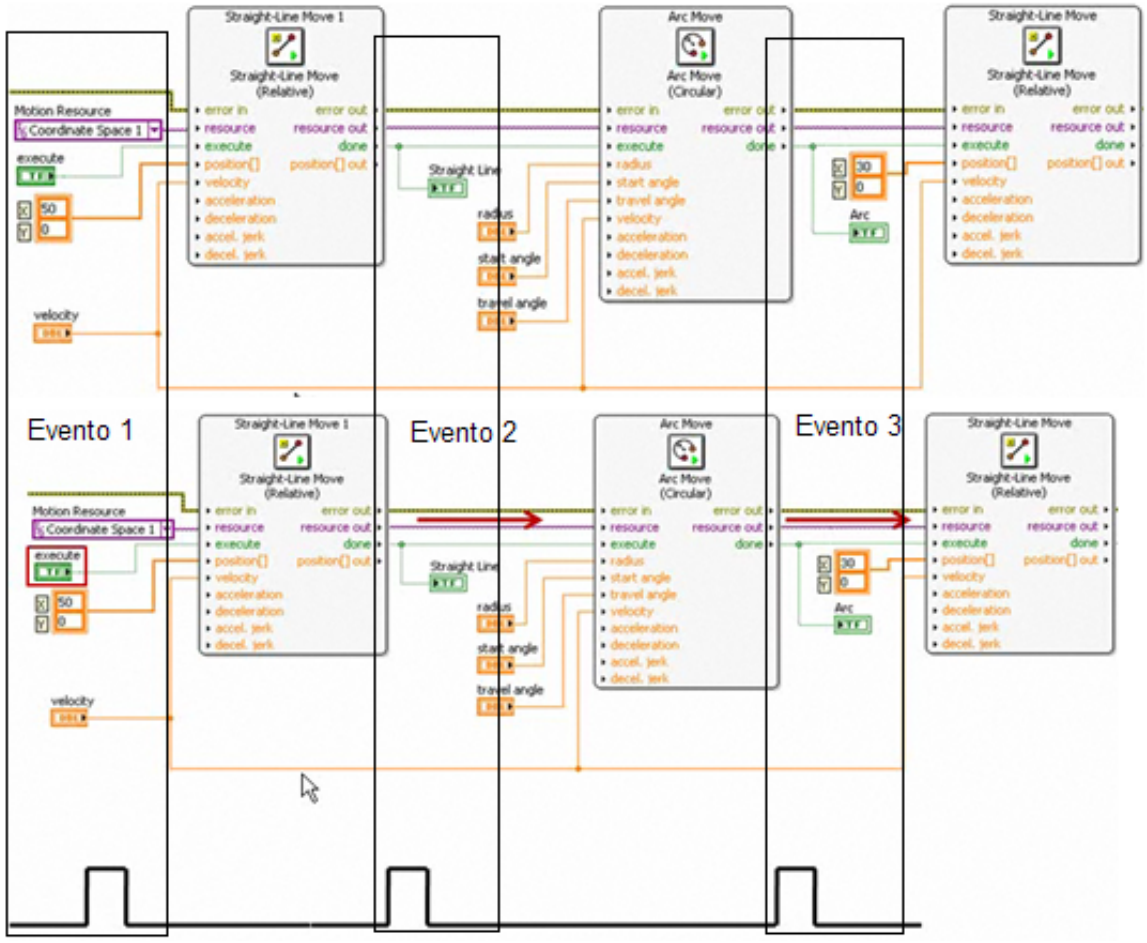
4.3.4. Características de programación para aplicaciones de *LabVIEW*

Dentro de esta sección se explicarán algunas características avanzadas de programación, como la combinación de varias funciones de movimiento para crear trayectorias controladas secuencialmente, también cómo agregar sensores de retroalimentación para realizar programación por lógica de eventos y finalmente cómo utilizar la función *trace path* para validar el diseño creado.

Las funciones de movimiento pueden ser conectadas de dos formas, en serie o en paralelo. Dichas conexiones son logradas aprovechando las características de las terminales *execute* y *done*, para controlar cuando se deben ejecutar y en qué orden deben realizarse.

Suponga que cierto mecanismo por razones de evitar un obstáculo debe realizar la siguiente trayectoria, desplazarse en línea recta, luego ejecutar una trayectoria en arco, para evitar el obstáculo, y finalmente volver a moverse en línea recta. Los movimientos deben ser ejecutados en serie, lo cual significa uno seguido del otro. La figura 93 ayudará a entender mejor cómo se programaría este tipo de trayectoria en *LabVIEW*.

Figura 93. **Programación de funciones ejecutadas en serie**



Lo primero es tener en cuenta que son tres funciones una para cada movimiento, la primera es la función de línea recta, la segunda es la función de arco y la tercera de nuevo la función de línea recta. Luego es importante resaltar que el recurso de movimiento (*resource*) es el mismo para las tres funciones y es un *coordinate space*, que agrupa los ejes X y Y juntos.

Para la primera función, línea recta uno, note que en la entrada de *position*, la cual es un arreglo de dos casillas, únicamente el valor correspondiente al eje X tiene un valor distinto de cero, lo cual indica que el mecanismo se desplazará, para esta función, únicamente sobre el eje X.

Al momento de activar el control *execute* sucederá lo siguiente, primero la acción anterior entregará un flanco positivo a la función línea recta uno (evento 1 en la figura) con lo que iniciará a moverse el mecanismo. Una vez que la función de línea recta haya finalizado el indicador *done* cambiará de falso a verdadero, lo que brinda un segundo flanco positivo para la entrada *execute* de la función de arco (evento 2 en la figura).

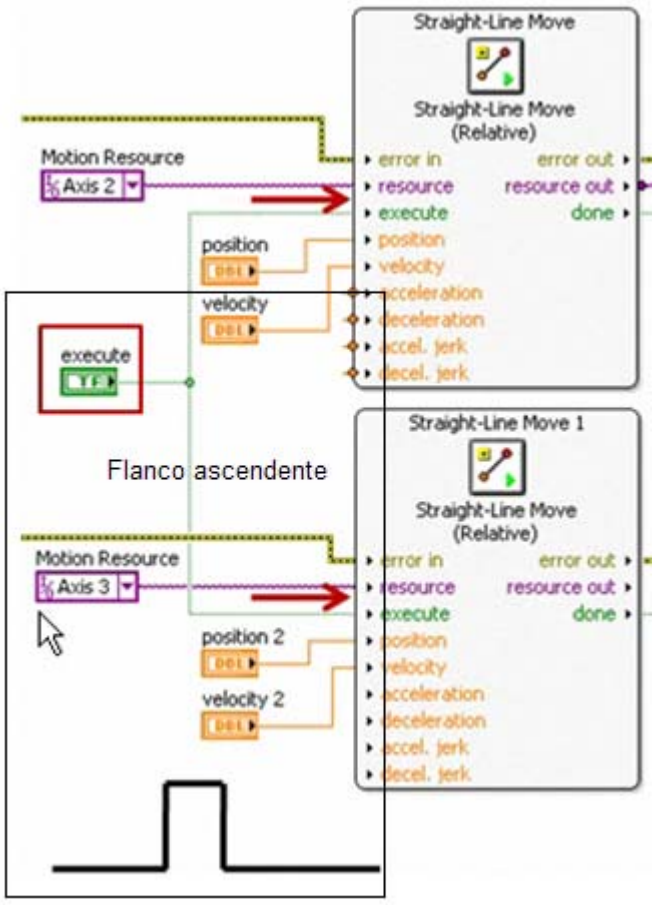
Con esto se iniciará a ejecutar el movimiento en arco, el cual una vez finalizado entregará un tercer flanco positivo en la salida *done* que la entrada de la función línea recta dos entenderá cómo la señal necesaria para iniciar el segundo movimiento del mecanismo en línea recta (evento 3 en la figura) y finalmente el indicador *done* de esta función indicará que el movimiento secuenciado ha concluido presentando un último flanco positivo el cual puede ser aprovechado por un indicador *led*.

Como se ve, es bastante simple lograr este tipo de programación secuenciada combinando las señales de las terminales *done* de una función previa con *execute* de una función posterior. Es importante resaltar que para las tres funciones se está utilizando la misma velocidad, ya que el valor proviene del mismo control y que no necesariamente debe ser así, de hecho cada función pudo haber sido programada con diferentes velocidades y aún así lograr secuenciarlas sin problema.

Para combinar funciones de movimiento en paralelo, suponga el siguiente escenario, en el cual en mecanismo debe mover un brazo hacia abajo sobre el eje Y al mismo tiempo que una faja transporta hacia el brazo (eje X) la pieza que debe ser tomada. Para este tipo de movimiento se requieren dos funciones de línea recta.

De nuevo este tipo de movimiento es bastante simple y similar al movimiento en serie. Una de las diferencia es que en para este movimiento un mismo control *execute* inicializará ambas funciones de línea recta simultaneamente, además para este tipo de movimiento del mecanismo se utilizarán dos ejes separados, el eje Y que en la figura 94 aparece como *axis* 2 y el eje X que aparece bajo el nombre de *axis* 3, y no un *coordinate space* como en el anterior. Una vez finalizados ambos movimientos, las terminales *done* entregarán flancos positivos que pueden ser aprovechados por el usuario para inicializar otra función o bien para indicar la finalización del movimiento.

Figura 94. **Programación de funciones ejecutadas en paralelo**



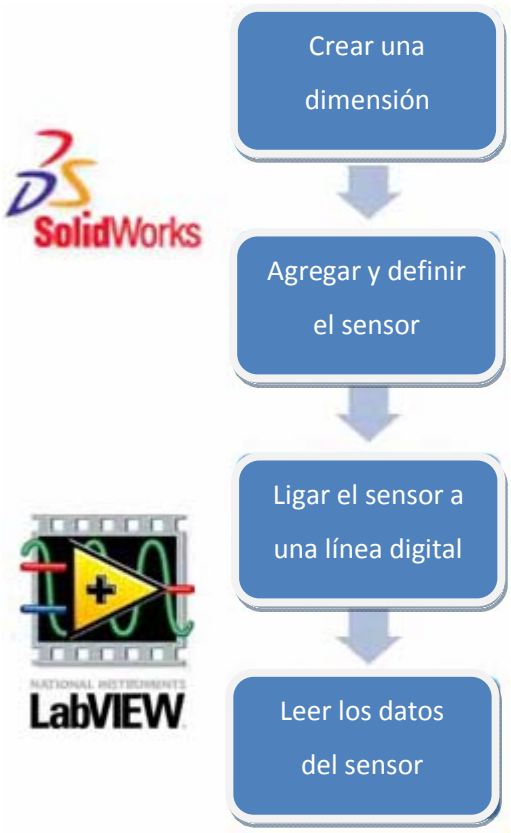
Como se ilustra en la figura 94, al momento de presionar el control *execute* tiene lugar un flanco ascendente (flanco positivo) que inicializa ambas funciones de manera simultánea y que funciona para el número de ejes que el usuario desee controlar, es importante resaltar que en este método se tiene control sobre todos los parámetros de la función tales como velocidad, aceleración, desaceleración y posición.

Otro método para controlar funciones de movimiento en paralelo es utilizando un *coordinate space* que, como ya se ha explicado anteriormente, lo que hace es agrupar el número de ejes que el usuario desee y presentarlos como un solo recurso de movimiento. En este método sólo se tiene control de la posición para cada eje por medio de un arreglo de constantes o bien un arreglo de controles, características como velocidad, aceleración y desaceleración serán todas iguales ya que para este método sólo se utiliza una función de movimiento para controlar todos los ejes.

Al trabajar en conjunto con *Solidworks* y *LabVIEW* el usuario cuenta con la posibilidad de crear sensores virtuales que funcionan como sensores reales dentro de un mecanismo. Los sensores virtuales que el usuario creará están basados en las dimensiones del ensamblaje y emitirán una señal verdadero o falso (lógica *booleana*) de acuerda a la condición de alerta que el usuario especifique.

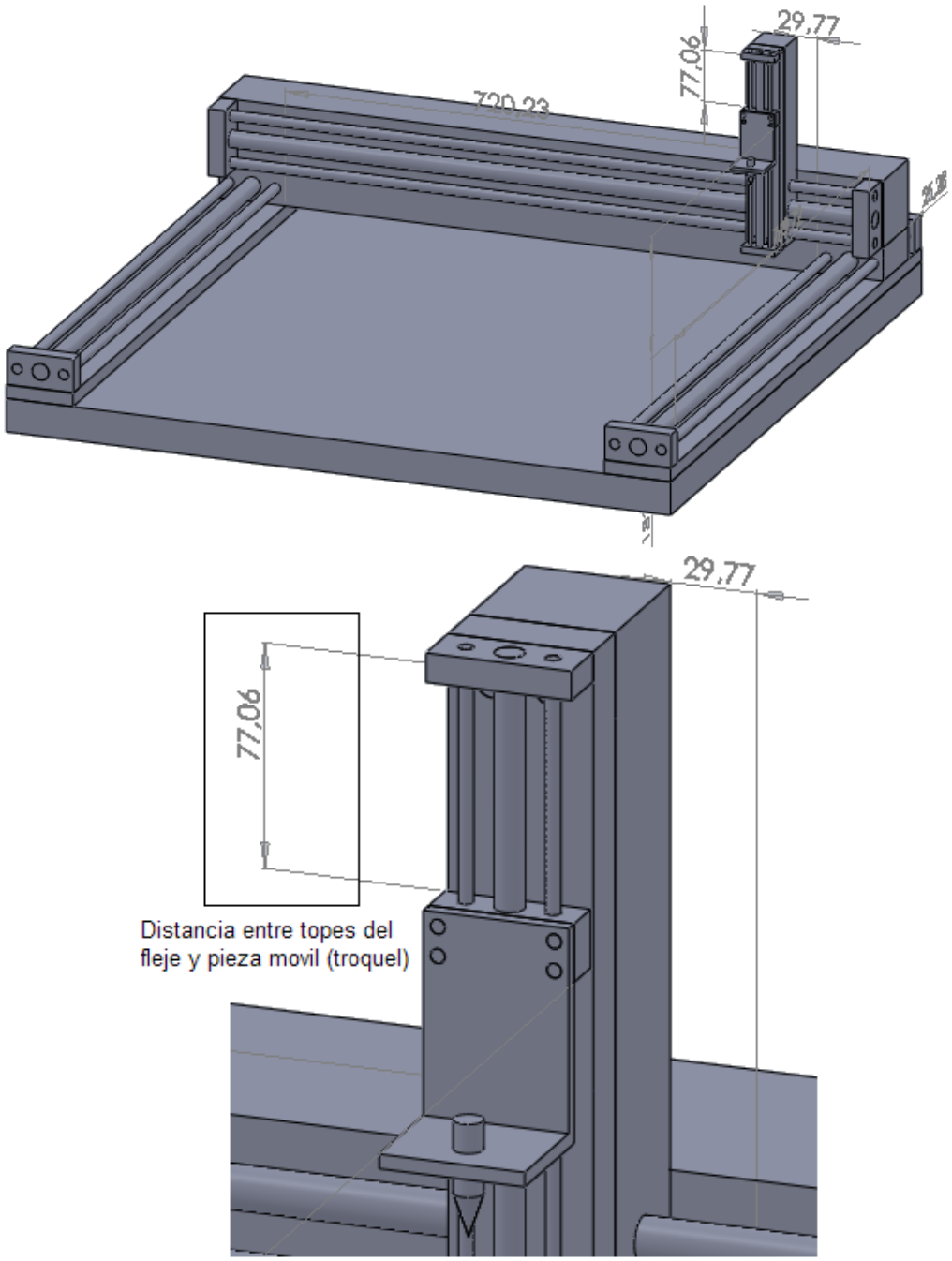
El primer paso para la creación de un sensor virtual es crear la dimensión a la cual estará relacionado dicho sensor, luego se crea el sensor y se definirán sus características. Estos dos pasos son realizados dentro de *Solidworks,* los siguientes pasos tienen lugar en *LabVIEW* y son mapear o ligar el sensor a una línea digital, y finalmente colocar una función para leer los valores del sensor. La figura 95 muestra un esquema de este proceso.

Figura 95. **Esquema para la creación de un sensor virtual**



Entonces el primer paso es agregar una dimensión, esta dimensión será la que represente el sensor y debe ser colocada entre dos componentes, por ejemplo en el prototipo de troqueladora que se presentó unos párrafos atrás, se puede colocar una dimensión en el fleje del eje Z, que controla los movimientos del troquel, para evitar que colisionen los topes del fleje con la parte móvil. La figura 96 ilustrará esta parte del mecanismo y su respectiva dimensión agregada.

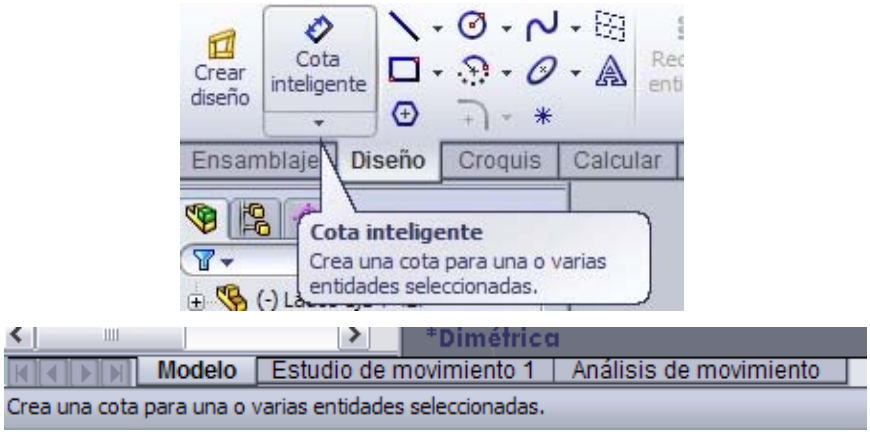
Figura 96. **Dimensión agregada entre dos piezas de un ensamblaje**



Para agregar la dimensión dentro del ensamblaje basta con hacer clic sobre el botón llamado cota inteligente, dentro de la barra de funciones de ensamblaje en *Solidworks*, es importante que el usuario se encuentre en la pestaña de modelo en la parte inferior en la ventana. Una vez seleccionada dicha función el usuario debe especificar las caras de ambas piezas entre las cuales desea colocar esta distancia inteligente y listo.

Se le llama cota inteligente y se utiliza precisamente esta distancia para que funcione como sensor, ya que al momento de mover el mecanismo la distancia cambiará de valor de acuerdo al movimiento de las piezas involucradas. Por ejemplo si ambas piezas se alejan la distancia crecerá y si se acercan la distancia disminuirá. La figura 97 muestra el botón y las pestañas seleccionadas para efectuar este procedimiento.

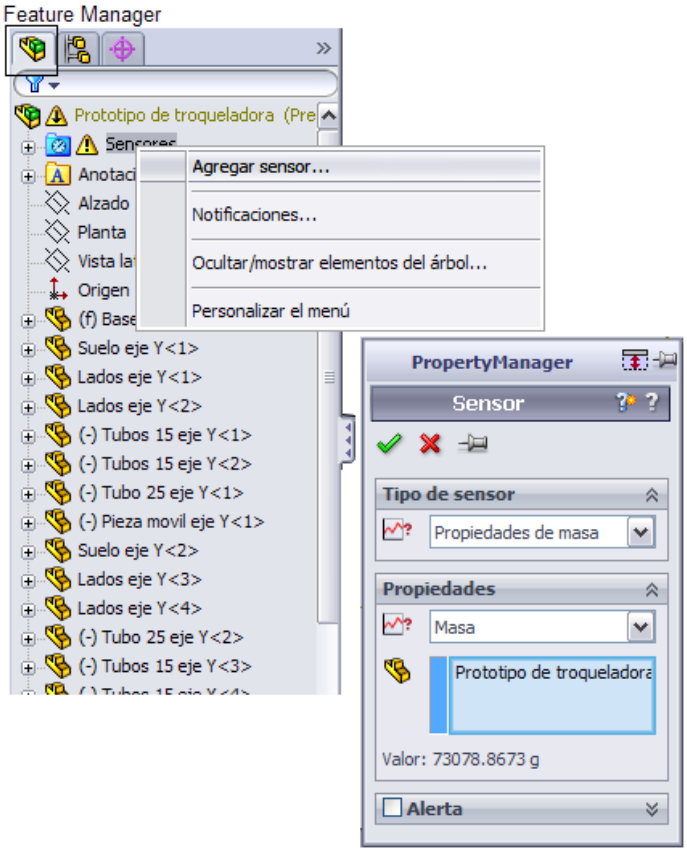
Figura 97. **Botón para agregar cota inteligente**



Ahora corresponde agregar el sensor en *Solidworks*, ya que hasta el momento sólo se ha creado la dimensión. Para lograrlo el usuario deberá hacer clic derecho en la carpeta de sensores en el árbol del *feature manager* en la parte izquierda de la ventana del ensamblaje en *Solidworks*.

El clic derecho desplegará un menú en el cual se selecciona el campo agregar sensor, lo que mostrará un nuevo menú que se ilustra en la figura 98 llamado *Property Manager.*

Figura 98. **Cómo agregar un sensor en *Solidworks***

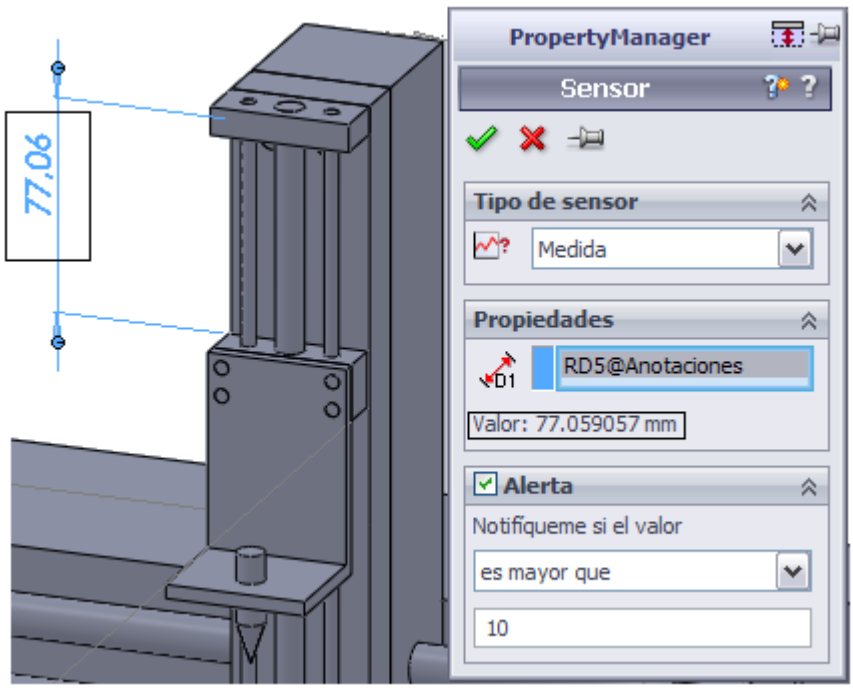


Una vez desplegada la ventana del *property manager* el usuario debe definir el sensor de la siguiente manera, en el espacio tipo de sensor debe seleccionar el campo medida, luego seleccionar la distancia a la cual será ligada el sensor (la que fue creada unos párrafos atrás) y finalmente escoger el tipo de alerta deseada por el usuario. Hay varios tipos de alertas para escoger y el usuario deberá utilizar la que más se adecue a sus necesidades.

Ahora se mostrará cómo realizar este proceso, primero se debe seleccionar el campo medida en el área de tipo de sensor, la ventana cambiará de acuerdo al tipo de sensor elegido, luego deberá especificar la distancia a la que desea ligar el sensor, para lograrlo únicamente debe situarse sobre su ensamblaje y dar clic sobre la distancia que medirá el sensor y de inmediato verá como en el área de propiedades se llena el campo vacio con unas letras, abajo se indica la distancia actual en milímetros bajo el nombre de valor y en el ensamblaje la distancia seleccionada cambia de color gris a naranja.

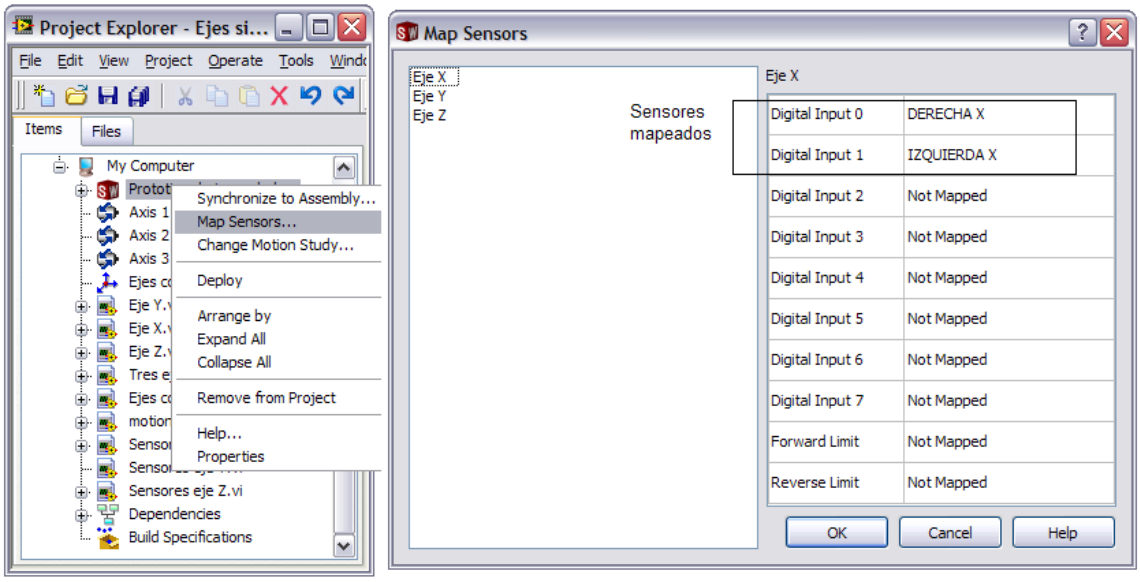
El campo de alerta deberá ser elegido para que el sensor indique cuando la distancia cumpla con la condición seleccionada por el usuario, por ejemplo si escogió es menor que, significa que el sensor indicará o dará una señal cuando la distancia entre las piezas sea menor que la distancia especificada por el usuario en el campo de abajo. Todos los valores numéricos expresados en las condiciones de alerta hacen referencia a milímetros. La figura 99 mostrará cómo se debería de ver la ventana de *property manager* una vez configurados todos los campos del sensor.

Figura 99. **Configuración de un sensor en *Solidworks***



El tercer paso es mapear o ligar a el sensor a una línea digital en *LabVIEW,* para lograrlo el usuario debe situarse sobre el ensamblaje de *Solidworks* en la ventana *Project explorer*, hacer clic derecho y seleccionar el campo *Map sensors*. Esta acción muestra una nueva ventana y como se observa en la figura 100, del lado izquierdo se tienen todos los ejes de *Solidworks* disponibles y del lado derecho se observan las líneas digitales para cada uno de estos ejes de *Solidworks* representados por *softmotion axes*. Para cada eje se cuenta con una lista de los sensores creados en *Solidworks*.

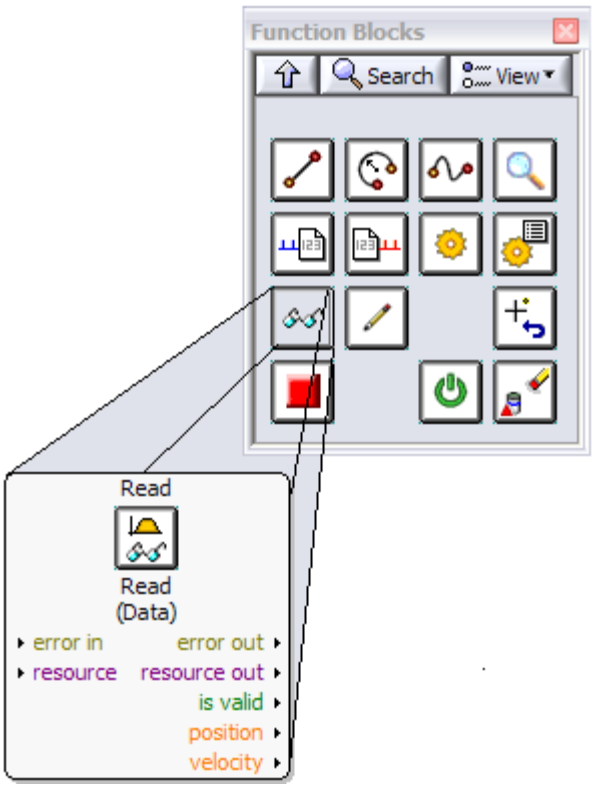
Figura 100. **Mapeado de sensores en *LabVIEW***



En esta ventana, se seleccionan el o los sensores para cada eje y de esta manera se mapean a una línea digital iniciando desde la línea digital cero, es decir el primer sensor seleccionado quedará ligado a la línea digital 0, el siguiente sensor a la línea digital 1 y así sucesivamente.

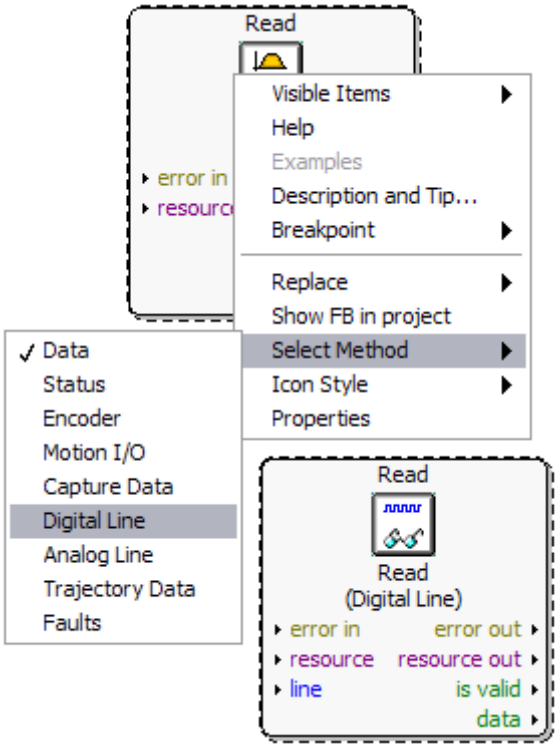
Finalmente se deberá leer el valor del sensor en *LabVIEW*, y para hacerlo se hará uso de la función *read* de la paleta de funciones del NI *softmotion module.* Esta función es encontrada en el mismo lugar que las demás funciones explicadas hasta ahora, *function palette*, *motion and vision*, NI *softmotion*, *function blocks* y finalmente *read function,* como lo ilustra la figura 101.

Figura 101. **Función *read* del NI *softmotion***



Una vez que se tiene la función *read* en el *block diagram*, se explicará cómo conectarla. Lo primero es seleccionar el método adecuado, ya que se están leyendo valores de líneas digitales, el método a seleccionar es *digital line*. Para lograrlo se hace clic derecho sobre la función, se selecciona el campo *select method* y finalmente se escoge el campo digital line y la función cambiará a la presentada en la figura 102.

Figura 102. **Bloque *read* con método *digital line***



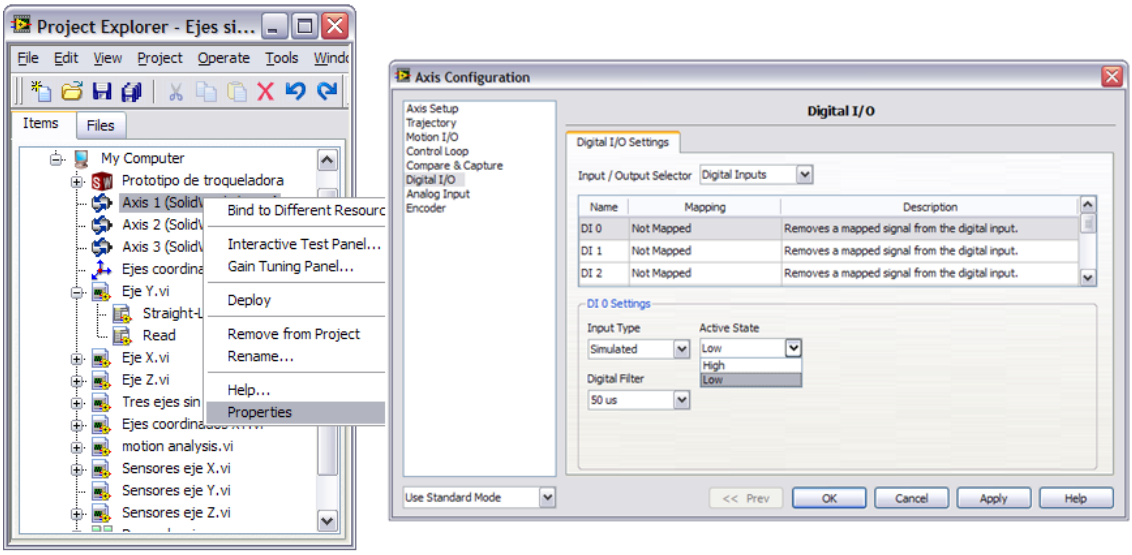
Hay dos entradas importantes *resource* y *line*. La entrada *resource* hace referencia al NI *softmotion axis* al cual ha sido mapeado el sensor, por ejemplo si se desea leer un sensor del eje X y dicho eje está ligado al *axis* 2 en el proyecto de *LabVIEW*, en la entrada *resoruce* deberá ligarse el *axis* 2. La otra entrada en discusión, la entrada line, es utilizada para especificar qué sensor será leído por esta función, por ejemplo si se desea leer el sensor de la derecha y está mapeado en la línea digital 1, entonces el usuario deberá ligar una constante numérica con valor 1 a dicha entrada.

El usuario debe tener en mente que las líneas digitales inician a contarse desde el cero, así el primer sensor mapeado, estará ligado a la línea digital cero, el segundo a la línea digital 1, etc. es importante indicar la línea adecuada ya que de lo contrario se estará leyendo el sensor equivocado.

En las salidas tenemos *is valid* y *data,* la primera entregará un valor *booleano* verdadero o un flanco positivo cuando la lectura se lleva a cabo de manera normal, la única manera de que entregue un valor falso o flanco negativo es que se pierda la lectura durante un ciclo del *scan engine* por errores de programación o similares. La salida *data* se encarga de avisar cuando la alarma se haya disparado, es vital mencionar que en *LabVIEW* los sensores están pre configurados como *active low,* que significa que al activarse la alarma enviará un flanco negativo o falso para indicar la ocurrencia de un evento.

Por esta razón usualmente en la programación de los sensores se coloca una compuerta *not* a la salida de la terminal data para convertir el flanco negativo en un flanco positivo cuando se activa el sensor. También es importante mencionar que se puede cambiar este comportamiento en los sensores haciendo clic derecho sobre el *softmotion axis* al cual está ligado el sensor, seleccionando *properties*, luego *digital I/O* que finalmente desplegará una ventana en la cual se puede modificar esta propiedad, en la esquina inferior derecha como lo muestra la figura 103.

Figura 103. **Cómo modificar el estado *active low* de los sensores en *LabVIEW***



Otra aplicación útil de los sensores es evitar colisiones entre piezas, para lo cual se crea una dimensión entre dos piezas y se coloca una alerta para indicar cuando están muy cercanas para detener el mecanismo antes de que éstas colisionen, el usuario debe tener en mente que los sensores también sirven para controlar la ejecución de eventos en secuencia ya sea serie, paralelo o la combinación de ambos y que esto se puede lograr por medio de la salida data y el flanco entregado por ésta al cumplirse la condición especificado en la configuración del sensor en *Solidworks*.

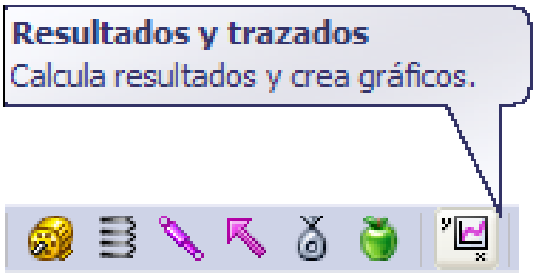
Otra herramienta de mucha utilidad es el *trace path plot*, lo que hace esta herramienta es dar seguimiento a la trayectoria recorrida por un punto especificado por el usuario dentro del ensamblaje, y dejar huella de los movimientos que ha realizado este punto durante una simulación.

Es de mucha utilidad cuando no es fácil visualizar el recorrido y la forma cómo ha realizado dicho recorrido una pieza específica, puede que haya realizado círculos mientras se desplaza y sin una huella, una marca del recorrido sería imposible observar que tipos de movimientos ha realizado el mecanismo.

Esta herramienta forma parte de la barra de herramientas de resultados y trazados en la ventana de *Solidworks*. Para configurarla se realizan los siguientes pasos, primero es hacer clic sobre el ícono de resultados y trazados en la parte inferior derecha de la ventana de *Solidworks*, luego en la ventana que desplegará dicha acción, el usuario deberá seleccionar la categoría desplazamiento/velocidad/aceleración y luego en el espacio de abajo, en la subcategoria, deberá seleccionar el campo ruta de trazo y finalmente especificar dentro del ensamblaje qué punto será el seleccionado para dejar huella de su recorrido y hacer clic en el cheque verde.

Las figuras 104 y 105 mostrarán el procedimiento descrito anteriormente, así como el resultado final de agregar una ruta de trazo a un punto, que lo que hará no será más que dibujar el recorrido que siga dicho punto durante una simulación.

Figura 104. **Agregar una ruta de trazo a un ensamblaje**



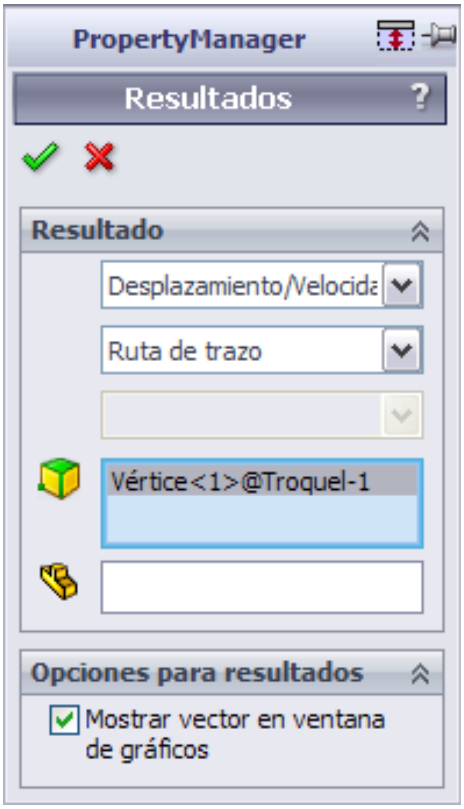
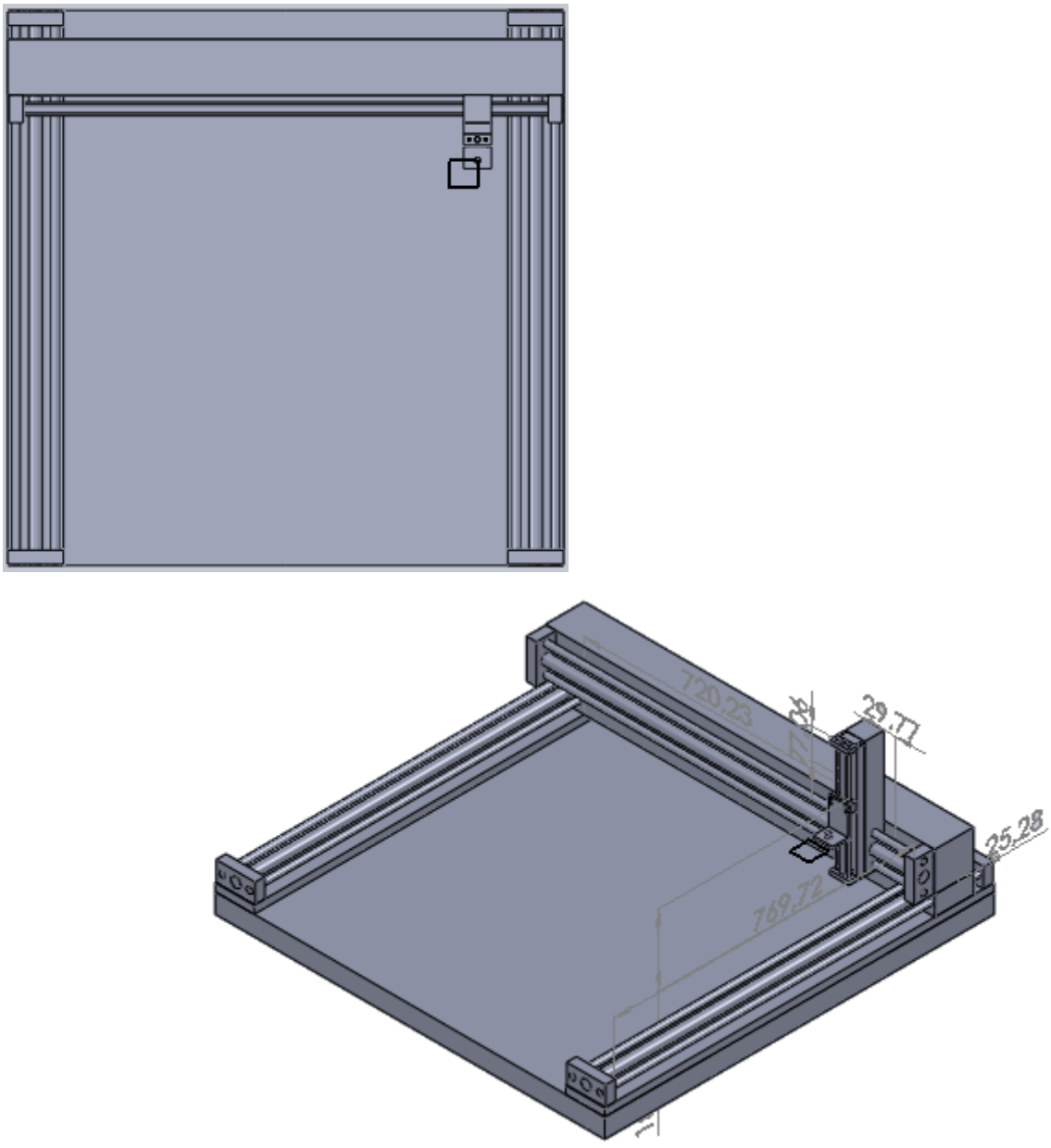
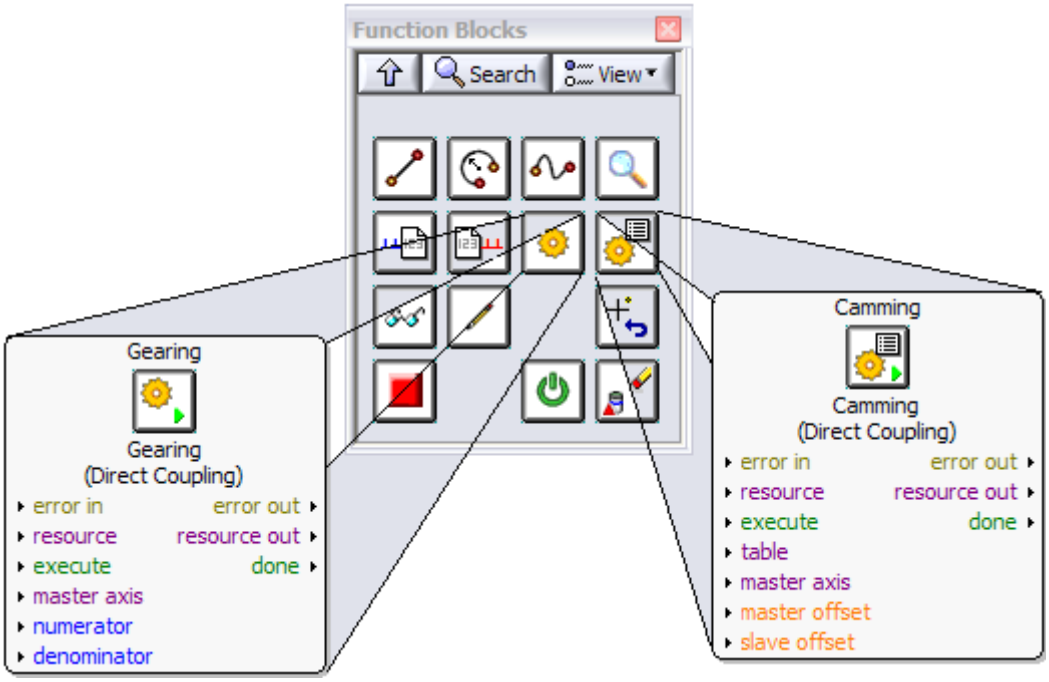


Figura 105. **Resultado de simulación de un cuadrado con ruta de trazo habilitado para el ensamblaje**



Dentro de las funciones de este módulo, existen dos funciones muy particulares que son para engranajes electrónicos (*electronic gearing*) y levas electrónicas (*electronic camming*) las cuales funcionan de manera similar y no se detallará mucho sobre ellas ya que dentro del desarrollo de la aplicación no serán utilizadas, la figura 106 ilustra dichas funciones.

Figura 106. **Bloques de función para engranajes y levas electrónicas**



La función de engranaje electrónico, es utilizada cuando se desea mantener una relación o una razón constante entre un eje llamado eje esclavo con respecto a otro eje llamado eje maestro, de tal manera que obedeciendo a esta relación, el eje esclavo responderá de acuerdo a lo que ejecute el maestro.

La relación entre ambos ejes puede ser directa o inversa y hace referencia a distancias recorridas únicamente. Por ejemplo en una relación directa (3:1) cuando el maestro se desplaza diez unidades el esclavo se desplazará treinta, por el otro lado, en la relación inversa el esclavo ejecuta únicamente una fracción de la distancia total del eje maestro para una razón 1:3 si el esclavo recorre diez unidades el esclavo recorrerá únicamente un tercio.

La leva electrónica funciona de la misma manera que los engranajes, la diferencia entre éstas dos radica en que la leva electrónica es utilizada cuando la relación que deben mantener el eje maestro y el esclavo cambiará conforme se lleva a cabo el proceso del que forman parte.

Otra forma de explicar esto es que los engranajes electrónicos mantienen una relación lineal durante toda la ejecución, mientras que la leva electrónica es utilizada cuando la aplicación requiere que la relación entre los ejes no sea constante o no mantenga una relación constante. Esta función utiliza, al igual que la función para contornos, una NI *softmotion table* pero en este caso se debe seleccionar cómo tipo de tabla *camming* y no *contour* en la configuración de la tabla.

Otro aspecto importante que merece ser mencionado es la capacidad que tiene la función *arc move* para ejecutar arcos en tres dimensiones tales como arcos helicoidales y arcos esféricos. Un movimiento helicoidal es un círculo que se extiende en forma de espiral a lo largo del eje Z, el mejor ejemplo de este movimiento es la forma que tienen las cadenas de ADN.

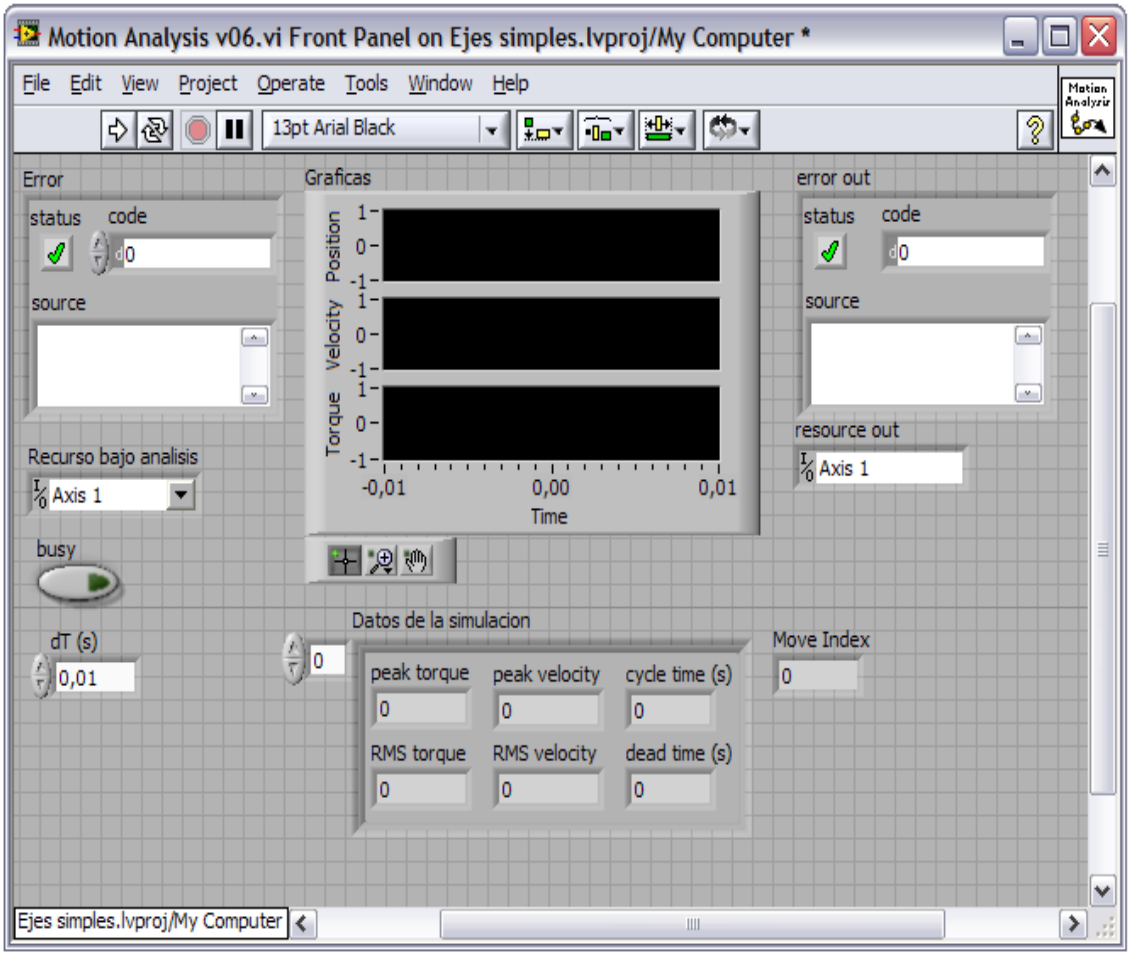
Arcos esféricos, éstos tienen la capacidad de formar sólidos utilizando el plano X, Y como base y brindando profundidad por medio del eje Z. La particularidad de este tipo de arco es que el eje Z puede ser rotado en diferentes formas para dar origen a distintos sólidos. Un último detalle respecto a los arcos en tres dimensiones es que desde que utilizan la función de arco requieren un NI *softmotion coordinate space* como recurso para funcionar.

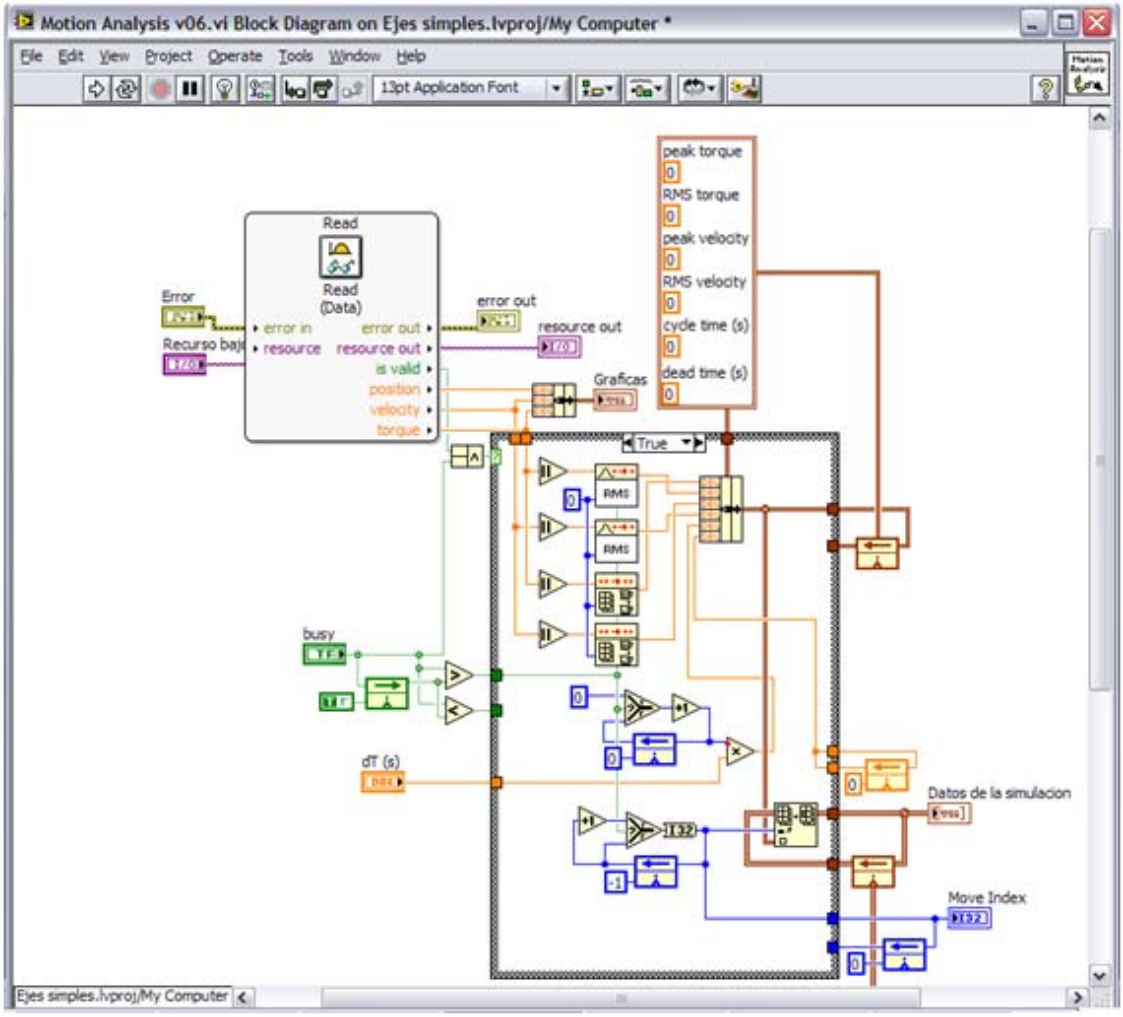
Finalmente se hablará sobre una herramienta que sirve para analizar ciertos parámetros y características de la simulación, los cuales pueden ser de utilidad a la hora de trasladar el prototipo virtual al plano físico. Esta herramienta es un VI llamado *Motion analysis.* Este VI puede ser descargado desde el sitio *web* de *National Instruments*.

Esta aplicación brinda al usuario gráficas de características de un motor como posición, velocidad y torque; también permite seleccionar el eje que estará bajo análisis, brinda datos del tiempo de ejecución y del tiempo de banda muerta entre ejecuciones subsecuentes y cuenta con un arreglo de datos para registrar datos como pico de torque, pico de velocidad, torque RMS y velocidad RMS. Los datos que proporciona esta aplicación deben ser tomados como lo que son, pruebas de simulación y no como datos 100% reales, sin embargo son de mucha ayuda a la hora de utilizarlos para dimensionamiento de motores.

De nuevo, es importante resaltar que estos datos deben ser estudiados cuidadosamente, si bien son consecuentes, responden y varían de acuerdo al cambio de distintos parámetros importantes dentro de la simulación, como la aceleración y fricción, refleja datos de escenarios ideales y por todos es sabido que en la vida real nada es ideal, la figura 107 muestra el *front panel* y el *block diagram* de este VI.

Figura 107. ***Front panel* y *block diagram* de la aplicación *Motion Analysis***




Motion Analysis v06.vi Block Diagram on Ejes simples.lvproj/My Computer *
File Edit View Project Operate Tools Window Help
13pt Application Font
Read
Read (Data)
error in
resource
error out
resource out
is valid
position
velocity
torque
Error
Recurso baja
error out
resource out
Graficas
peak torque
RMS torque
peak velocity
RMS velocity
cycle time (s)
dead time (s)
True
RMS
RMS
busy
dT (s)
Datos de la simulacion
Move Index
Ejes simples.lvproj/My Computer

5. DISEÑO Y PRESENTACIÓN DE LA APLICACIÓN PARA EL CONTROL DEL PROCESO DE TROQUELACIÓN

Ha llegado el momento de poner en práctica todo lo aprendido en el capítulo cuatro acerca de *Solidworks* y *LabVIEW* para lograr integrar las características de ambos programas de una manera exitosa y desarrollar una aplicación en *LabVIEW* capaz de controlar un prototipo virtual de una troqueladora en *Solidworks*.

Se dará inicio presentando el modelo final del prototipo virtual de la troqueladora creado en *Solidworks*, luego se explicará la forma en que se utilizó *LabVIEW* para desarrollar la aplicación de control, en esta parte se explicarán algunos detalles sobre las funciones de movimiento que no fueron mencionados en el capítulo anterior, además de otros detalles propios de la programación; y para terminar se presentará la aplicación completa, se explicará qué función cumple cada uno de los componentes dentro en ella y se mostraran imágenes de dicha aplicación puesta en marcha.

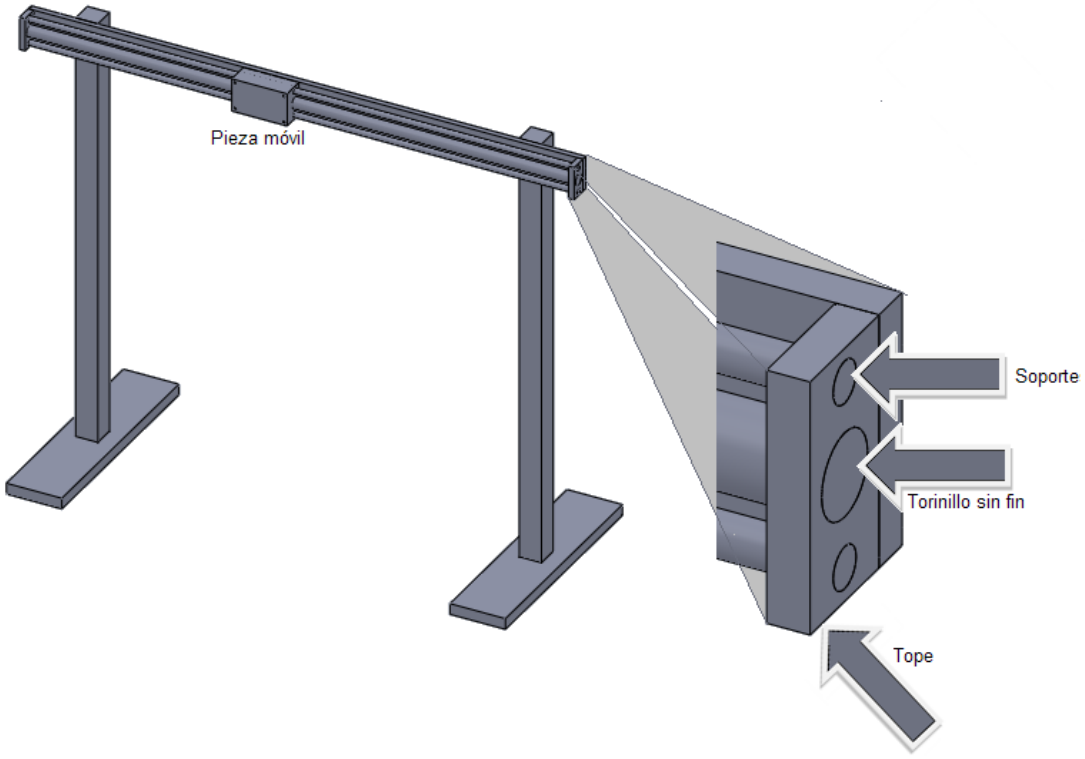
5.1. Prototipo virtual de la troqueladora

Solidworks es un programa cuya versatilidad permite al usuario realizar cualquier clase de diseño, desde un mecanismo sencillo y de pocas piezas como una tijera hasta un mecanismo complejo y de varias piezas como una máquina industrial.

Para el caso bajo análisis se realizó el prototipo de una troqueladora, que está compuesta por mecanismos para dar movilidad al troquel en tres dimensiones. El mecanismo utilizado para mover el troquel en la dirección deseada es un mecanismo conocido como fleje, que consta de un tornillo sin fin, dos soportes paralelos al tornillo sin fin, cuya superficie es lisa, dos topes para mantener tanto el tornillo sin fin como los soportes en sus respectivas posiciones y una pieza móvil que encaja con la rosca del tornillo sin fin y con los soportes.

Este mecanismo funciona de tal manera que cuando el tornillo rota en un sentido la pieza móvil, se desplazará en una dirección y cuando cambia el sentido de rotación del tornillo, la pieza móvil se desplazará en la dirección opuesta de tal manera que la máquina tenga libertad de movimiento. El mismo tipo de mecanismo fue utilizado para cada uno de los ejes de movimiento de la troqueladora, la figura 108 muestra un mecanismo de fleje.

Figura 108. **Mecanismo de fleje utilizado en la troqueladora**

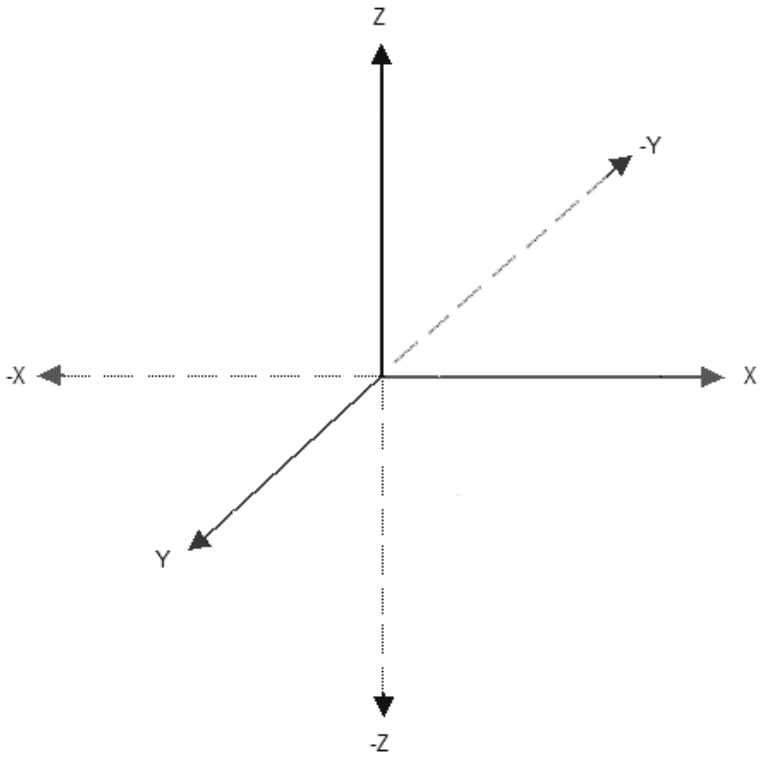


Para generar el movimiento sobre el eje Y la troqueladora cuenta con dos mecanismos de flejes montados paralelo uno respecto del otro, sobre la pieza que sirve de soporte para toda la máquina; los dos ejes restantes (ejes X y Z) cuentan con un solo mecanismo de fleje.

La pieza que sirve de soporte, posee un área total de 1000 milímetros cuadrados, es importante recordar que todas las medidas para este prototipo de troqueladora están en milímetros. Dicha pieza cuenta con un área de trabajo de 800 x 840 milímetros cuadrados, cuyo punto de inicio para trabajar se encuentra en las coordenadas 10 milímetros para X y -0.11 milímetros para Y.

La referencia de movimiento es la siguiente, sobre el eje X los movimientos hacia la izquierda serán negativos y hacia la derecha positivos, para el eje Y los movimientos hacia abajo serán positivos y los movimientos hacia arriba serán negativos y de la misma manera funcionará para el eje Z, la figura 109 ilustra.

Figura 109. **Marco de referencia para los movimientos de la máquina**



También cuenta con un espacio para deslizar la lámina o el material a troquelar de tal forma que el espacio vacío del área de trabajo sea reemplazado por el material a troquelar. Ésto se hizo asi, para que al momento de que el troquel perfore por completo el material, el usuario no tenga que preocuparse por perforar el suelo o la superficie sobre la que está apoyada la máquina.

El diseño de la troqueladora está hecho para trabajar con materiales de 20 milímetros de espesor como máximo, de utilizar un material con un espesor mayor, habría un problema a la hora de deslizar el material para que cubra el área de trabajo, ya que éste no entraría en dicha ranura.

Para mantener el material a trabajar fijo al soporte de la troqueladora, alrededor del área de trabajo se perforaron agujeros, por arriba y por abajo, para colocar tornillos con superficie de goma y de esta forma hacer una especie de prensa que evite que el material, al ser perforado o al momento de extraer el troquel se, mueva de su lugar. Para tener libertad a la hora de colocar los tornillos en la parte de abajo del soporte, a éste, se le han incluido tres barras, dos a los lados y una en la parte de atrás, de tal manera que la parte del frente tenga un espacio por el cual el usuario tenga acceso a la parte de abajo de la máquina.

Para el diseño final de la troqueladora, el orden de secuencia de pasos para cargar el material y dejarla lista para iniciar el proceso es:

- El primer paso sería contar con el material a troquelar, cuidando que no tenga un grosor mayor de 20 milímetros.
- A continuación deslizar el material por la ranura que se encuentra en la parte frontal de la máquina hasta que llegue al tope, el tope indica que la pieza a trabajar cubre por completo el área de trabajo.
- Asegurar la pieza a la base colocando los tornillos con goma en la superficie que estará en contacto con el material, para evitar deslizamiento. Una vez asegurado el material el usuario debe recordar que el área máxima de trabajo es de 800 x 840 milímetros cuadrados y que el diseño a realizar con la máquina no puede exceder dichas dimensiones.

Además es importante tener en cuenta que la superficie sobre la que esté montada la troqueladora debe contar con un recubrimiento de caucho o algún otro material que sirva para amortiguar la caída de la pieza finalizada ya que ésta será completamente cortada y caerá al suelo. Para evitar que la pieza caiga de golpe, se ha realizado un evento dentro de la programación de la aplicación que ayudará con este inconveniente.

Por último la figura 110 muestra el diseño final de la troqueladora, que básicamente es la misma que se presentó en el capítulo anterior, salvo algunas modificaciones que se hicieron para adecuarla y facilitar al usuario la operación de la misma.

Figura 110. **Diseño final de la troqueladora**

Todas las dimensiones de la máquina están en milímetros ya que *Solidworks* y *LabVIEW* trabajan en milímetros y de haber diseñado la máquina con dimensiones en metros un movimiento de 10 milímetros sería muy difícil de observar, además de consumir mayor recursos durante la simulación ya que la misma sería mucho más larga.

5.2. Descripción de la aplicación de control para la troqueladora

Ahora ha llegado la hora de explicar de qué manera se combinaron las funciones de movimiento, cuyas características ya fueron explicadas, para crear un panel de control para el proceso de troquelado que se llevará a cabo con el prototipo virtual presentado en la figura 110.

En algunos casos se hará mención de algunas herramientas de programación que se utilizaron y que no fueron descritas en párrafos anteriores, porque no forman parte del módulo *softmotion* o bien porque no son necesarias para establecer la comunicación entre *Solidworks* y *LabVIEW.* Sin embargo fueron vitales a la hora de programar la aplicación de control y en orden de explicar, de la mejor manera posible, cómo funciona la misma serán tocadas en su debido momento.

5.2.1. Funcionamiento del panel de control

Lo primero que se hará es explicar cómo funciona el panel de control para la troqueladora, es decir la secuencia de pasos que el usuario deberá seguir para iniciar el proceso de troquelado de una manera exitosa. Luego se explicará paso a paso cómo se programó la aplicación, para finalmente hacer la presentación del panel completo.

El panel de control para la troqueladora cuenta con cuatro pestañas, la primera de ella es la de controles principales, la siguiente es la de datos eje X, luego datos eje Y y la pestaña final datos eje Z. La pestaña de controles principales cuenta con un control para seleccionar el diseño que el usuario desea que la máquina realice, este control muestra una imagen de la pieza finalizada y algunos datos relacionados a las dimensiones de la pieza. Para el caso particular de este trabajo de graduación, el control presenta únicamente dos diseños para elegir, aclarando que se pueden programar tantos diseños como el usuario desee.

Continuando con los controles, el usuario tendrá a la vista cuatro controles más, un control llamado punto de inicio (Pto. de Inicio) el cual hace que la máquina se mueva de cualquier posición en el área de trabajo al punto establecido como el punto inicial de todos los diseños. El control de inicio que da la señal al mecanismo para empezar a troquelar el diseño seleccionado, el control de corte final que indica a la máquina cuando realizar el corte del contorno de la pieza y el control de paro que sirve para detener toda actividad de la máquina en cualquier instante durante el proceso de troquelado.

Además de controles, la pestaña de controles principales, cuenta con algunos indicadores entre los cuales encontraremos indicadores numéricos para la posición y velocidad en los tres ejes de movimiento e indicadores luminosos para los sensores de límite de posición en cada eje y para cada paso del proceso. Los indicadores para los sensores de límite de posición se activarán cada vez que la máquina supere los límites de posición establecidos, detendrán el accionar para el eje de movimiento al que estén ligados y no se desactivaran hasta que la posición deje de superar el límite establecido.

Los indicadores para monitorear el proceso son los siguientes, indicador de diseño que estará activado cada vez que el troquel se encuentre perforando el material en orden de realizar el diseño seleccionado, indicador de cero que permanecerá encendido mientras la máquina se sitúa en el punto de inicio establecido para el diseño , el indicador de troquel que estará encendido cada vez que el troquel sube hasta una posición definida para dejar de perforar el material y cuando está sobre el nuevo punto a perforar y desciende hasta atravesar por completo el material.

También encontrará el indicador de diseño completo que se activará una vez que el diseño del panel ha sido terminado, el indicador de corte final que se activará cuando la máquina está realizando el corte del contorno del panel una vez que el diseño interior del panel ha sido finalizado y por último encontrará el indicador de fin que se activa únicamente cuando se ha realizado tanto el diseño del panel como el corte del contorno del mismo e indica que el proceso de troquelación del diseño seleccionado ha concluido.

La segunda, tercera y cuarta pestaña muestran datos del desempeño del motor para los ejes X, Y y Z respectivamente, así como gráficas de la posición, velocidad y torque de los mismos. Estas pestañas están identificadas como datos Eje X, para la segunda, datos Eje Y, para la tercera y datos Eje Z para la cuarta y última pestaña del control.

Existe una secuencia de pasos que se deberán seguir para realizar el diseño seleccionado en el material a utilizar por el usuario. Realizar de forma indebida la secuencia de pasos a ser descrita en el siguiente párrafo, puede ocasionar que el diseño final del panel sea incorrecto y que el bloque de material quede inservible para un nuevo diseño, resultando en una total perdida de materia prima y tiempo de producción.

- Paso 1: seleccionar el diseño a realizar, como se dijo para este caso específico se cargaron dos diseños, el primero, diseño 1, hará una cuadrícula con bordes de 10 milímetros y espacio entre la cuadrícula de 20 milímetros, cuadrados de 105 milímetros y la pieza final tendrá un área de 500 milímetros cuadrados. El segundo diseño, diseño 2, entregará al final del proceso un panel con diez círculos de radio 20 milímetros en 2 filas de 5 círculos cada una, un circulo de radio 35 milímetros para la chapa del panel y tres rectángulos con área de 100 x 50 milímetros cuadrados y un área total de 635 x 600 milímetros cuadrados.

- Paso 2: una vez elegido el diseño, deberá ajustar el mecanismo al punto cero establecido por el fabricante, para lograr esto deberá presionar el botón con el nombre Pto. de inicio. Al momento de presionarlo debería de observar como el color del botón cambia a verde, inmediatamente después verá como el indicador de cero se activa y simultáneamente el mecanismo se moverá hasta llegar a la posición de inicio o cero. Una vez alcanzado este punto observará como el indicador de cero se desactiva, hasta este punto usted tiene el mecanismo ajustado y listo para iniciar a trabajar.

- Paso 3: iniciar el proceso de troquelado y para lograrlo deberá presionar el botón con el nombre Inicio. De nuevo cuando lo presione observará como el botón cambia a verde, al mismo tiempo que el indicador de diseño se enciende y el mecanismo inicia a moverse trazando un dibujo del diseño sobre el área de trabajo. Este dibujo que hará el mecanismo, será la simulación de la realización del diseño.

Durante la realización del diseño, que tarda aproximadamente 15 minutos y que el tiempo variará de acuerdo a las características de la computadora tales como microprocesador y memoria RAM, tienen lugar dos eventos uno la activación y desactivación del indicador de diseño y el segundo la activación del indicador de troquel.

Estos eventos obedecen a que cuando se está perforando el material en la forma seleccionada el indicador de diseño estará activo y el de troquel inactivo, pero al momento de terminar de perforar, por ejemplo un círculo, el mecanismo necesita levantar el troquel hasta un punto en el cual pueda moverse con completa libertad por toda el área de trabajo sin realizar perforación alguna sobre el material, llegar al nuevo punto para continuar con el diseño y descender para perforar el material y poder continuar con el mismo.

Entonces al momento de terminar con el diseño del círculo, el troquel debe levantarse y esta acción causa que el indicador de diseño se desactive y que el indicador de troquel se active y permanezca activo hasta que el troquel alcance la altura que le de libertad de moverse sin dañar el diseño.

Una vez logrado esto el indicador de troquel se desactiva y el indicador de diseño se activa de nuevo mientras el mecanismo se desplaza hasta la nueva posición para continuar con el proceso, habiendo alcanzado dicha posición, el troquel debe descender para perforar y esto activa la secuencia de desactivación del indicador de diseño y la activación del indicador de troquel. Cuando éste ha descendido hasta haber perforado el material y poder continuar con el diseño, se apagará el indicador de troquel y una vez más se activará el indicador de diseño. Dicha secuencia se repetirá tantas veces como el diseño lo requiera.

Finalizado el diseño podrá observar como el indicador de diseño completo se activa por un instante, y luego la máquina no hará nada más hasta que el usuario no presione el botón de corte final. Esto se pudo haber hecho de manera automática y el mecanismo inmediatamente después de finalizar el diseño interior de la pieza, cortaría el contorno y la pieza quedaría finalizada. Pero si observa detenidamente el diseño de la máquina verá que debajo del área de trabajo no hay nada que detenga la pieza cuando se corta el contorno y si no se coloca algo debajo de la pieza simplemente caería al suelo y podría dañarse.

Es por eso que para evitarlo se ha dejado a decisión del usuario cuando cortar el contorno de la pieza, de manera que tenga tiempo suficiente de colocar algo debajo de la pieza que evite que caiga al suelo y que a su vez no interfiera con el corte del contorno. Y desde que el diseño ha sido seleccionado y se conoce donde será el corte del contorno, colocar algo que interfiera con el corte final es muy improbable.

Una vez colocado el soporte para que la pieza no caiga al suelo, el cuarto paso sería presionar el botón de corte final el cual cambiará a color verde y a su vez iniciará a mover el mecanismo hasta el punto especificado en la programación e iniciará con el corte final o bien el corte del contorno de la pieza. Durante este evento tendrá lugar la activación y desactivación de los indicadores troquel y corte final, ya que de nuevo el mecanismo necesita descender el troquel antes de iniciar con el corte final y una vez completado el corte final, subirlo de nuevo. Una vez el troquel ha ascendido y se ha desactivado su respectivo indicador se encenderá por un momento el indicador de FIN que señala el final del proceso.

Recordando el orden de los pasos para programar:

- Paso 1: seleccionar el diseño deseado.
- Paso 2: ajustar el mecanismo al punto de inicio por medio del control con el mismo nombre.
- Paso 3: presionar el botón de Inicio para arrancar el mecanismo y realizar el diseño interior de la pieza.

Una vez concluido el diseño se cuenta con una pausa en el proceso controlada por el usuario para colocar algo que detenga la pieza para que al momento de reactivar el proceso y realizar el corte final, ésta no caiga directamente al suelo. El cuarto paso, estando segura la pieza para que no caiga al suelo, es presionar el botón corte final que volverá a poner en marcha el mecanismo para concluir la pieza y finalizar de esta manera el proceso.

5.2.2. Programación del panel de control

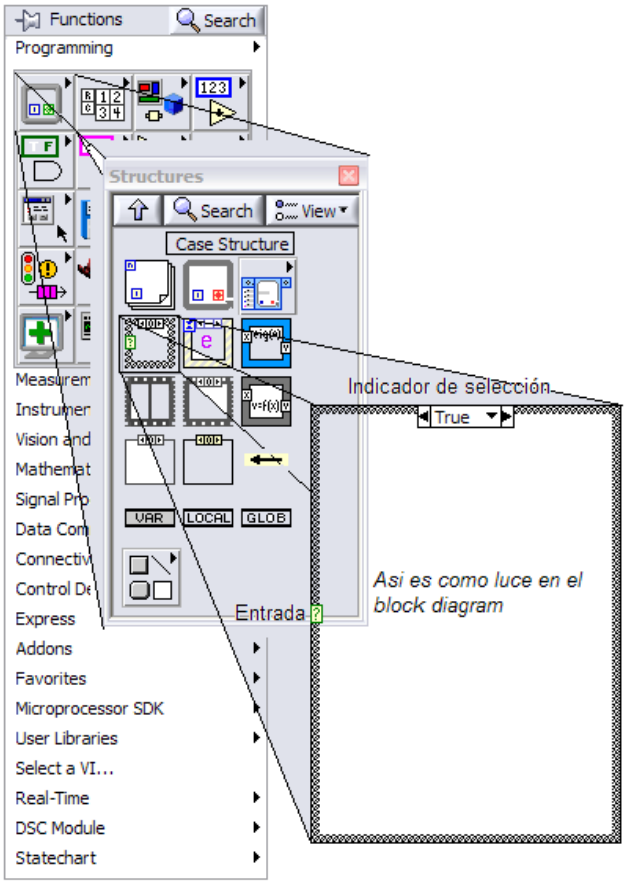
Hasta este punto se ha descrito cómo es que funcionará la aplicación una vez que se encuentre en ejecución. En los siguientes párrafos se hablará sobre cómo se logró que esto funcionara de manera adecuada integrando las funciones de movimiento que ya se conocen con otras herramientas propias de la programación las cuales también serán descritas en su momento.

La tónica para explicar que hace cada parte del programa será relacionar cada parte del proceso de troquelación que se describió en el apartado anterior, con su respectivo bloque de programación gráfica. De tal manera que sabiendo cómo debería de funcionar, se relacione de una mejor manera con la lógica que se utilizó para realizar la aplicación.

5.2.2.1. Programación del selector de diseños

La programación del selector de diseños se realizó con una herramienta conocida en la programación estándar o de texto como la sentencia *select case*, la cual posee una entrada alfa-numérica que tiene un número específico de posibles letras o dígitos a elegir y a cada uno de las posibilidades está ligado un proceso a realizar. Para el caso de *LabVIEW* se utilizo el análogo de esta función, conocido como *case structure* (*case*) y es ilustrado en la figura 111.

Figura 111. **Localización del *case structure***



Como se observa en la figura anterior el *case structure* cuenta con una entrada que acepta datos *booleanos*, enteros, cadenas de caracteres y números y de acuerdo al dato especificado en la entrada, esta estructura de programación sabe qué caso seleccionar y qué caso o casos ignorar. Dentro de esta estructura se coloca el código de programación para cada caso y el indicador ayuda a saber en qué caso está situado.

El indicador además de informar al programador en que caso se encuentra, sirve para agregar o eliminar casos y para saber qué caso es el caso por *default*, es decir el caso que aparecerá de primero. Esta estructura por definición siempre viene con dos casos en el indicador, los cuales son verdadero y falso y como se puede suponer la entrada es por definición *booleana*. Pero tanto la entrada como el indicador cambiarán de acuerdo al tipo de dato que se conecte a la entrada, para el caso de esta aplicación el dato conectado es numérico y en el indicador existirán los casos 0 y 1.

Para agregar más casos simplemente se debe hacer clic derecho sobre el indicador y de acuerdo al caso presente seleccionar el campo *add case before* o *add case after* para agregarlo antes o después respectivamente y de manera similar para cambiar el caso por default, el usuario deberá situarse sobre el caso deseado, hacer clic derecho sobre el indicador y seleccionar el campo *make this the default case*.

Además del case se utilizó un control y un indicador de imágenes. El control *text ring* se utilizó como la entrada del *case,* de tal manera que contará con la posibilidad de tener un control en el *front panel* con la capacidad de seleccionar texto, pero que el texto sea traducido a un número a la hora de ser leído por la entrada del *case* en el *block diagram,* de tal manera que a la hora de seleccionar por ejemplo, Diseño 2 en el *front panel*, la entrada del case en lugar de detectar la cadena de caracteres “Diseño 2”, lea el número 1 que es el asignado a dicho texto por el control.

De una manera similar se utilizó el indicador *pict ring*, el cual sirvió para poder mostrar la imagen del diseño de acuerdo a las opciones disponibles en el control *pict ring*. Entonces el indicador *pict ring* se conectó al control *pict ring* y a su vez a la entrada del *case*, logrando que ambas herramientas, tanto el indicador *pict ring* como el *case structure*, obedecieran y realizaran acciones especificas de acuerdo al diseño seleccionado del control *pict ring*.

Y de esta manera es cómo se realizó la programación del primer paso del proceso que es seleccionar un diseño de los diseños disponibles cargados en el panel de control de la troqueladora. Las figuras 112 y 113 ilustrarán la parte descrita tanto en el *front panel* como en el *block diagram.*

Figura 112. ***Case structure*, control *text ring* e indicador *pict ring* en el *block diagram***

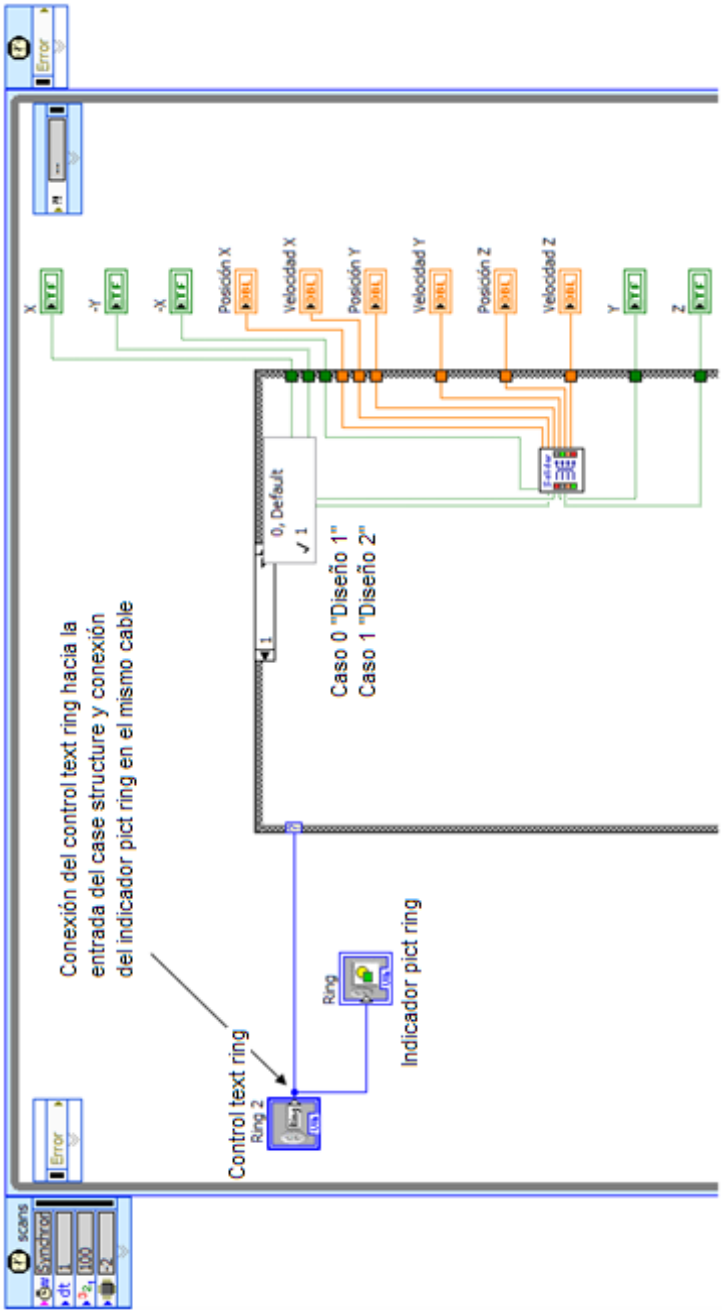
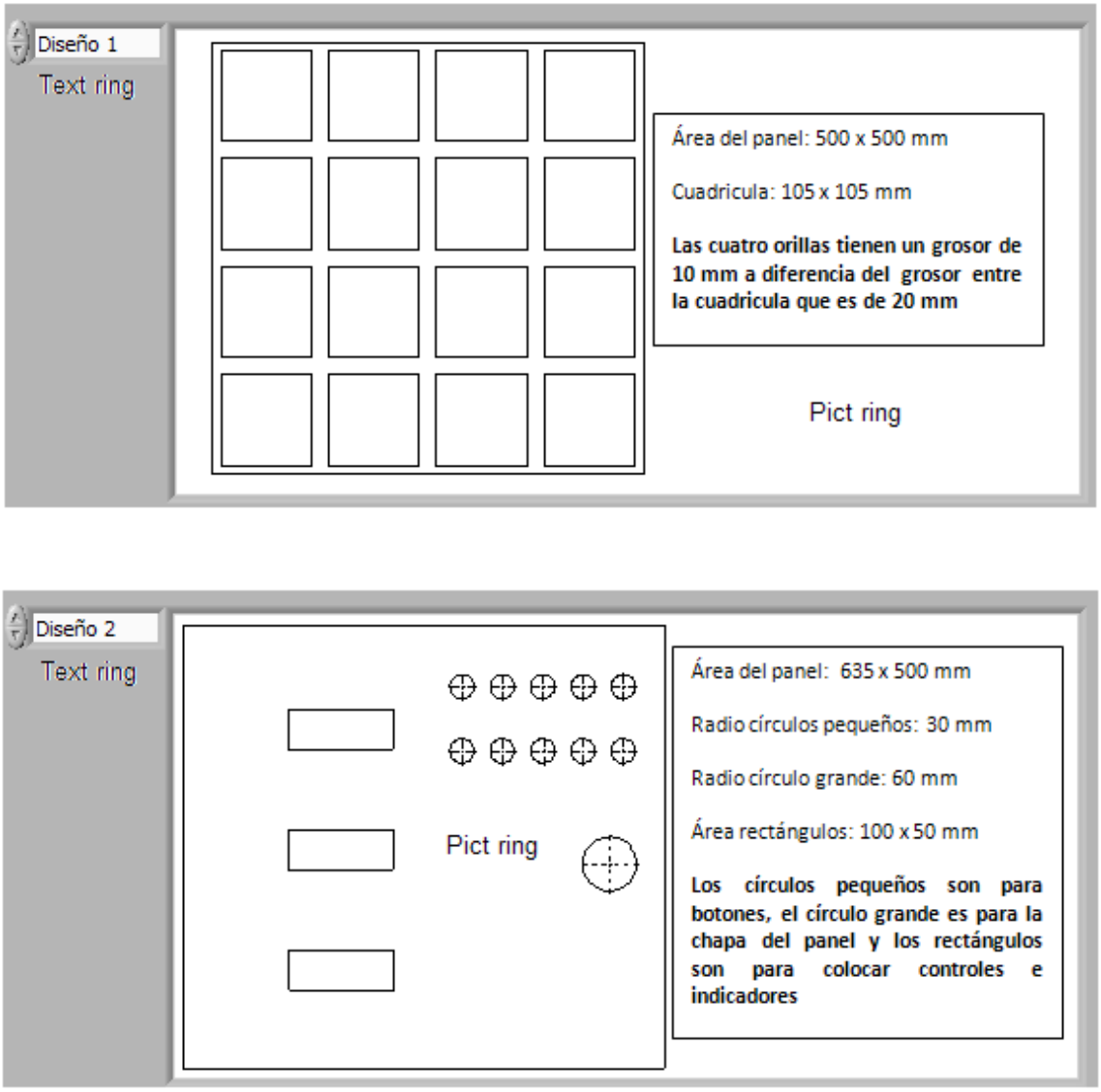


Figura 113. **Control *text ring* e indicador *pict ring* en el *front panel***



5.2.2.2. Programación de sensores de posición e indicadores de posición y velocidad

Para el caso de los sensores encargados de controlar que el movimiento de la máquina no exceda los límites establecidos, se utilizó exactamente la misma técnica descrita en el capítulo 4 para agregar sensores. En su lugar se describirá cómo se combinó el bloque *read* con el bloque, que también forma parte del módulo *softmotion*, *stop move*.

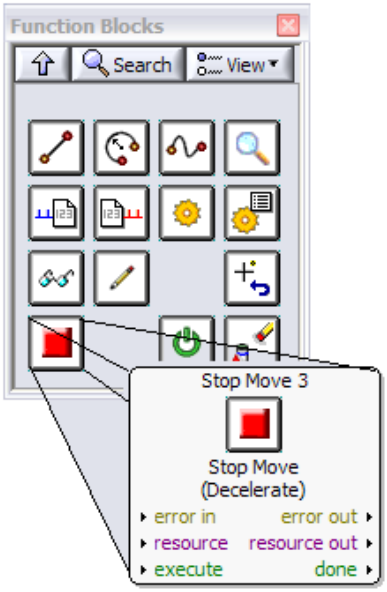
La función *stop move* sirve para detener el movimiento en un *softmotion axis* determinado por la entrada *resource* de la función, es decir si la entrada *resource* de la función *stop move* tiene ligado el eje X, entonces al momento de activar la función por medio de la entrada *execute*, ésta detendrá cualquier movimiento que esté realizando el eje X, además al igual que la mayoría de funciones de este módulo cuenta con varios métodos de operación entre los cuales está, detener el movimiento desacelerando, de inmediato y deshabilitar el drive.

El primer método como su nombre lo indica, detiene el movimiento por medio de una desaceleración que está estrechamente relacionada con la inercia que tenga el mecanismo a la hora de realizar dicha acción, es decir si el mecanismo tiene una velocidad alta, el tiempo que le tomará a la función detenerlo será mucho mayor que el que utilizaría para detener el mismo mecanismo con una velocidad mucho menor.

El segundo método detiene el mecanismo de una forma casi inmediata sin desaceleración alguna y el tercer método deshabilita el drive es decir deja el mecanismo desconectado del control de tal forma que se detendrá cuando la propia inercia del mecanismo ha disminuido lo suficiente como para continuar en movimiento.

La manera de seleccionar estos métodos de operación es igual que con las demás funciones, se sitúa sobre ella, hace clic derecho, selecciona el campo *select method* y finalmente utiliza la que considere más conveniente. Para este caso en particular se utilizó el método de paro inmediato. La figura 114 ilustrará como luce esta función.

Figura 114. **Función *stop move* del módulo *softmotion***

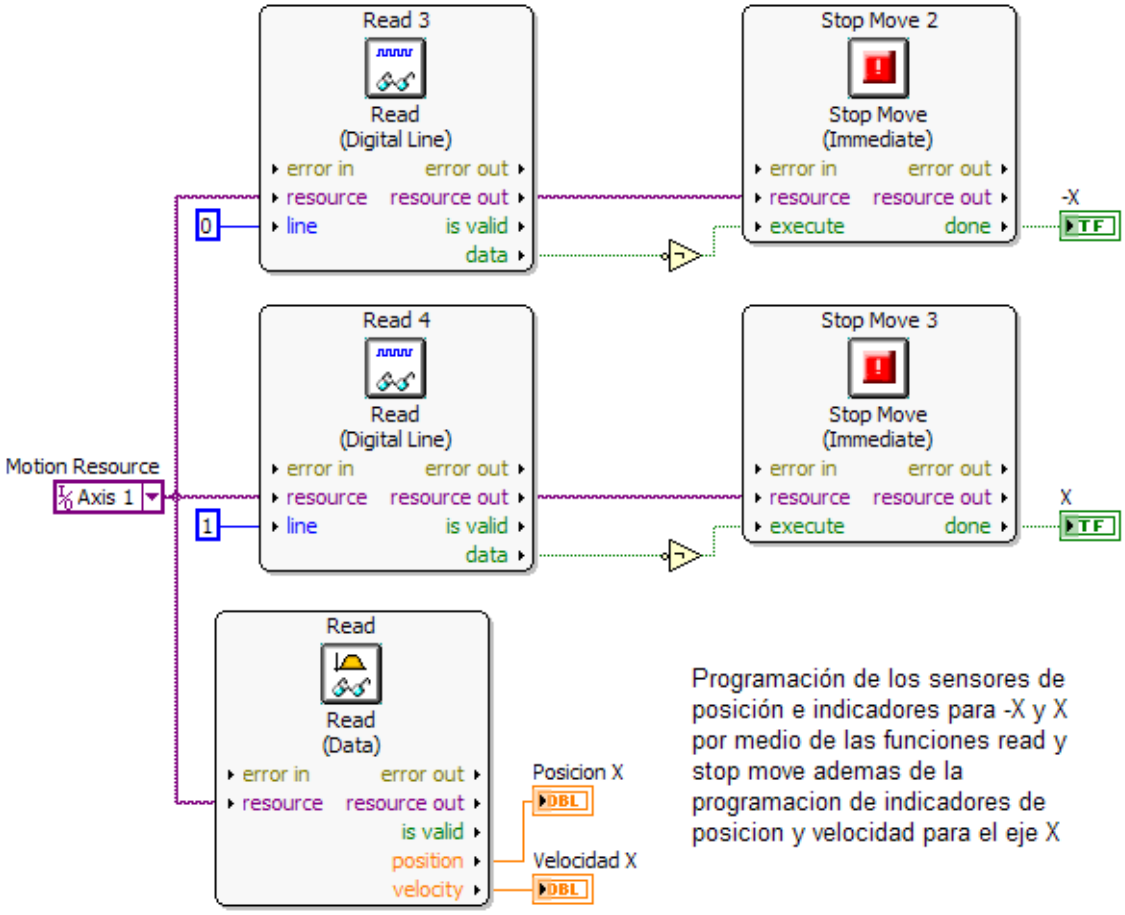


Ahora se mostrará cómo se combinaron ambas funciones para lograr que los sensores activaran un indicador y a su vez detuvieran el movimiento del eje al cual se encuentran ligados. Lo primero fue seguir el procedimiento para agregar sensores que a grandes rasgos involucra crear cotas inteligentes, añadir los sensores al ensamblaje, colocar las condiciones de activación, todo esto hecho en *Solidworks*, mapear los sensores y finalmente utilizar la función *read* en modo de línea digital para leerlos por el otro lado hecho en *LabVIEW*.

Entonces lo que se hizo fue colocar ambos bloques en serie para que cuando la función *read* fuese activada por la condición especificada, ésta enviará un flanco ascendente a la entrada *execute* de la función *stop move* y detuviese cualquier movimiento ejecutado por el eje al que se encuentra vinculada, inmediatamente después de haber cumplido con su tarea, detener el eje, la función *stop move* entregará en la salida *done* un flanco ascendente que es aprovechado para activar el indicador presente en el *front panel* de la aplicación para el control de la troqueladora.

El mismo procedimiento se realizó para cada uno de los indicadores de posición para cada eje de movimiento de la máquina. En total se pueden encontrar, en la pestaña de controles principales, cinco indicadores de posición uno para la posición –X, otro para su contraparte X, uno para –Y, uno para Y y finalmente un indicador para –Z. La figura 115 muestra una parte de la programación que se utilizó para este procedimiento.

Figura 115. **Programación de sensores de posición**



Con respecto a la programación de los indicadores de posición y velocidad lo único que se hizo fue utilizar un bloque *read* con el método *data* activado, especificar en la entrada *resource* sobre que eje debía extraer datos y colocar indicadores en las salidas *position* y *velocity* respectivamente para conocer la posición y velocidad de dicho eje. La programación de esto se puede observar en la figura anterior.

5.2.2.3. Programación del ajuste al punto de inicio

La programación del ajuste al punto de inicio se realizó de una manera muy sencilla, lo único que se hizo fue colocar una función de línea recta cuya entrada *resource* fue ligada a un *coordinate space* de dos ejes (X, Y) para llevarlos al punto de inicio que es (10;-0.11) con una sola función. Las coordenadas para X y Y fueron colocadas como valores constantes, razón por la cual no aparecen en la pestaña de controles principales.

Para activar el indicador correspondiente a esta acción dentro del proceso, se utilizó una característica propia de las funciones del módulo *softmotion*, que se explicará cómo activarla a continuación. Esta característica consiste en agregar una salida más a la función, dicha salida será agregada con el nombre *busy* y lo que hace es mantener un flanco positivo mientras que la función está realizando su tarea.

Para activarla hay que hacer doble clic sobre la función, lo que abrirá una ventana con las entradas y salidas disponibles de la función, se busca la salida *busy*, se activa la casilla junto a ella y se hace clic en *OK*. Finalmente a esta nueva salida se le cablea un indicador (indicador cero en la pestaña de controles principales) el cual se encenderá, como se dijo hace un instante, cuando la función se encuentre realizando su tarea, que en este caso será llevar el troquel al punto de inicio establecido.

5.2.2.4. Programación de los diseños que realizará la troqueladora

En los siguientes párrafos se explicará la lógica que se utilizó para programar cada uno de los dos diseños cargados en la aplicación de control para la troqueladora. Trataré de llevar al lector paso a paso sobre lo que se hizo y porque se hizo de esa manera.

El primer diseño fue programado con un grupo de herramientas, mientras que el segundo se realizó con otro grupo y se hizo de tal manera que el lector pudiese tener en cuenta que ventajas y desventajas brinda programar de una forma o de otra. Además, se pretende que esto sirva para iniciar con la creación de un criterio de que herramientas son más convenientes de acuerdo a las necesidades requeridas para una aplicación específica.

5.2.2.4.1. Programación del diseño 1

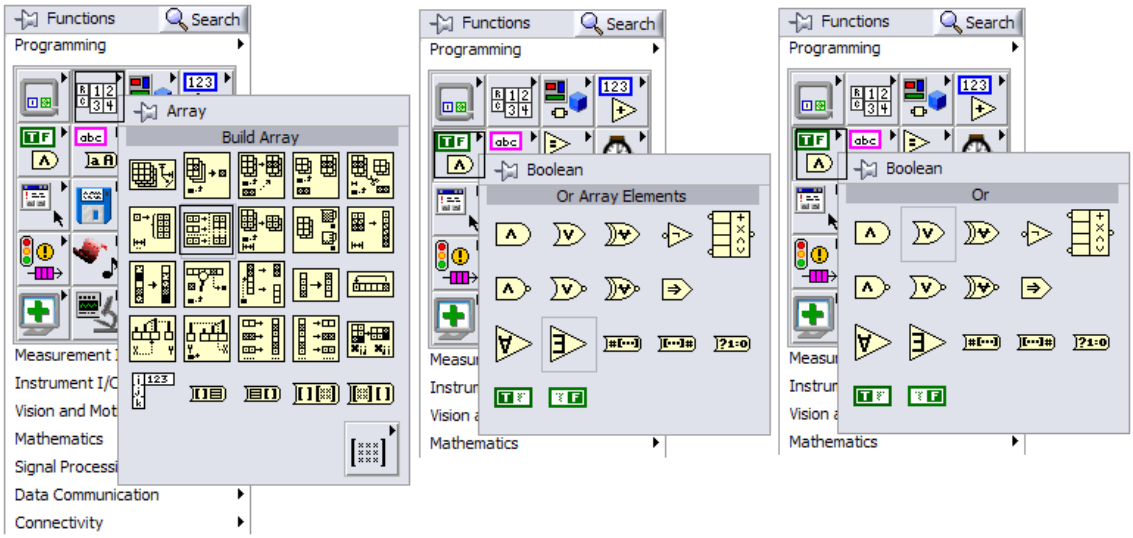
El primer diseño para elegir en la aplicación de control es un diseño para realizar un patrón de cuadrícula con bordes externos de 10 milímetros, bordes entre la cuadrícula de 20 milímetros, cuadrados de 105 milímetros cuadrados y área total de 500 milímetros cuadrados. Además el punto de inicio para el diseño se encuentra en 10 milímetros para el eje X y -0.11 milímetros para el eje Y.

La programación del diseño incluyó las siguientes herramientas de programación, la función para líneas rectas, la función para contornos, la función para construir arreglos, la función de arreglos *booleanos or* y una función *booleana* sencilla *or*.

La función para líneas rectas, como su nombre lo indica traza líneas rectas sobre el eje o ejes que tenga ligados a la entrada *resource*, para este caso se utilizó para subir y bajar el troquel, así como para mover el mecanismo de acuerdo al diseño. La función para contornos se encarga de realizar un trazo sobre el eje o ejes que tenga ligados a la entrada *resource*, siguiendo las coordenadas especificadas en una tabla la cual está ligada a la entrada *table.* Esta función se utilizó para el trazado de los cuadrados del diseño.

También se utilizaron otras funciones que no habían sido mencionadas como la función *build array*, esta función se encarga de recibir una cantidad n de entradas de datos de diversos tipos y convertirlas en un arreglo de n elementos que contiene los datos de cada entrada individual conectada a la función. La función *or array* es otra función nueva que se utiliza para crear una función *booleana or* de tantas entradas como el usuario desee, la característica principal de esta función es que su entrada deber ser del tipo *array*, es decir únicamente acepta arreglos de datos, mas nunca entradas sencillas y finalmente la función *booleana or*, esta es una función *or* básica de dos entradas. Las tres funciones son ilustradas en la figura 116.

Figura 116. **Funciones *build array*, *or array* y *or***



Para programar esta aplicación, lo primero que se hizo fue colocar una función de línea recta con un *coordinate space* conteniendo los ejes X y Y para poder mover el mecanismo hasta el primer punto donde se iniciará el diseño, partiendo del punto de inicio (10; -0.11). La coordenada de dicho punto es (0, 10.11) milímetros, lo cual indica que el eje X se desplazará diez milímetros hacia la izquierda y el eje Y 10 milímetros hacia abajo ambos en forma simultánea. La entrada *execute* de esta primera función de línea recta, se encuentra conectada al botón Inicio que aparece en la pestaña de controles principales de manera que al presionarlo, éste sea el primer movimiento que haga.

La salida *busy*, está conectada al indicador de diseño de la pestaña controles principales, de tal manera que mientras la función de línea recta está llevando el mecanismo del punto (10; -0.11) al punto (0;-10.11), dicho indicador permanezca activo.

La salida *done* está conectada a la entrada *execute* de la segunda función de línea recta, para que al concluir el trabajo de la primera función de línea recta el flanco positivo entregado por *done* habilite a la segunda función de línea recta.

La segunda función de línea recta, controla el eje Z y por lo tanto en este punto del código se encargará de bajar el troquel del punto 0 al punto -142, que es la distancia a la cual logrará atravesar por completo el material colocado en el área de trabajo, de nuevo a esta función se le habilitó la salida *busy* para que mientras se encarga de bajar el troquel el indicador troquel en la pestaña de controles principales esté encendido.

Hasta ahora se han realizado en serie los siguientes pasos, al presionar el botón inicio la primera función de línea recta moverá el mecanismo del punto (10; -0.11) al punto (0; 10.11), luego la segunda función de línea recta bajará el troquel desde 0 hasta -142 milímetros. El tercer paso para el diseño del cuadriculado seria el trazo del primer cuadrado y para esto se utiliza la función de contorno, cuya entrada *execute* será habilitada por la salida *done* de la función de línea recta que bajará el troquel. A la función de contorno se le han ligado un *coordinate space* de dos ejes (X y Y) y una tabla que realizará el cuadrado de 105 milímetros en la parte de las entradas.

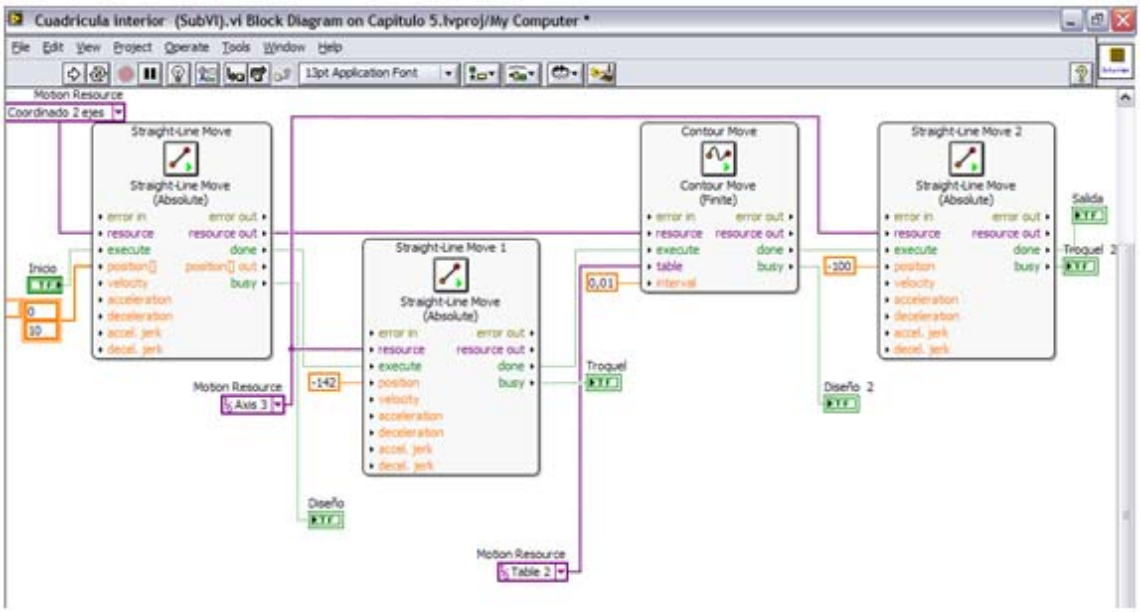
Por el otro lado las salidas utilizadas en esta función son una vez más la salida *busy* y la salida *done.* La salida *busy* es utilizada para mantener activo el mismo indicador de diseño, que mantuvo activado la primera función de línea recta cuando movió el mecanismo del punto de inicio al punto (0; 10.11), al realizar el trazo del cuadrado de 105 milímetros cuadrados, finalmente la salida *done* es utilizada para habilitar la tercer función de línea recta utilizada para subir el troquel una vez que ha terminado de trazar el primer cuadrado del diseño.

La tercera función de línea recta, al igual que la segunda, está ligada al eje Z para controlar el movimiento del troquel, pero en esta ocasión en lugar de bajarlo, tiene la función de subirlo hasta el punto -100. La razón para subirlo hasta -100 en lugar de subirlo hasta el punto 0 es por ahorro en tiempo de ejecución, ya que al subir el troquel a -100 se asegura que el mecanismo pueda moverse libremente por el área de trabajo sin dañar el diseño.

Una vez más se utiliza la entrada *busy,* de la tercera función de línea recta, para mantener activado, durante la ejecución del movimiento hacia arriba del troquel, el mismo indicador que se activó cuando el troquel se llevó del punto 0 hasta el punto -142 y de nuevo el flanco positivo que entrega la función en la salida *busy*, al haber concluido su trabajo podrá ser utilizado para habilitar otro movimiento.

Hasta este punto se ha creado un cuadrado de 105 milímetros cuadrados siguiendo la siguiente secuencia, mover el mecanismo del punto de inicio al punto (0; 10.11), luego se bajó el troquel hasta el punto -142, en seguida se realizó el trazado del cuadrado, para finalmente volver a subir el troquel al punto -100 y darle libertad al mecanismo de moverse hasta el siguiente punto sin dañar el diseño, el código se muestra en la figura 117.

Figura 117. **Código para el trazo del primer cuadrado**



Para realizar un cuadrado se usaron tres funciones de línea recta y una función de contornos, entonces si se deben realizar 16 cuadrados, que son los que componen el primer diseño, se deberán utilizar 48 funciones de línea recta y 16 funciones de contorno, que representan muchas funciones, muchos cables y una gran cantidad de espacio.

Es en este punto donde toma importancia el concepto de *subVI*, que cómo recordará, cuando se describió en el capítulo tres, se dijo que un *subVI* es una subrutina que puede ser utilizada las veces que sea necesario en los *VIs* que lo requieran, y para este caso es una herramienta ideal ya que el diseño es repetitivo, entonces antes de continuar con la explicación sobre cómo se realizó la programación del primer diseño, en el siguiente apartado se mostrará cómo crear un *subVI*.

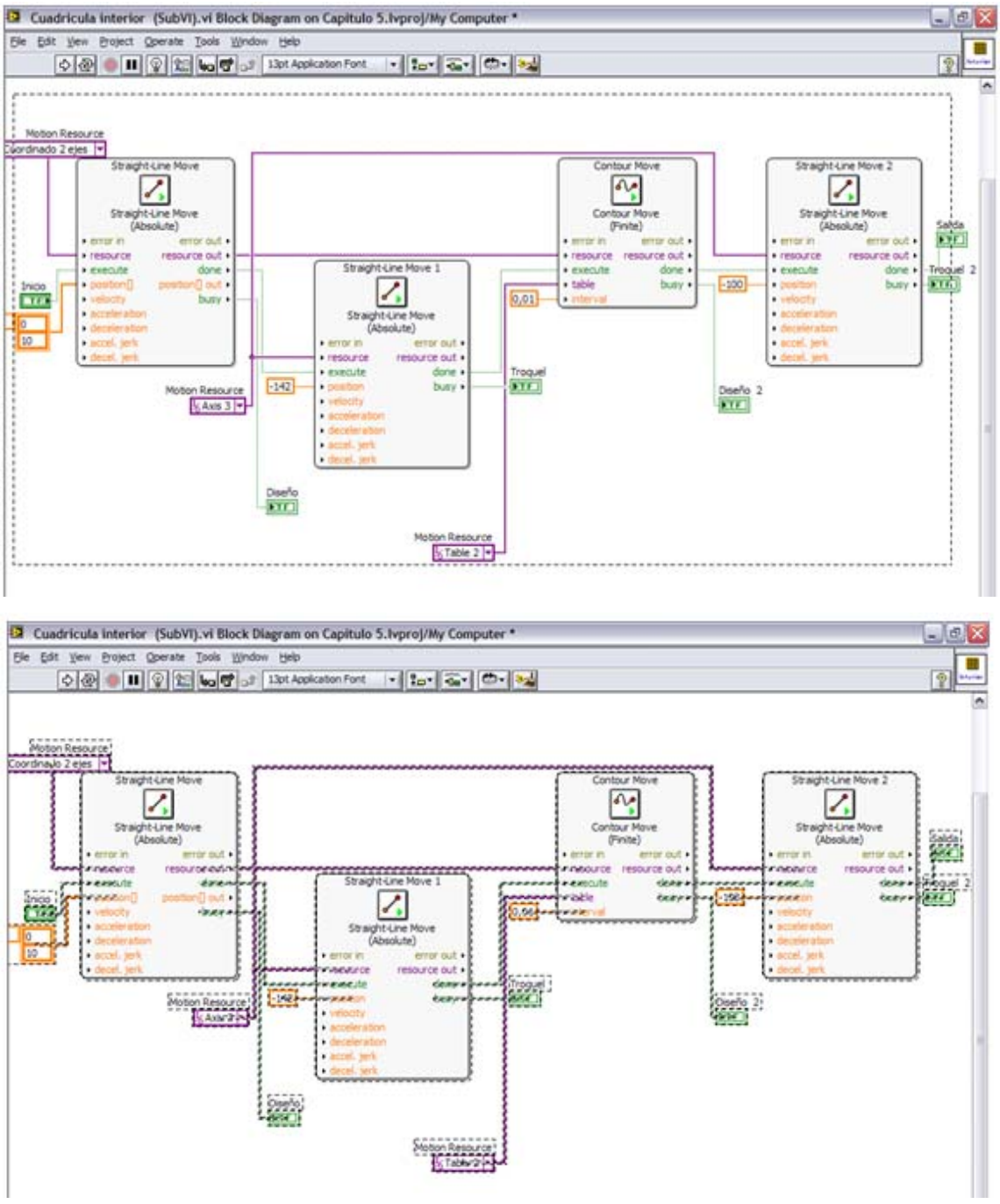
5.2.2.4.1.1. Creación de un *subVI*

En muchas ocasiones la mayoría de personas que programan en *LabVIEW,* al crear sus códigos, de hecho están trabajando con *subVIs*, pero no lo saben es por eso que a continuación se explicará el proceso para la creación de un *subVI* utilizando como ejemplo el código explicado en el párrafo anterior para la creación de un cuadrado.

El primer paso es tener el código de lo que se desea que haga el *subVI* listo y seguro de que funciona como se desea, recuerde que el *subVI* hace la función de una subrutina en programación de texto, para este caso el *subVI* tendrá la capacidad de mover el mecanismo hasta un punto específico y mantener activo un indicar mientras lo hace, bajar el troquel y mantener activo un indicador mientras lo hace, trazar el cuadrado y mantener activo un indicador mientras lo hace y finalmente volver a subir el troquel y mantener activado un indicador mientras lo hace.

Una vez con el código gráfico listo y seguro de que ejecuta una función específica como se desea, el siguiente paso es seleccionar todo el código, para seleccionar código basta con hacer clic, mantenerlo presionado y buscar crear un cuadrado alrededor del código que se quiere convertir en *subVI.* El usuario sabrá que el código ha sido seleccionado ya que al soltar el botón del ratón, una línea punteada, que se mueve emulando hormigas, rodeará el código tal y como se observa en la figura 118.

Figura 118. **Cómo seleccionar código en el *block diagram***



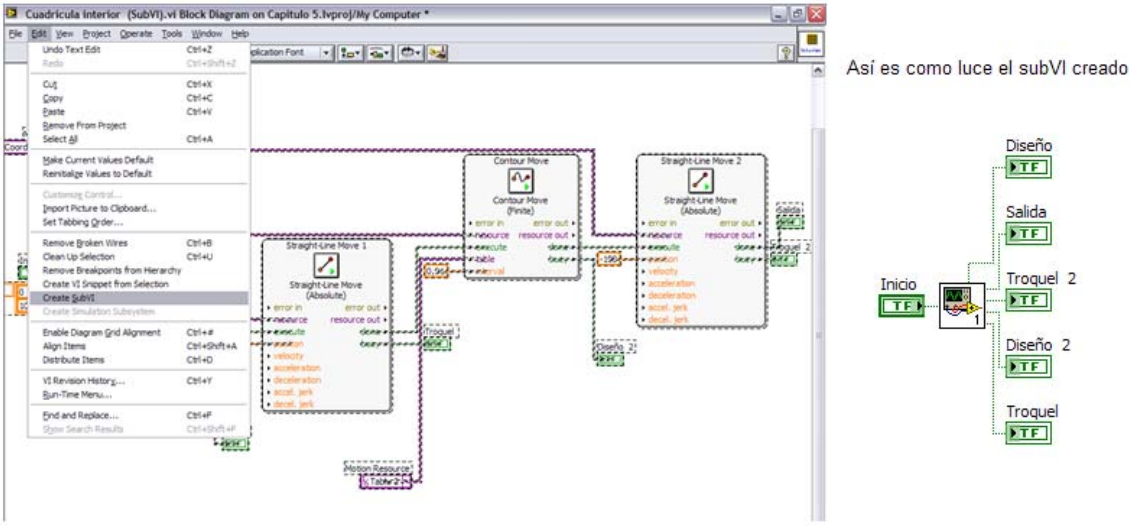
Cuadricula interior (SubVI).vi Block Diagram on Capitulo 5.lvproj/My Computer *
File Edit View Project Operate Tools Window Help
13pt Application Font
Motion Resource
Coordinado 2 ejes
Straight-Line Move
Straight-Line Move (Absolute)
Straight-Line Move 1
Straight-Line Move (Absolute)
Contour Move
Contour Move (Finite)
Straight-Line Move 2
Straight-Line Move (Absolute)
Inicio
Salida
Troquel 2
Troquel
Diseño
Diseño 2
Motion Resource
Axis 3
Motion Resource
Table 2
-142
0,01
-100
0
10

Ahora que se tiene seleccionado el código el tercer paso es dirigirse al menú *Edit* y seleccionar el campo *create subVI.* Una vez hecho esto podrá observar como la computadora queda estática por un momento y de pronto todo el código gráfico queda convertido en una cajita con un ícono y con todos los controles e indicadores conectados a ella.

Un punto importantísimo y que no se mencionó en el primer paso, es que el *subVI* será creado con las entradas y salidas que tengan cableados controles e indicadores respectivamente, por ejemplo si el código del cuadrado no hubiese tenido un control para la entrada *execute,* el *subVI* seria creado sin una conector llamado *execute*. Las constantes por su parte no son tomadas en cuenta en las entradas y salidas del *subVI* porque son constantes.

Entonces cumpliendo todos los pasos descritos hasta ahora *LabVIEW* automáticamente creará un *subVI* el cual luce como el que se muestra en la figura 119. Otra característica importante del *subVI* es que se puede modificar el ícono que aparece dentro del cuadro que representa el *subVI* y también se puede modificar el orden de las terminales o conectores con que cuenta el *subVI.*

Figura 119. **Creación de un *subVI***



Creación del subVI

Como puede observar el *subVI* ha sido creado con un control para la entrada *execute* e indicadores para troquel y diseño que corresponden a las salidas *busy* de las funciones restantes y por último el indicador salida que corresponde a la salida *done* de la tercer función de línea recta ligada al eje Z.

Para modificar el ícono que representará al *subVI* es necesario hacer lo siguiente, doble clic sobre el *subVI*, lo que desplegará en pantalla el *front panel* del *subVI*, luego sobre el ícono del *front panel*, en la esquina superior derecha hacer clic derecho y seleccionar el campo *edit icon*, lo que mostrará en pantalla una aplicación similar a la conocida aplicación para edición de imágenes de *Windows paint brush,* llamada *icon editor,* con la que podrá modificar el ícono de la manera deseada, una vez a gusto con el diseño simplemente hará clic en *OK* y verá como el ícono en el *front panel* cambia, las figuras 120 y 121 ilustran el proceso.

Figura 120. **Modificación del ícono del *subVI***

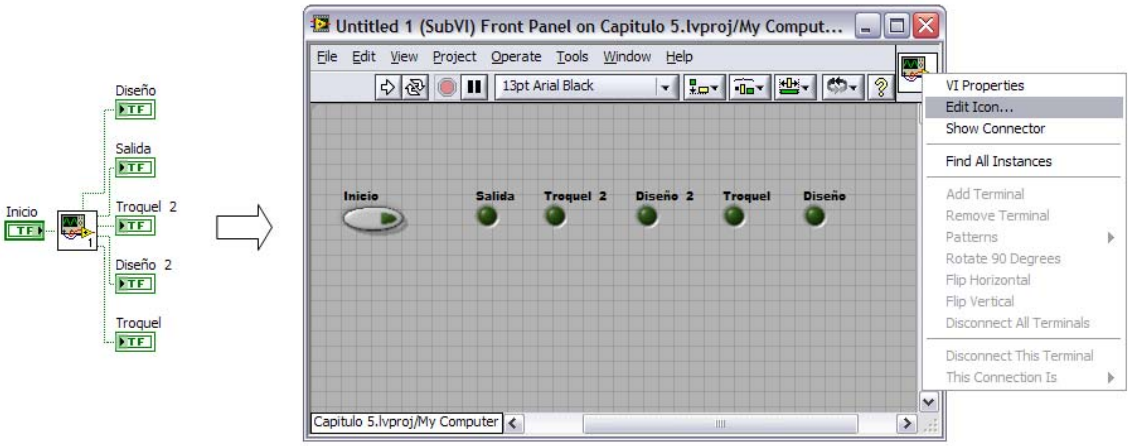
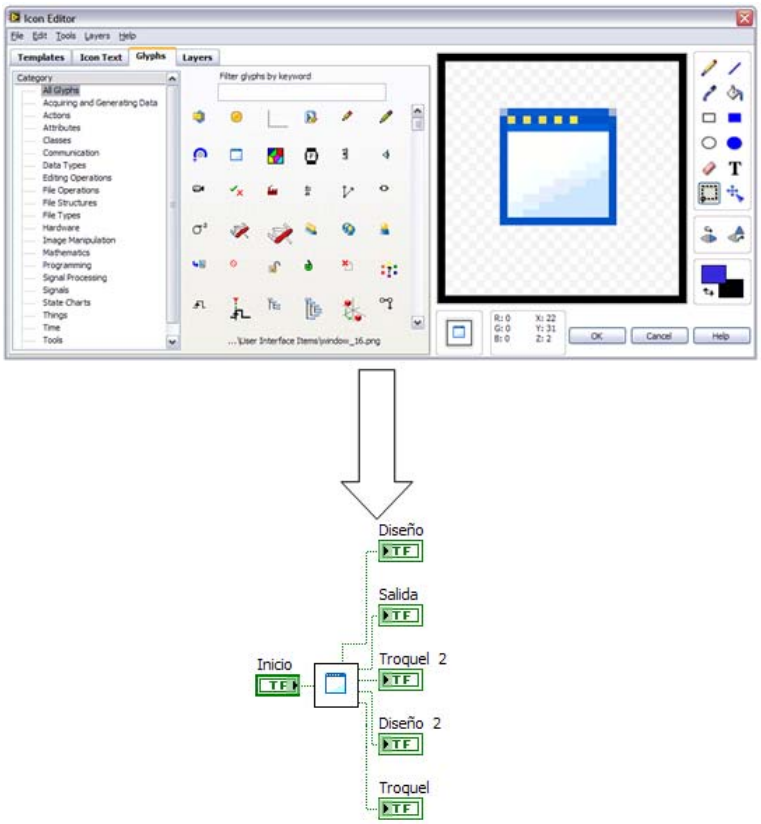


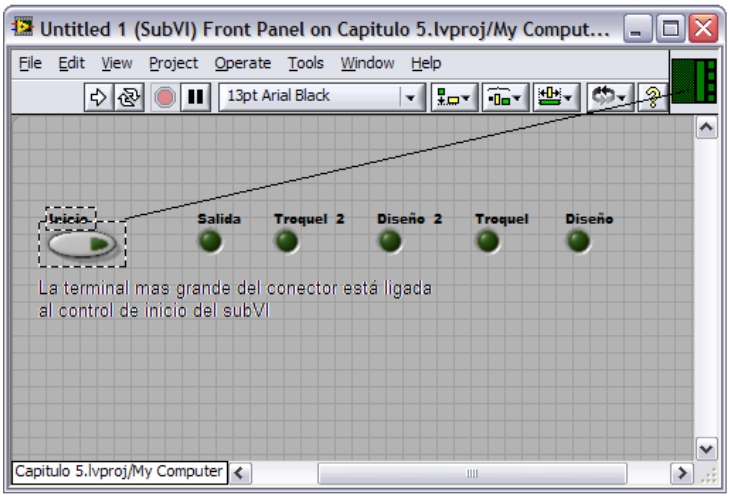
Figura 121. **Aplicación para modificar el ícono del *subVI***



La otra característica, con méritos suficientes para ser explicada, acerca de los *subVIs* es la capacidad para cambiar el orden de las terminales del mismo. Para hacer esto el procedimiento es el siguiente, primero doble clic sobre el *subVI* para tener en pantalla el *front panel*, luego clic derecho sobre el ícono, en seguida seleccionar el campo *show connector*.

De forma inmediata luego de seleccionar show connector, verá como el ícono es sustituido por una especie de rompecabezas, esto nos indica cómo se encuentran conectadas las terminales del *subVI*. Si es comparado el conector del *subVI* con el orden de las terminales, notará que coinciden ya que del lado izquierdo el conector tiene sólo una terminal y del lado derecho tiene 5, para identificar cual es cual, basta con hacer clic sobre una de las divisiones del conector y verá como en el *front panel* un rectángulo punteado rodea a la entrada o salida ligada a ese conector tal y como lo muestra la figura 122.

Figura 122. **Conector de las terminales del *subVI***



Para modificar este orden lo único que se debe hacer es clic derecho sobre la imagen del conector y seleccionar el campo *disconnect all terminals*, de esta forma las terminales cambiarán a color blanco y el usuario tendrá libertad para ligar las terminales con el conector como mejor le convenga.

Para conectar una terminal específica del conector con un control o un indicador en el *front panel*, lo único que debe hacer es dirigirse hacia el conector y hacer clic sobre la terminal a la que desea ligar algún elemento, luego dirigirse hacia el *front panel* y si hacer clic sobre el elemento que desea ligar a esa terminal. Sabrá que lo ha hecho de manera correcta si la terminal en el conector cambia de blanco a verde y el elemento en el *front panel* es rodeado por una línea punteada, la figura 123 y 124 ilustran la manera de hacerlo.

Figura 123. **Desconexión de las terminales del conector**

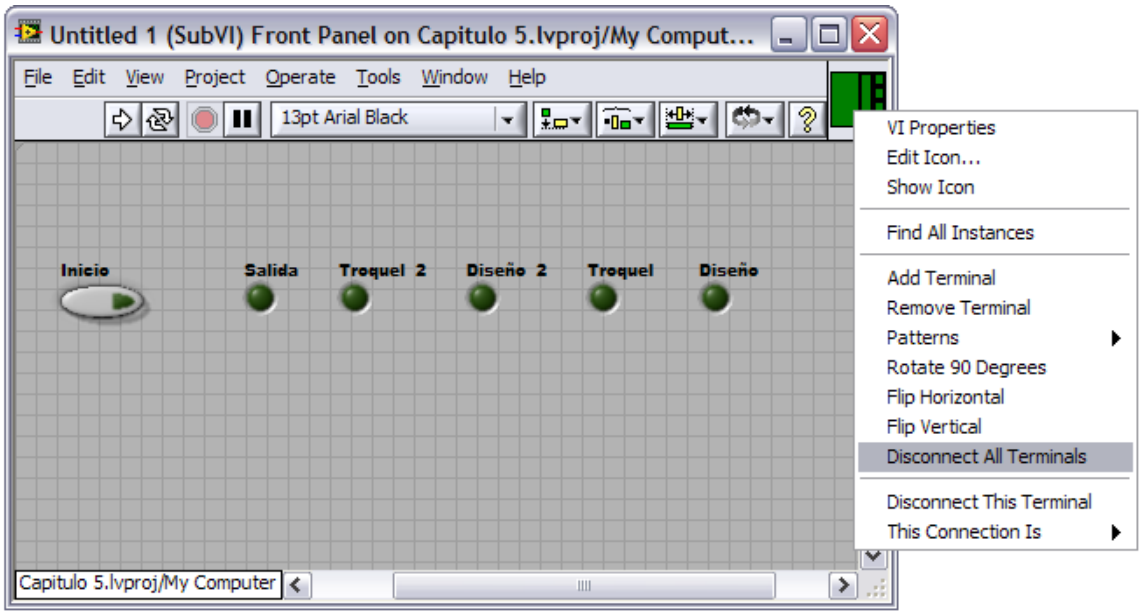
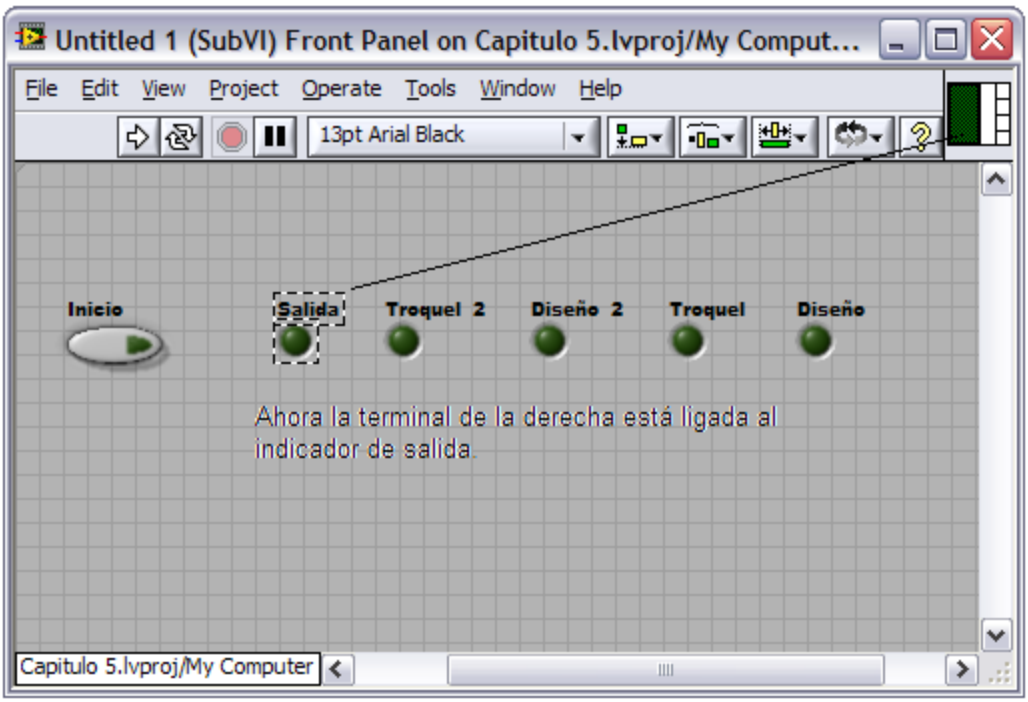


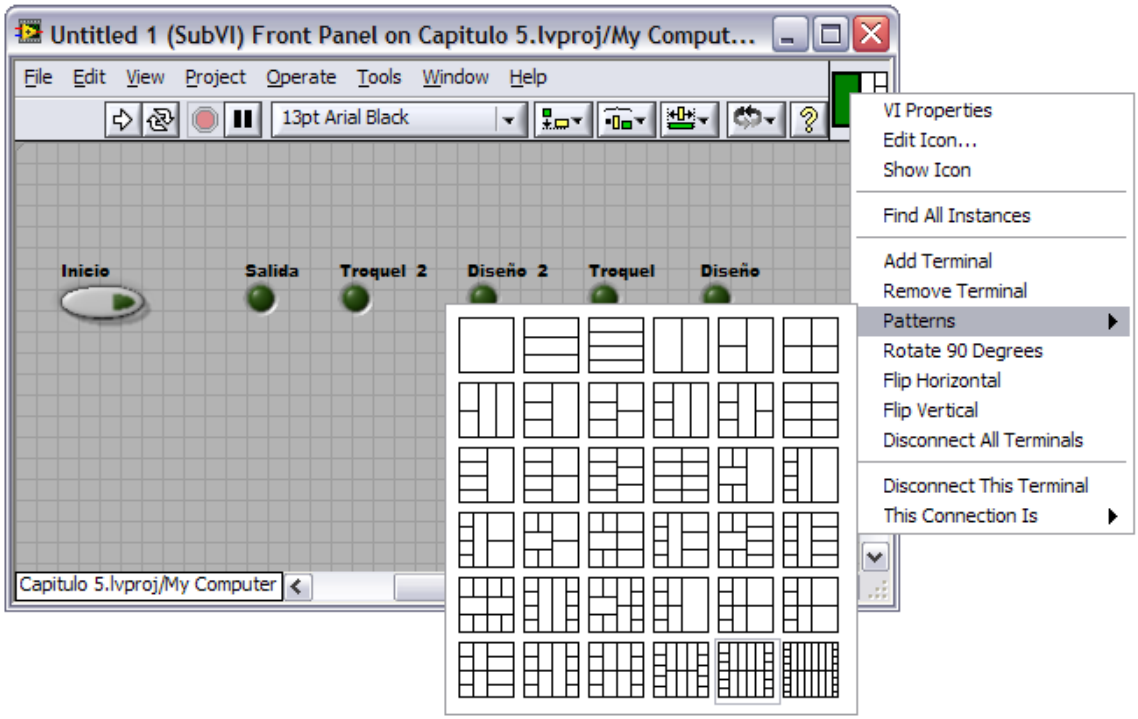
Figura 124. **Cambio del orden de conexión de las terminales en el conector**



Además, también se puede cambiar el patrón de divisiones del conector por si se desean agregar más elementos al *subVI*, como por ejemplo una función más, o más indicadores y controles.

Para esto se repite el procedimiento anterior pero en lugar de seleccionar *disconnect all terminals*, seleccionará el campo *patterns*, que le mostrará una serie de patrones de terminales ya definidos para que seleccione el que considere más conveniente; algunos recomiendan escoger los patrones con el mayor número de terminales ya que nunca se sabe cuando se necesitaran más elementos dentro de un *subVI*. La figura 125 ilustra estos patrones.

Figura 125. **Patrones de terminales predefinidos**



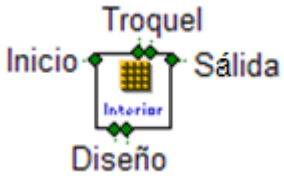
Ahora que conoce cómo crear un *subVI*, en los siguientes párrafos se podrá continuar explicando cómo fue que se usó esta herramienta para ahorrar trabajo de programación y espacio de código en la realización del programa del primer diseño.

Retomando la situación previa a explicar lo referente a los *subVIs*, el lector se encontraba con que para realizar el diseño del cuadriculado se necesitarían 48 funciones de línea recta y 16 funciones de contorno. Ahora que sabe que es el *subVI*, resulta lógico que en lugar de utilizar las 48 funciones de línea recta y las 16 de contorno, se haga un *subVI* que trace el cuadrado.

Entonces se utilizó el procedimiento recién descrito para la creación del *subVI* y el resultado fue una función, una rutina que hace lo que necesito. Primero mueve el mecanismo hasta una posición específica, luego baja el troquel, realiza el cuadrado y de nuevo sube el troquel; este *subVI* se nombró "interior" y su ícono es una cuadrícula.

El *subVI* interior posee seis terminales, una llamada inicio que es la entrada para el flanco positivo que iniciará con el proceso, tiene dos salidas para el indicador troquel, una por cada función utilizada para controlarlo, dos salidas para el indicador diseño, una para la función que mueve el mecanismo y otra para la función de contorno que realiza el cuadrado y por último la terminal de salida, que se utiliza para habilitar cualquier otra función o *subVI* conectada en serie. La figura 126 ilustra este *subVI*.

Figura 126. ***SubVI* interior**



Una vez probado y aprobado, el siguiente paso fue crear copias del *subVI* interior, 15 para ser específico y la forma de crear las copias del *subVI* es ir a la carpeta donde se guardó el original y ahí crear las copias necesarias, guardarlas con nombres diferentes al original y finalmente agregarlas una por una al *Project explorer* de *LabVIEW*.

Ahora que las copias han sido creadas la única modificación que se debe hacer tiene lugar en la primera función de línea recta, ya que ésta es la encargada de mover el mecanismo al siguiente punto donde se deberá de realizar el proceso de nuevo, se debe modificar la constante ligada a la entrada *position*, de tal forma que el nuevo valor ingresado coincida con los parámetros establecidas para el diseño.

El primer *subVI* lleva el mecanismo al punto (0; 10.11), por lo tanto para cumplir con la condición de espaciar los cuadrados del diseño 20 milímetros, el segundo *subVI* debería mover el mecanismo a el punto (-125; 10.11) que corresponden a los 105 milímetros del cuadrado más los 20 milímetros de separación, el tercer *subVI* tendrá el punto (-250; 10.11) y el cuarto *subVI* el punto (-375; 10.11) de tal manera que al colocar los cuatro *subVIs* en serie se realice el trazo de cuatro cuadrados de derecha a izquierda.

Posteriormente se agregarán otros cuatro *subVIs*, también en serie con los anteriores, pero con sus respectivas constantes de posición modificadas de tal forma que ahora se tracen los cuadrados de izquierda a derecha, iniciando en el punto (-375; 135.11) para el primer *subVI*, el segundo llevará el mecanismo al punto (-250; 135.11), el tercero al punto (-125; 135.11) y el cuarto en el punto (0, 135.11).

Hasta este punto se cuenta con dos filas de cuadrados de 105 milímetros cuadrados, con separación de 20 milímetros entre cada uno de ellos. Para finalizar el diseño se colocan los ocho *subVIs* restantes en serie, junto con los anteriores y se deberá modificar la constante de posición de la primera función de línea recta en cada *subVI*, así es cómo se completará el diseño del cuadriculado.

Ahora un detalle importante, que no es tan evidente, es que la tabla que requiere la función de contorno nunca fue modificada y el por qué residen en lo siguiente. Como recordara se mencionó, cuando se explicaron las características de esta función, que su punto de referencia o cero siempre era relativo a la última posición tabulada en la tabla, entonces, cuando el *subVI* termina de trazar el cuadrado y habilita el siguiente *subVI* por medio de la terminal salida, la función de contorno toma la nueva posición donde dibujará el cuadrado como el nuevo cero y es por eso que la tabla nunca se modifica para este diseño, lo cual es una ventaja importante ya que de lo contrario habría que crear 16 tablas con coordenadas específicas para cada cuadrado.

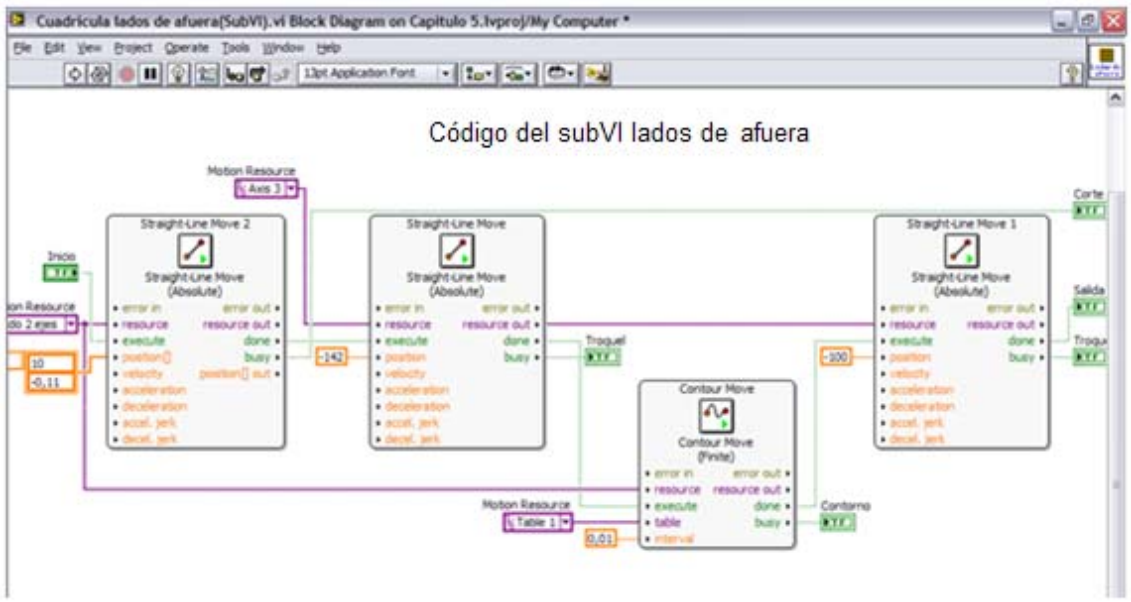
Una vez colocados los 16 *subVIs* en serie para realizar los 16 cuadrados con las distancias especificadas en el diseño 1, hay otro problema, desde el momento que se cuenta con 16 *subVIs* y cada uno de ellos tiene terminales para los indicadores de troquel y diseño respectivamente, presentes en la pestaña de controles principales, cómo hacer para que cada grupo de 32 terminales activen su respectivo indicador, ya sea el de diseño o el de troquel mientras estos bloques están en operación.

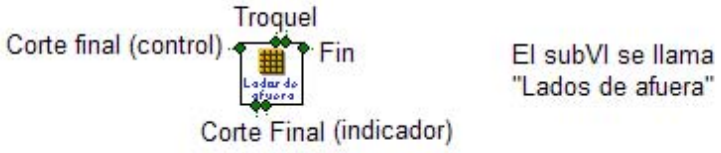
La solución es sencilla y utiliza la herramienta *build array* y la herramienta *or array*, la forma de combinarlas es la siguiente. *Build array,* permite tener varias entradas sencillas y convertirlas en un arreglo de datos tan grande como el usuario desee, con esta herramienta logramos convertir 32 terminales de salida que deben ser ligadas al indicador de diseño, en una sola terminal de salida con datos del tipo *array*. Ahora la función *or array*, se encarga de convertir la salida de datos del tipo *array* en una salida *booleana* que obedecerá la tabla de verdad de la función *booleana or* que entrega verdadero cuando una de sus entradas es verdadera.

En resumen lo que se hizo con ambas herramientas fue crear una función *booleana or* de 32 entradas cuya salida se ligó al indicador diseño y de la misma manera se utilizó un *build array* y un *or array* para formar una segunda función *booleana or* de 32 entradas para ligar su salida al indicador troquel, asegurando de esta forma que cada vez que una de las terminales de salida, ya sea del indicador diseño o del indicador troquel, esté activa, la función *or* de 32 entradas convertirá ese valor en un flanco positivo en su salida lo cual a su vez activará el indicador respectivo en la pestaña de controles principales.

Dando una mirada hacia atrás, se podrá observar que ya casi se completa el diseño, salvo por el corte final o contorno, que para crearlo de nuevo se recurrió a la creación de un *subVI*. Para el *subVI* del corte final se utilizaron las mismas funciones que para el cuadrado, tres de línea recta y una de contorno, pero esta vez la tabla ligada a la función de contorno es distinta ya que debe trazar un cuadrado de 500 milímetros cuadrados y las coordenadas para la entrada de posición de la primera función de línea recta son las del punto inicial (10; -0.11), la figura 127 muestra el código gráfico y el *subVI*.

Figura 127. **Código gráfico e ícono del *subVI* para el corte final**



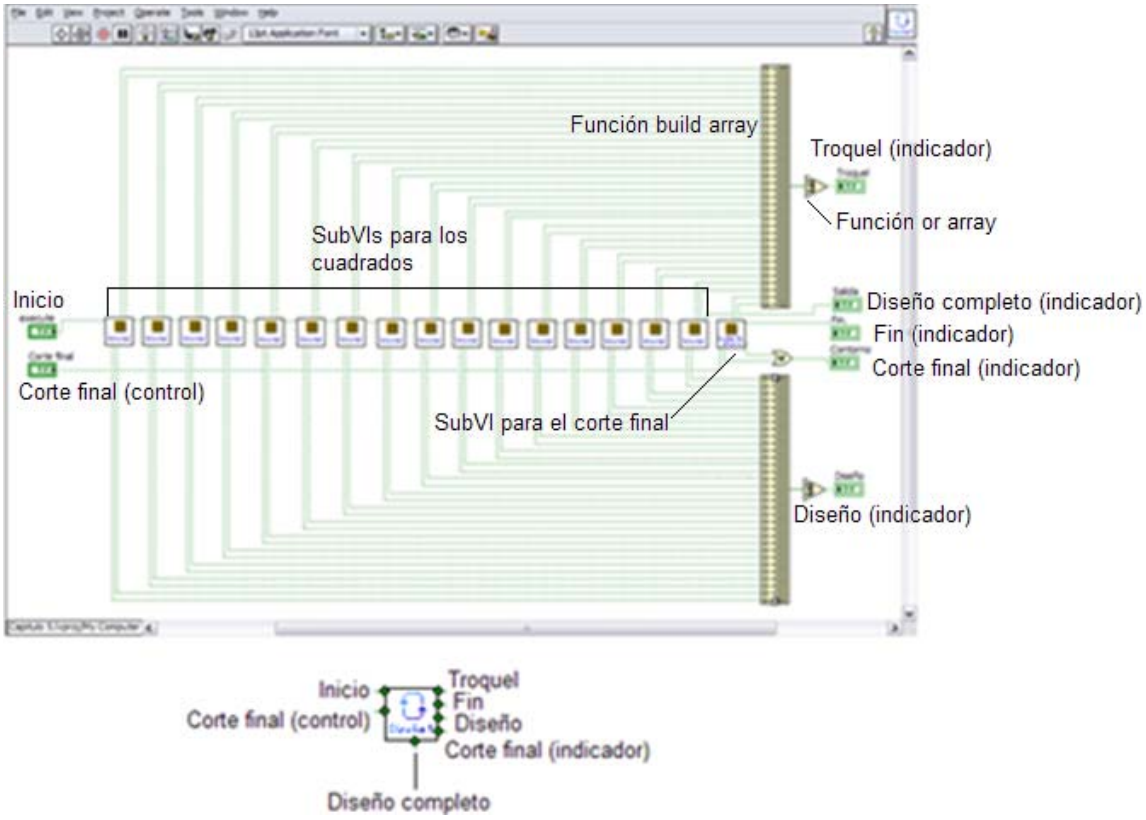


Además, el *subVI* “lados de afuera” no está conectado en serie con los *subVIs* previos, éste posee su propio control, ligado al control corte final en la pestaña de controles principales, ya que en este punto es donde se hace una pausa en el diseño de la pieza, para colocar un soporte y evitar que ésta caiga al suelo una vez que haya finalizado el corte final. Además, en lugar de tener salidas para el indicador diseño, las tiene para el indicador de corte final también presente en la pestaña de controles principales.

Ya se ha explicado cómo se programó cada parte del diseño 1 por separado, ahora es tiempo de mostrar cómo fue programado el diseño 2, para finalmente mostrar cómo es que todo está unido para que funcione de manera adecuada y se logre plasmar el diseño, tal y como aparece, del control al mecanismo. Pero antes de continuar, es importante mostrar como lucen todos los *subVIs* de los que se habló conectados entre sí, para dar lugar a su vez a un nuevo *subVI* llamado “Diseño 1”.

El *subVI* diseño 1 es un *subVI* formado por todos los *subVIs* necesarios para realizar el diseño del cuadriculado y el corte del contorno de 500 milímetros cuadrados, también cuenta con todas las terminales para los controles e indicadores presentes en la pestaña de controles principales, los cuales son inicio, corte final (control), troquel, diseño, diseño completo, corte final (indicador) y por último la terminal para el indicador de fin, que señala el final de todo el proceso. En resumen el *subVi* diseño 1 es quien posee dentro de sí todo el código gráfico que se ha explicado algunos párrafos atrás y la figura 128 ilustra como luce por dentro y por fuera (ícono).

Figura 128. **Interior y exterior del *subVI* "Diseño 1"**



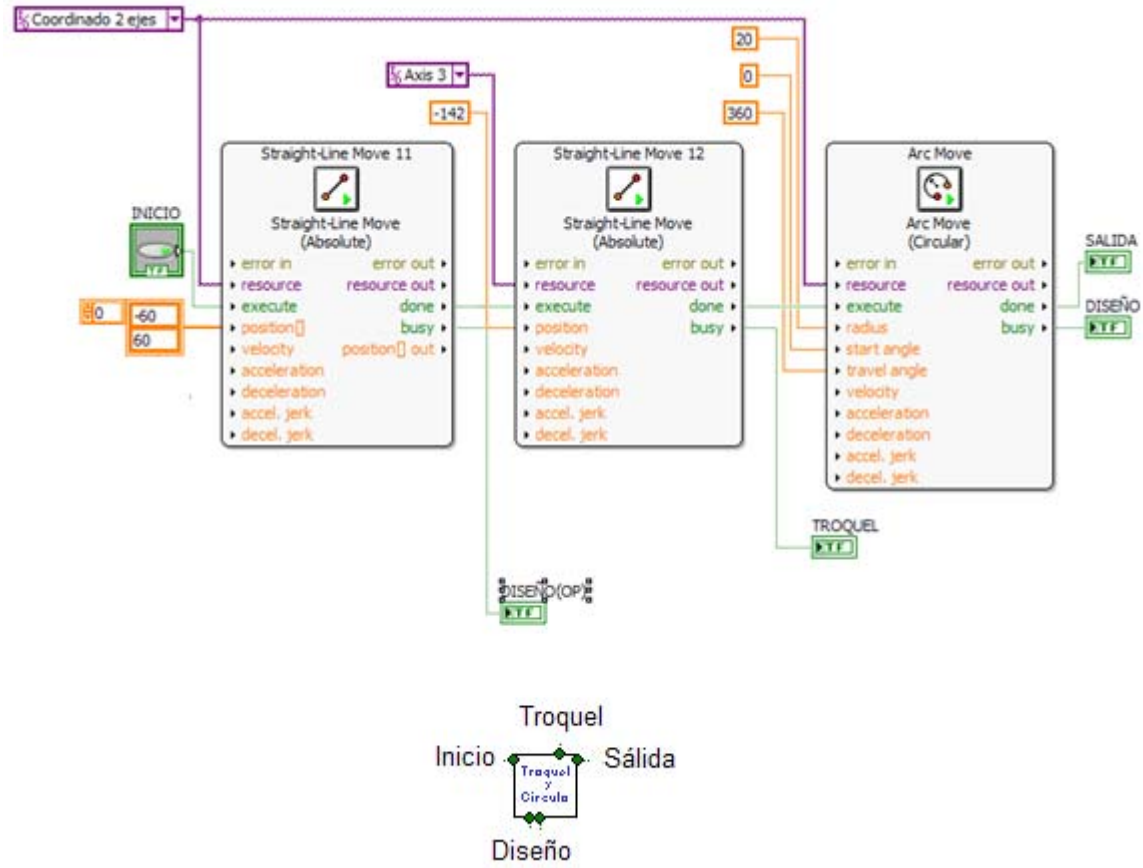
Con esto se nota que el *subVI* aparte de reducir el número de componentes y el espacio utilizado, ayuda a detectar de manera fácil los errores ya que encierran una parte del código en un espacio reducido, identificado y de fácil acceso. Cosa que sería muy difícil si en lugar de los *subVIs* se tuviera en pantalla casi 50 funciones cableadas.

5.2.2.4.2. Programación del diseño 2

La programación del diseño 2 utilizó la misma lógica que la programación del diseño 1. Se utilizó un *subVI* para trazar los círculos, un *subVI* para trazar los rectángulos y un *subVI* para el corte del contorno de la pieza. La diferencia entre la programación de este diseño con el diseño 1, es que para la realización de los cuadrados en lugar de utilizar la función de contornos, se utilizaron cuatro funciones de línea recta y de igual forma se programó el corte del contorno de la pieza. A continuación se muestran las partes que componen cada *subVI* que forma parte de este diseño.

El primer *subVI* que se creó fue para trazar un círculo y está compuesto por una función de línea recta, con un *coordinate space* de dos ejes ligado a la entrada *resource* y que se encarga de mover el mecanismo al punto donde se iniciará con el trazo del círculo, para este caso el punto es (-60, 60). Le sigue una segunda función de línea recta, cuyo trabajo es mover el troquel desde cero hasta el punto -142 y finalmente la función de arco para dibujar el círculo, la cual tiene programado realizar un círculo con radio de 20 milímetros, con ángulo de inicio cero y ángulo de viaje 360 grados. La figura 129 muestra como luce el código y el ícono para este *subVI* llamado “círculo y troquel”.

Figura 129. **Código e ícono para *subVI* encargado de realizar un círculo**



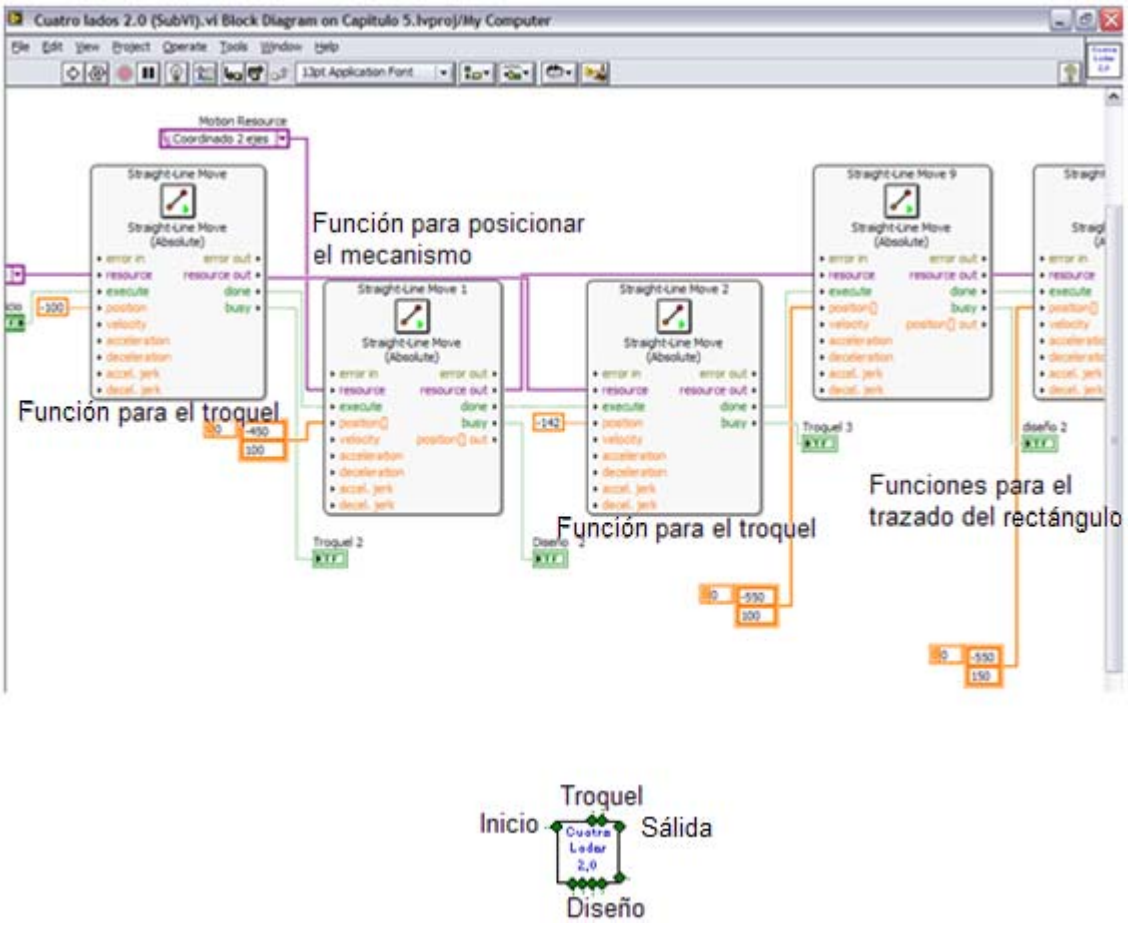
Como se puede observar también se utiliza la salida *busy* para mantener activos los indicadores de diseño y troquel, presentes en la pestaña de controles principales, además de la salida *done* para habilitar la siguiente función colocada en serie. Como observará, la lógica de programación es exactamente la misma, lo que difiere en este diseño 2 son los elementos utilizados para trazar el diseño.

Este *subVI* se copió diez veces más, ya que el diseño requiere que realicen 10 círculos de 20 milímetros de radio y un círculo mas, para la chapa del panel, de 35 milímetros de radio. Es importante mencionar que a cada copia del *subVI* le fue modificada la constante de la primera función de línea recta, para que mueva el mecanismo al punto exacto donde se debe trazar el siguiente circulo. Como dato relevante, se debe mencionar que los círculos tienen separación entre sí de 30 milímetros.

El segundo *subVI* utilizado para el diseño dos, es el que debe trazar los rectángulos de 100 x 50 milímetros cuadrado y está compuesto por siete funciones de línea recta, la primera para subir el troquel, la segunda para mover el mecanismo al punto del trazo, la tercera para bajar el troquel y las cuatro restantes para trazar el rectángulo. De nuevo se utilizó la constante ligada a la entrada *position*, para controlar la posición de los componentes, un *coordinate space* como recurso para controlar el movimiento del mecanismo y el trazo del cuadrado y el eje Z como recurso para controlar el troquel.

También se utilizó la misma técnica de la salida *busy* para controlar los indicadores de diseño y troquel del panel de controles principales. El *subVI*, llamado “cuatro lados” fue copiado dos veces más, ya que el diseño requiere de tres rectángulos con una separación de 100 milímetros entre sí. Para programar el rectángulo bastó con colocar las coordenadas de cada vértice del rectángulo en la constante de posición de cada función de línea recta y en cada copia se modificaron estas coordenadas para mantener la separación de 100 milímetros entre cada rectángulo. La figura 130 ilustra parte del código gráfico y el Ícono del *subVI*.

Figura 130. **Código e ícono del *subVI* para trazar un rectángulo**



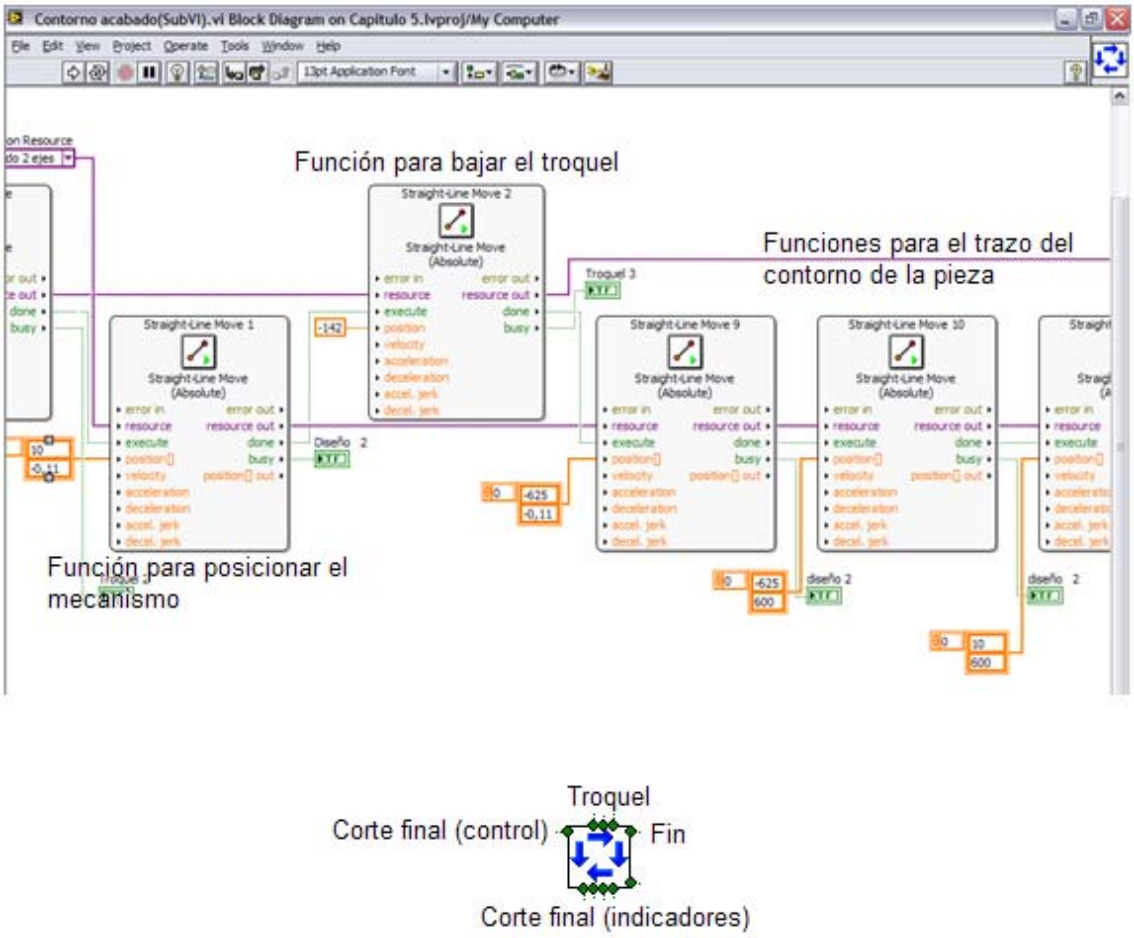
Cómo se puede observar en el ícono, hay cinco terminales que hacen referencia al indicador de diseño, esto debido a que se usaron cuatro funciones de línea recta para el rectángulo, por lo que hay cuatro salidas de *busy*, en lugar de una función de contorno que haría el mismo rectángulo con una sola salida *busy*; la quinta salida *busy* que hace referencia al indicador diseño es de la función encargada de mover el mecanismo hasta el nuevo punto de trazo.

Finalmente el último *subVI* que compone el diseño 2, es el encargado de realizar el corte del contorno de la pieza. Este *subVI*, también posee su propio control para inicializarlo, al igual que el del diseño 1, ligado al botón corte final del panel de controles principales. Hay que recordar que en este punto del proceso es cuando se debe colocar algo que detenga la pieza en su lugar, para que cuando se lleve a cabo el corte final la pieza no caiga al suelo.

Este *subVI* se llama “contorno” y está compuesto por ocho funciones de línea recta, la primera encargada de subir el troquel, la segunda de mover el mecanismo al punto donde se iniciará con el corte del contorno, la tercera para bajar el troquel, cuatro más para realizar el contorno, que por cierto tiene 635 x 600 milímetros cuadrados de área; y una última para subir el troquel una vez que ha finalizado el corte del contorno.

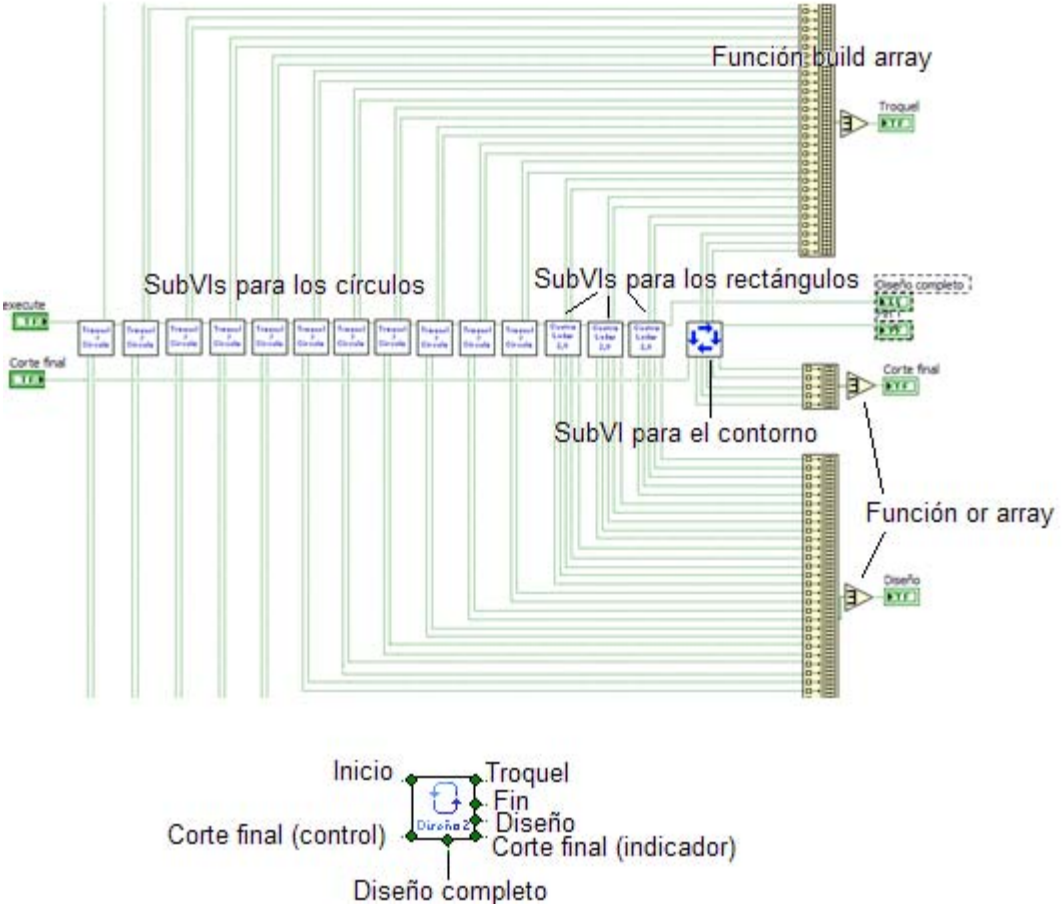
Cómo supondrá de nuevo, se aclarará que se modificaron las constantes de la entrada posición de cada función de línea recta para que realicen los trazos adecuados y que este *subVI* también aporta terminales de salida ligadas a los indicadores corte final y troquel presentes en la pestaña de controles principales en el panel de control del mecanismo. La figura 131 ilustrará parte del código gráfico y el ícono del *subVI*.

Figura 131. **Código gráfico e ícono del *subVI* para el corte del contorno del diseño 2**



Ahora que se explicó, cómo se crearon los *subVIs* que dan forma a la pieza correspondiente al diseño 2, es tiempo de mostrar como luce el código gráfico dentro del *subVI* llamado diseño 2 y presentar su ícono antes de pasar al siguiente punto que es ver como luce todo el código de la aplicación de control, la figura 132 ilustra lo descrito en las líneas previas.

Figura 132. **Código e ícono del *subVI* "Diseño 2"**



5.2.2.5. Código gráfico de programación para aplicación final de control

En los siguientes párrafos se mostrará como luce el código de programación para la aplicación completa del panel de control para la troqueladora. Se explicará que hace cada parte del código y porque fue colocado en ese lugar específico, y como siempre se utilizarán las imágenes.

La primera parte del código que será introducida es la parte donde se encuentra el *subVI* que he creado para los indicadores. Este *subVI* se llama "salidas" y contiene los indicadores para los sensores de límite de posición de la máquina, indicadores identificados como –X, X, -Y, Y y –Z. También incluye los indicadores numéricos para la velocidad y aceleración de cada uno de los ejes de movimiento de la máquina.

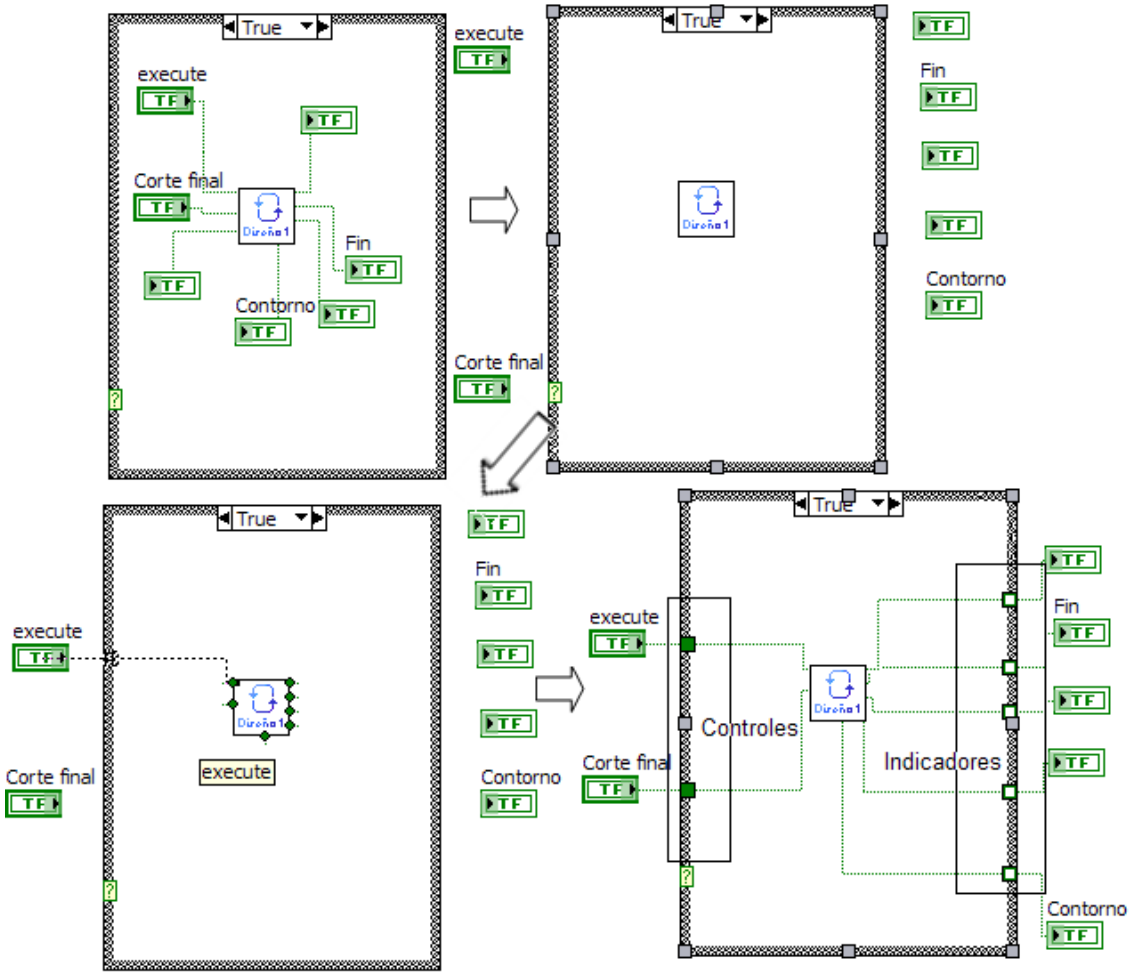
Además, podrá observar el *timed loop*, que sirve para sincronizar la comunicación entre el código de *LabVIEW* y la simulación por medio del *scan engine*, el *case structure,* utilizado para que el control permita al usuario seleccionar un diseño de los dos disponibles. También notará el control *text ring* ligado a la entrada del *case structure* y el indicador *pict ring* ligado a este mismo cable y cuya función en conjunto es seleccionar el diseño a realizar y que la imagen cambie cuando de acuerdo al diseño 1 o diseño 2.

Y finalmente el último detalle importante que verá, son unos cuadritos de colores que se encuentran sobre el borde del *case structure*, dichos cuadritos se conocen como nodos de datos y su función es ingresar o extraer datos del case *structure*. Por ejemplo un nodo ligado a un control ingresará el dato emitido por el control y lo llevará hasta la función o VI que lo requiera y que está dentro del *case stucture*, de manera similar un nodo ligado a un indicador, tomará el dato emitido por la función o VI y lo llevará fuera del *case structure* hasta el indicador al que esté ligado.

La utilidad de los nodos de datos reside en que al colocar controles e indicadores fuera del *case structure,* éstos pueden comunicarse con el código dentro del *case structure*, sin importar si el código dentro del *case* cambia. Dicho de otro modo, el utilizar nodos de datos para ligar controles e indicadores externos al *case structure,* hace que éstos funcionen con cualquier código dentro del *case structure*, evitando el trabajo de colocar controles e indicadores para cada código presente en los diferentes casos contenidos en el *case structure.*

Para colocar un nodo de datos se hace lo siguiente, primero dentro del *case structure* se crean los controles e indicadores necesarios, luego se seleccionan, uno por uno, y se llevan fuera del *case structure*. Habiendo hecho esto, se crea un cable desde el control o indicador hacia la terminal de conexión a la que quedará ligado, dentro del *case structure*, pasando por el borde del mismo, en ese momento observará cómo se forma un cuadrado con una línea punteada sobre dicho borde, luego sigue el camino hasta llegar a la terminal destino y listo, se habrá creado el nodo de datos ligado al control o indicador deseado, la figura 133 ilustra.

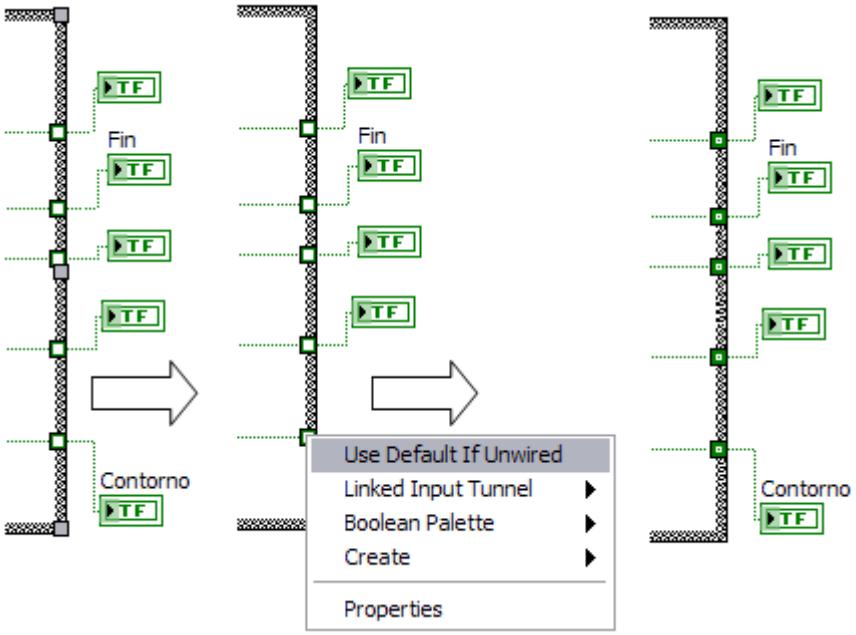
Figura 133. **Creación de un nodo de datos**



Ahora, si es observador notará que los nodos para controles están completamente coloreados, mientras que los nodos para los indicadores tienen un espacio en el centro de color blanco, esto es un error de programación que no permitirá la ejecución del programa. Para arreglarlo basta con posicionarse sobre un nodo de datos ligado a un indicador, hacer clic derecho sobre él y entre las opciones que despliega, seleccionar la opción *use default if unwired*.

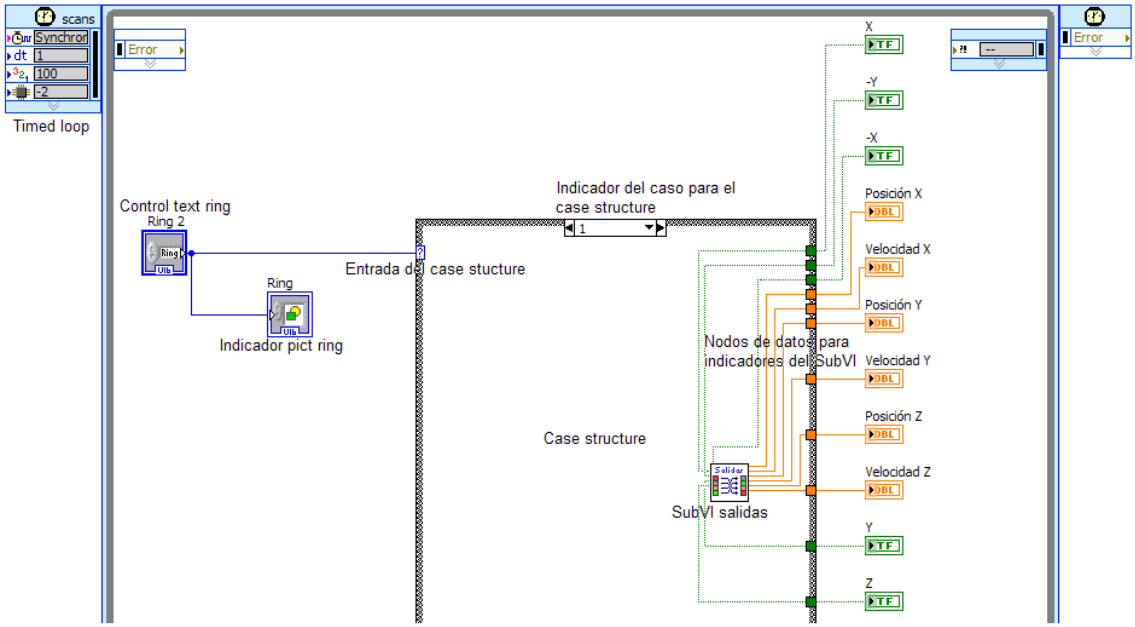
Luego de haber hecho esto notará como el nodo cambia y aún posee un punto blanco en el centro pero esta vez es más pequeño, el mismo proceso se deberá repetir para cada nodo ligado a un indicador. La figura 134 muestra el procedimiento.

Figura 134. **Habilitación de un nodo de datos ligado a un indicador**



Luego de haber hecho un paréntesis, para explicar lo referente a los nodos de datos, es momento de presentar la primera figura del código de la aplicación final de control, que fue descrita al inicio de este apartado. La figura 135 ilustra el código.

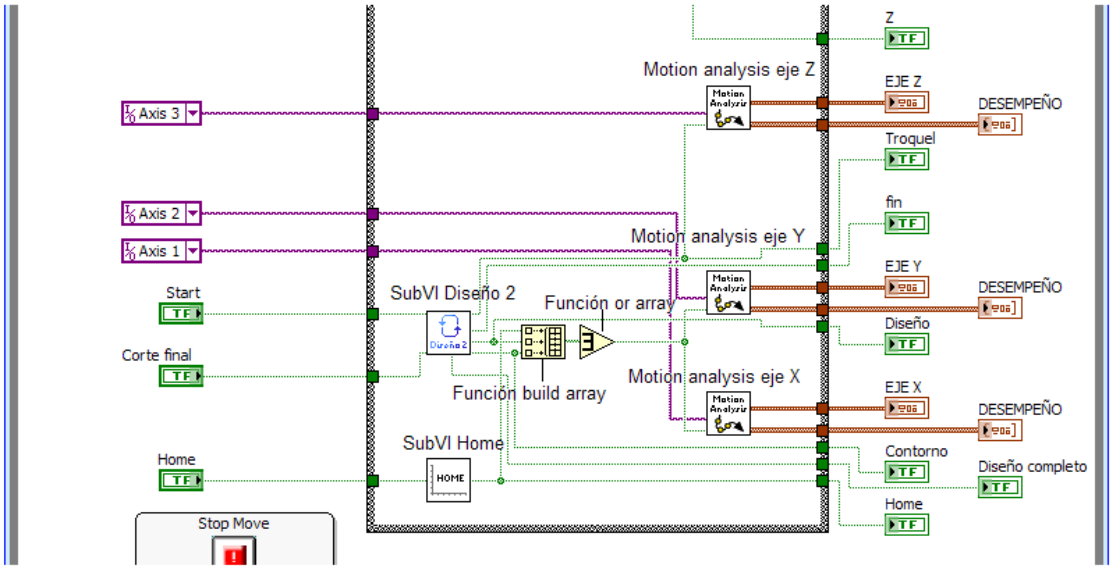
Figura 135. **Primera figura del código de la aplicación final de control**



La segunda figura del código de la aplicación final de control presentará el *subVI* diseño, el *subV*I para el ajuste de cero llamado "*home*" y tres *subVIs* de la herramienta *motion analysis* para extraer datos del desempeño de cada eje de movimiento.

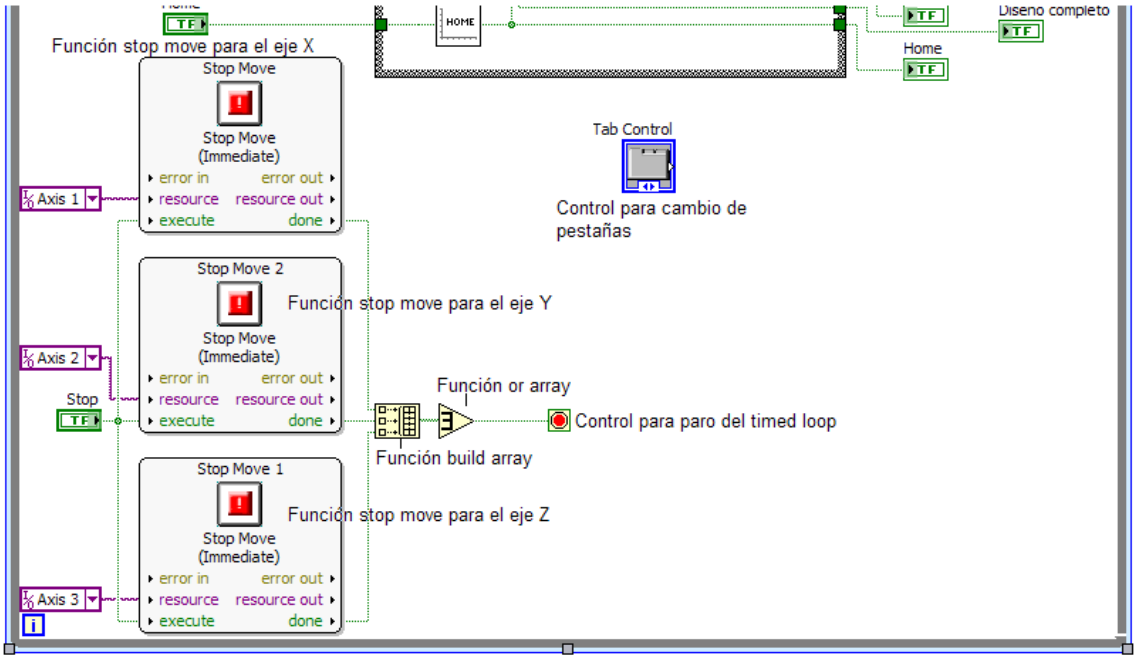
El *subVI* diseño, posee dentro de sí todo el código para la realización del diseño seleccionado, el *subVI* diseño 1 realiza el primer diseño y diseño 2 realiza el diseño 2. El *subVI home*, es el encargado de llevar el mecanismo a la posición de inicio, el punto (10; -0.11) y finalmente hay tres *subVIs* que se encargan de extraer datos como, torque, velocidad, tiempo de ejecución y tiempo en espera, además de mostrar gráficas para posición, velocidad y torque del eje al que están ligados. La razón por la que aparecen tres *motion analysis* es porque cada uno de ellos toma datos del eje X, Y y Z respectivamente, la figura 136 muestra el código.

Figura 136. **Segunda figura del código de la aplicación final de control**



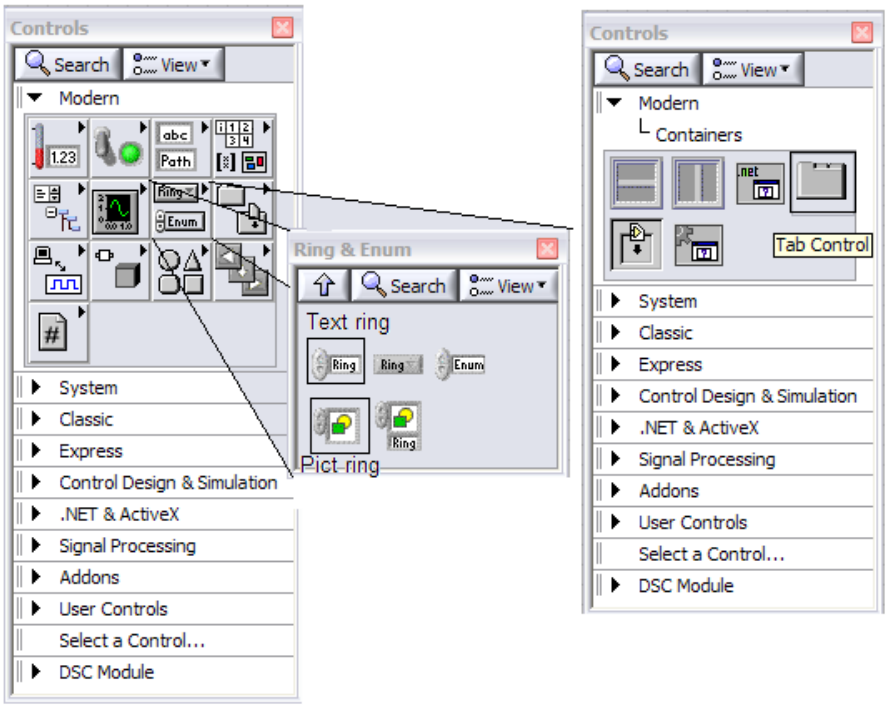
Y por último la tercera imagen del código para la aplicación de control mostrará las funciones *stop move*, que fueron ligadas al control de paro del panel de controles principales. Se utilizaron tres funciones *stop move*, una para cada eje de movimiento del mecanismo, ligadas entre sí de tal manera que cuando se presione el botón de paro, el flanco positivo que genera dicho control, habilite a las tres funciones de manera simultánea y de esta forma se detenga la ejecución tanto de la aplicación de *LabVIEW* como cualquier movimiento que se esté llevando a cabo en *Solidworks*. La figura 137 muestra el código.

Figura 137. **Tercera figura del código de la aplicación final de control**



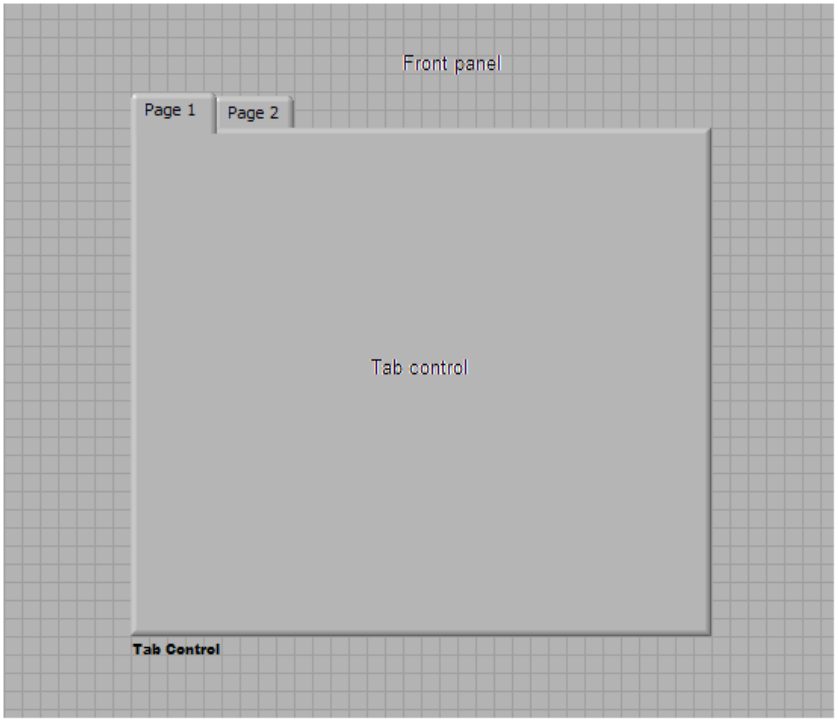
Ya se ha hablado acerca del código en el *block diagram*, ahora es turno de comentar la programación de la aplicación en el *front panel*. Existen algunas herramientas como los *containers* que sólo pueden ser colocadas estando en el *front panel.* Uno de los *containers* utilizados para esta aplicación, es el control para cambiar de pestaña llamado *tab control,* otros elementos utilizados para esta aplicación y que sólo pueden ser colocados en el *front panel* son el *text rign* y el *pict ring*. Dichas herramientas pueden ser localizadas en la paleta de funciones, en la pestaña *modern* y en el apartado llamado *enum and ring* y en el apartado *containers*, la figura 138 muestra donde encontrarlos.

Figura 138. **Apartado *enum and ring* de la paleta de funciones en el *front panel***



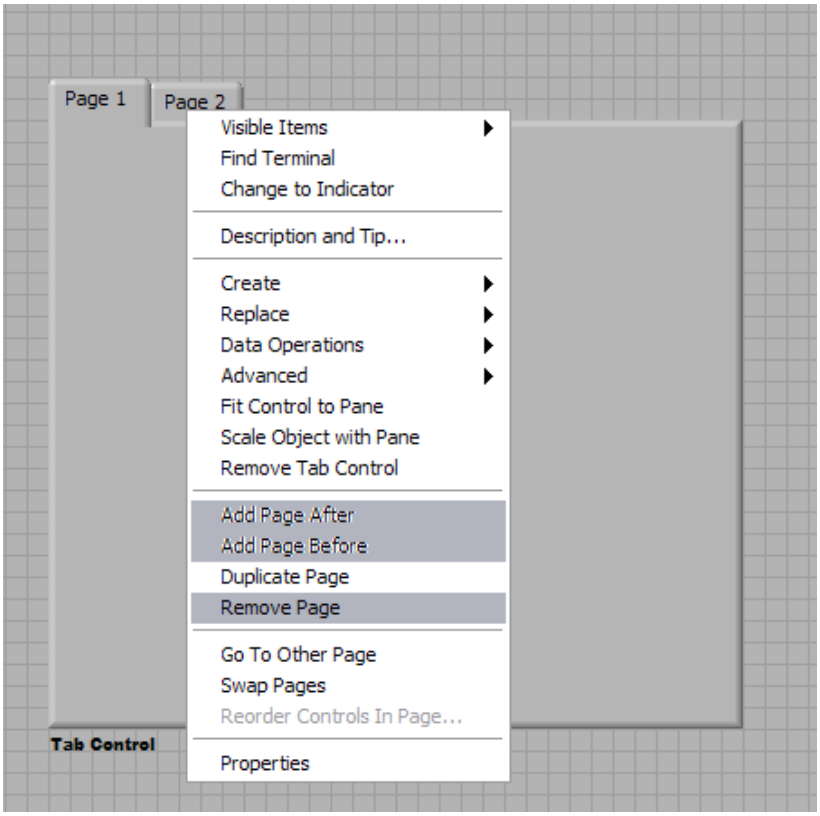
Entonces, el *tab control* es un *container* o contenedor y su función es brindar un espacio para colocar controles y/o indicadores y definir la manera de presentarlos, generalmente es por medio de diferentes pestañas. Cada pestaña brinda un espacio de tamaño definido para colocar los controles e indicadores deseados, se pueden agregar o eliminar pestañas al gusto del usuario, además puede poner el nombre que desee a cada pestaña, teniendo en cuenta que entre más pestañas agregue menor será el tamaño para colocar el nombre, lo cual puede ser corregido aumentando el tamaño definido del *tab control*, la figura 139 muestra como luce un *tab control.*

Figura 139. ***Tab control* en el *front panel***



Para agregar pestañas basta con hacer clic derecho sobre alguna pestaña y seleccionar el campo *add page before* para agregar una pestaña antes de la pestaña que se encuentre activa, o bien seleccionar *add page after* para colocar una pestaña después de la pestaña activa. Para eliminar la pestaña, se sigue un procedimiento similar, hará clic derecho sobre una pestaña y selecciona el campo *remove page*, este comando automáticamente eliminará la pestaña que se encuentre activa y dejará sólo las pestañas inactivas en el *container,* la figura 140 muestra el procedimiento.

Figura 140. **Menú para agregar y/o eliminar pestañas**



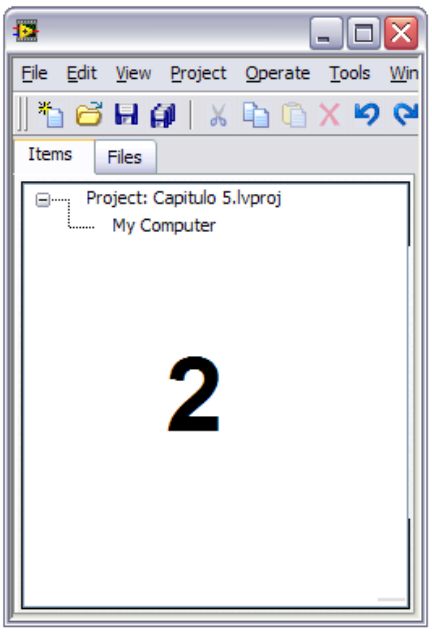
Para colocar algún control o indicador dentro de una pestaña basta con arrastrarlo y colocarlo en la posición deseada y listo, automáticamente queda ligado a la pestaña en la cual se colocó. Con respecto al control *text ring* y al indicador *pict ring* no mencionaré más que pertenecen al grupo llamado *enum and ring* y que sólo pueden ser colocados estando en el *front panel*.

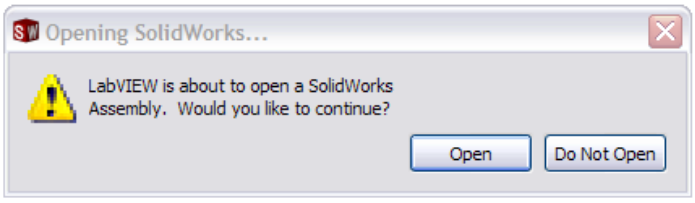
5.3. Presentación de la aplicación final de control

En esta parte se hará el mejor esfuerzo para presentar por medio de imágenes como luce la aplicación completa cuando se encuentra en funcionamiento, además se explicará paso a paso, pero de forma breve, cómo ejecutar la aplicación de manera correcta.

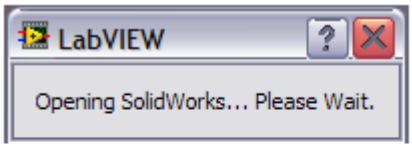
Para ejecutar la aplicación se requiere lo siguiente, lo primero es tener *LabVIEW* corriendo sobre la computadora, luego abrir el proyecto donde se realizó la programación de la aplicación, llamado capítulo 5, con esto me refiero a tener abierto el *Project explorer* de la aplicación de control. Una vez abierto el *Project explorer*, él mismo indicará que para cargar de manera correcta el proyecto se requiere abrir el archivo de *Solidworks* llamado prototipo de troqueladora a lo cual se hará clic en el botón *open*, la figura 141 ilustra.

Figura 141. **Apertura del proyecto de LabVIEW y del ensamblaje en Solidworks**



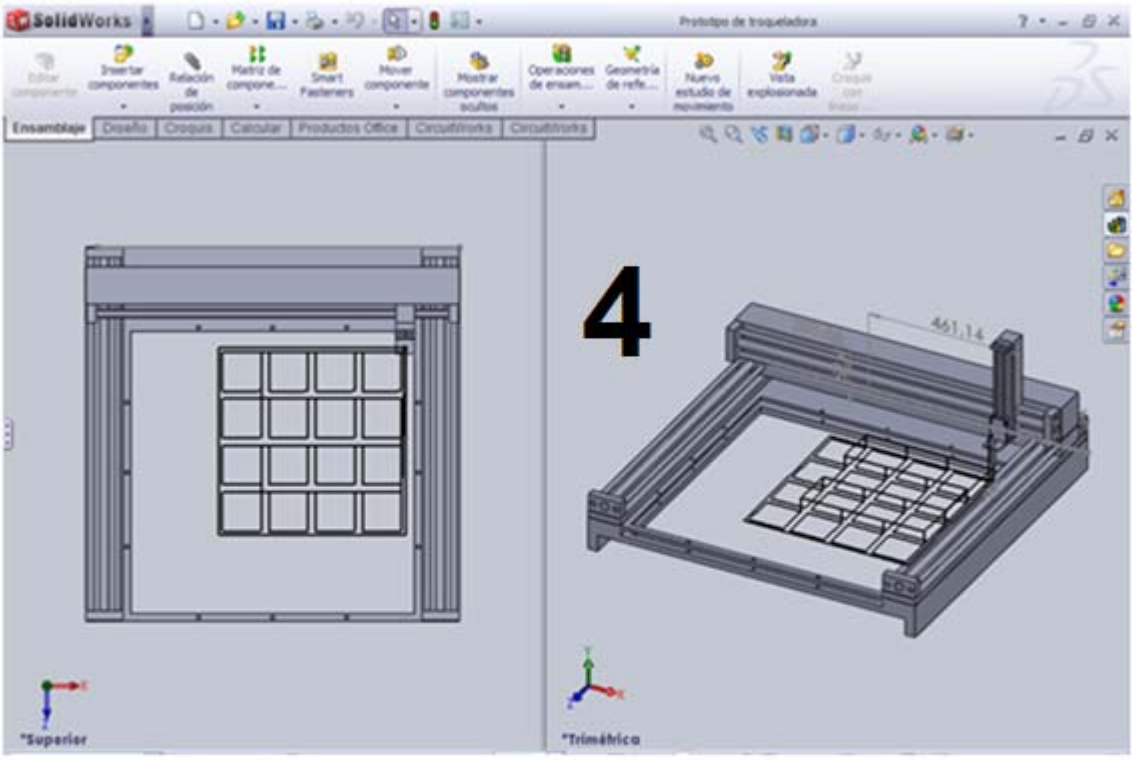


Capítulo 5 indicando que debe abrir el ensamblaje del prototipo de troqueladora creado en solidworks



3

Apertura de solidworks desde labview



Prototipo de troqueladora abierta en solidworks

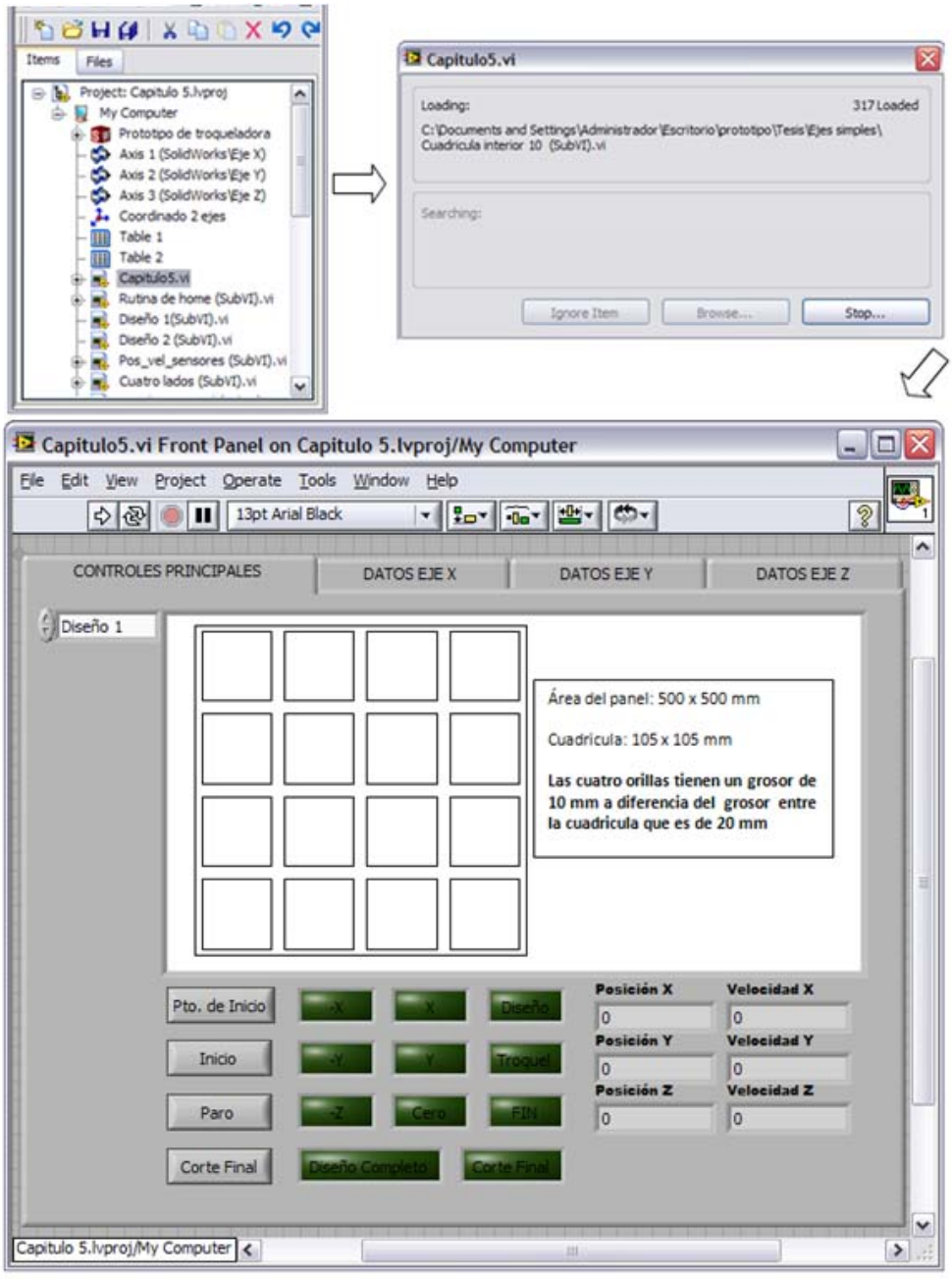
Al abrir el archivo, prototipo de troqueladora, *LabVIEW* se encargará de manera automática de ejecutar *Solidworks* y presentar en pantalla el mecanismo que ha sido creado para la simulación del proceso de troquelación.

Este proceso tomará unos cuantos minutos, pero al final del mismo se habrá completado el primer paso. Es recomendable, antes de continuar con el procedimiento revisar que el mecanismo se encuentre en la pestaña estudio de movimiento y que el tipo de movimiento que se está utilizando sea análisis de movimiento, ya que de lo contrario no será posible ejecutar la aplicación.

El segundo paso es abrir el VI principal, para este caso, el VI principal también se llama capítulo 5 y se puede encontrar buscando en la lista de componentes que pertenecen al proyecto con el mismo nombre. Para abrir el *VI* principal basta con ubicarlo dentro de la ventana del *Project explorer* y hacer doble clic sobre él.

El usuario observará como se cargan los componentes necesarios para que el *VI* principal funcione, una vez cargados todos los archivos y dependencias necesarias, tendrá en pantalla la aplicación de control para la troqueladora y habrá completado el segundo paso. La figura 142 muestra como se hace.

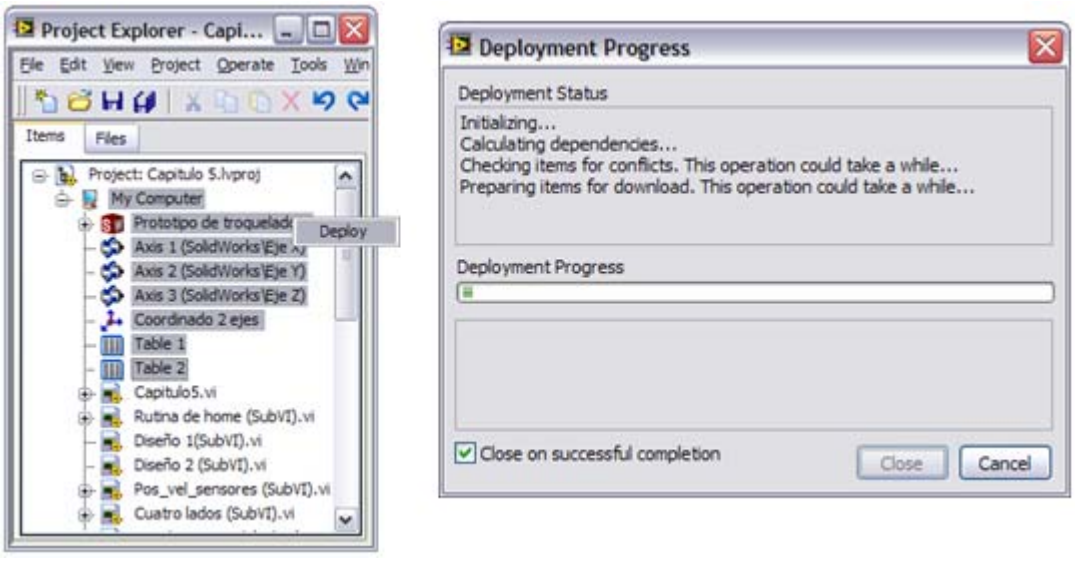
Figura 142. **Apertura del *VI* principal “Capítulo 5” y vista del *front panel* de la aplicación de control**



Con el *front panel* de la aplicación de control en pantalla, el siguiente paso es comunicar *Solidworks* y *LabVIEW* por medio del *scan engine.* En orden de lograr esto, se debe situar sobre el *Project explorer* de la aplicación, seleccionar todos los elementos citados a continuación iniciando desde el ícono de *My computer*, siguiendo con el ensamblaje de *Solidworks* y sus ejes, cada *softmotion axis*, el *softmotion coordinate space* y finalizando con ambas *softmotion table*.

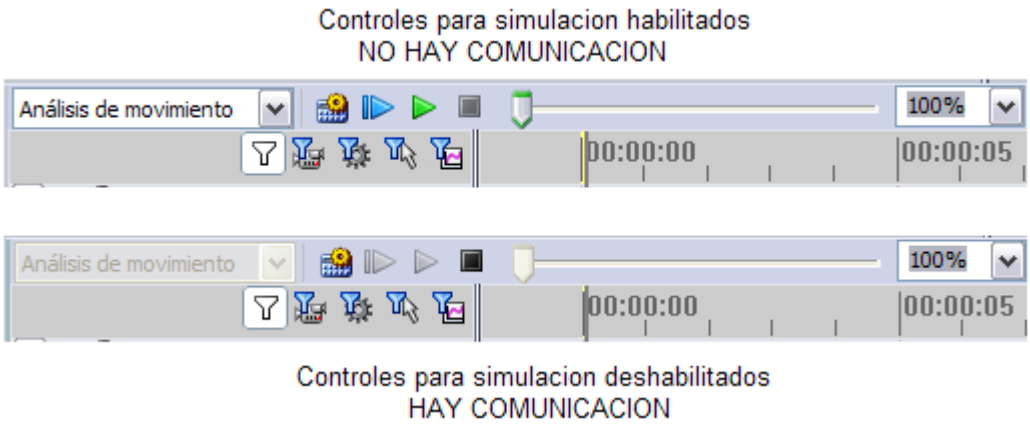
Una vez seleccionados estos elementos se hace clic derecho sobre alguno de ellos y se selecciona el campo *deploy*, lo cual se ocupará de cargar todo lo necesario para entablar la comunicación entre *Solidworks* y *LabVIEW*. La figura 143 muestra el proceso.

Figura 143. **Inicialización de comunicación entre *Solidworks* y *LabVIEW***



El paso anterior tendrá como resultado entablar la comunicación entre *Solidworks* y *LabVIEW,* y para estar seguro de que la comunicación se ha establecido con éxito basta con observar la pestaña de controles para la ejecución de simulaciones en *Solidworks*. Esta pestaña mantiene todos los botones habilitados siempre y cuando no exista comunicación entre ambos programas, pero cuando la comunicación ha sido establecida únicamente el botón de stop se encuentra habilitado. La figura 144 muestra estos botones para ambos casos.

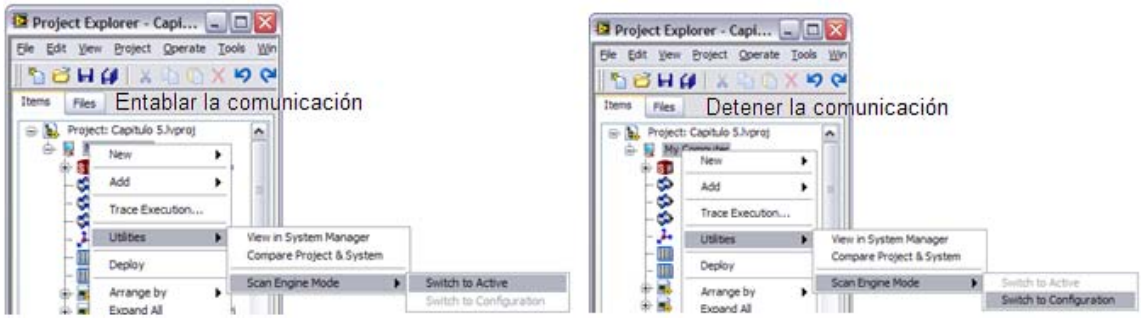
Figura 144. **Controles para simulación en *Solidworks* cuando hay comunicación y cuando no hay comunicación entre ambos programas**



Para entablar la comunicación entre *Solidworks* y *LabVIEW* por primera vez, se siguen los pasos descritos en el párrafo anterior. Para hacerlo una segunda vez, así como en ocasiones posteriores, basta con hacer clic derecho sobre el ícono *My computer* en el *Project explorer*, dirigirse al campo *utilities*, seleccionar el campo *scan engine mode* y seleccionar el campo *switch to active.*

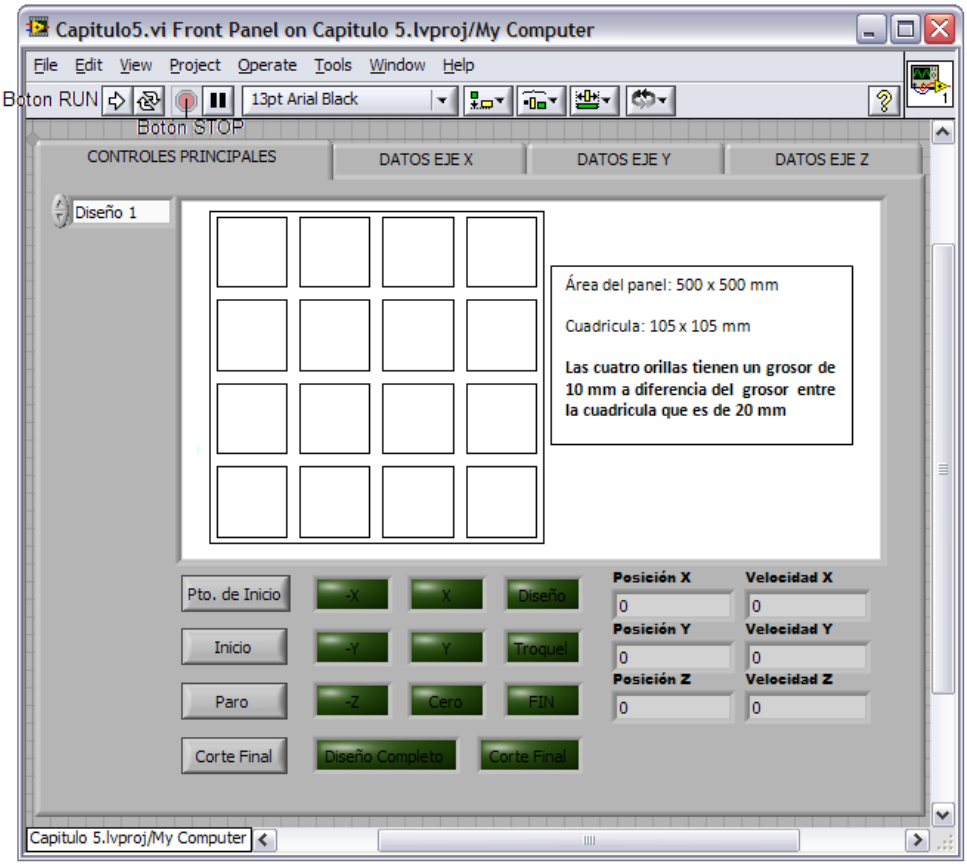
Al hacer esto podrá observar como en la ventana de *Solidworks* de nuevo los controles de simulación cambian de habilitados a inhabilitados una vez que se ha establecido la comunicación. Para detener la comunicación entre *Solidworks* y *LabVIEW*, se debe dirigir al ícono de *My computer*, hacer clic derecho sobre él, posicionarse sobre el campo *utilities*, luego *scan engine mode* y finalmente seleccionar la opción *switch to configuration.* Luego de haber ejecutado este procedimiento, podrá verificar que la comunicación ha terminado, chequeando los controles de simulación los cuales deberían de haber vuelto al estado inicial. La figura 145 ilustra.

Figura 145. **Activacion y desactivación del *scan engine***



Con la comunicación establecida entre *Solidworks* y *LabVIEW*, solo resta un paso por hacer y es ejecutar la aplicación de control. Para ejecutarla, lo primero es tener la aplicación en pantalla, luego dirigirse hacia la parte superior izquierda y situarse sobre el botón que tiene dentro una flecha, dicho botón es el botón de *run.* Con respecto a este botón lo que hay que resaltar es que cuando la aplicación tiene errores presenta una flecha rota de color gris y no una blanca y completa como el caso de la figura 146.

Figura 146. **Aplicación de control para troqueladora**



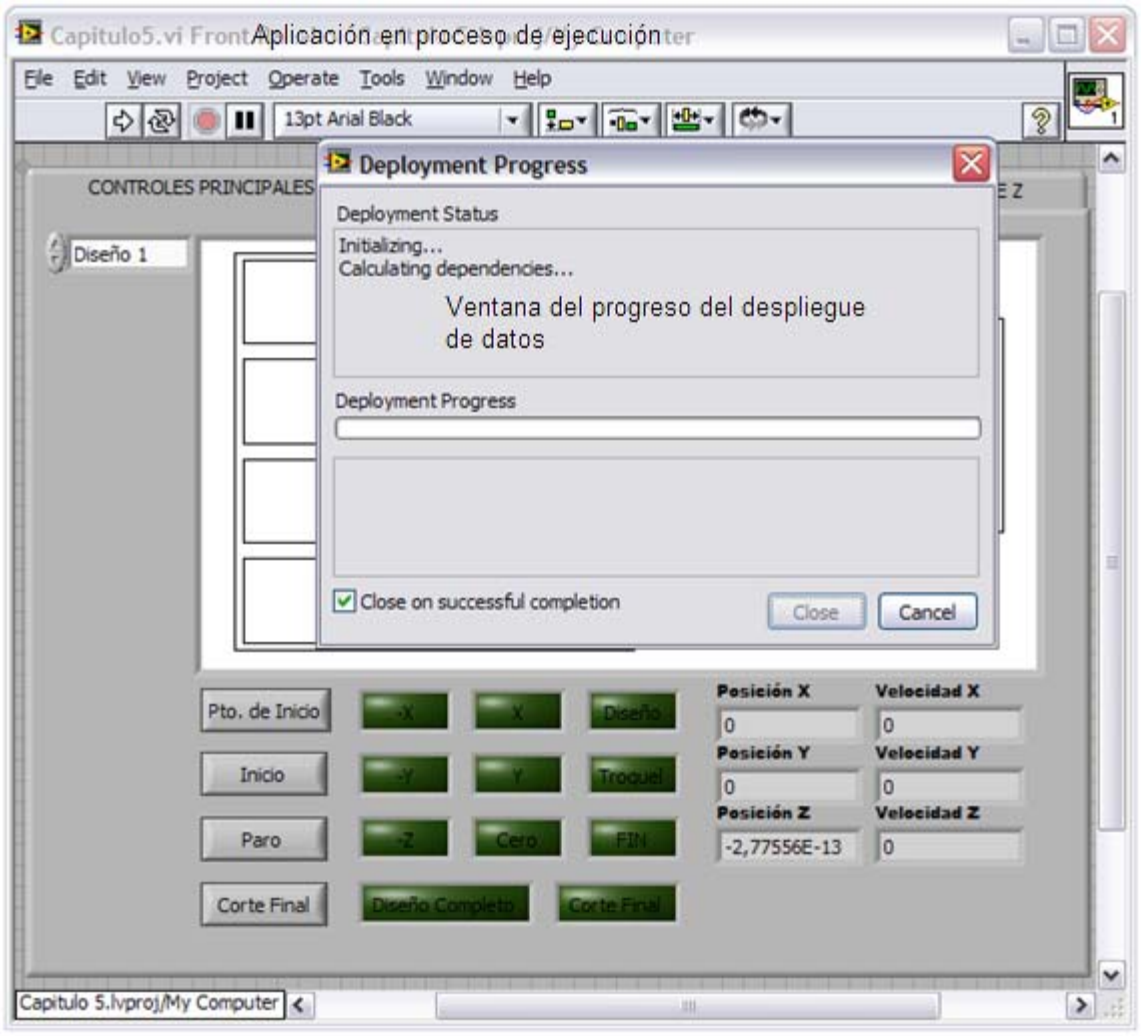
El siguiente botón, que tiene dos flechas una siguiendo a la otra, es el botón para *run continuos,* este tipo de ejecución ejecuta el mismo código una y otra vez infinitas veces, a diferencia del *run* que ejecuta el código una sola vez, luego está el botón de stop y por último el botón para pausa.

Retomando el punto, que es ejecutar la aplicación de control una vez que se ha establecido la comunicación entre *Solidworks* y *LabVIEW,* ya que se ha hecho clic en el botón *run* del *front panel* de la aplicación de control, observará como se muestra una ventana que indica que se están cargando los datos e inmediatamente después la aplicación de control estará en ejecución.

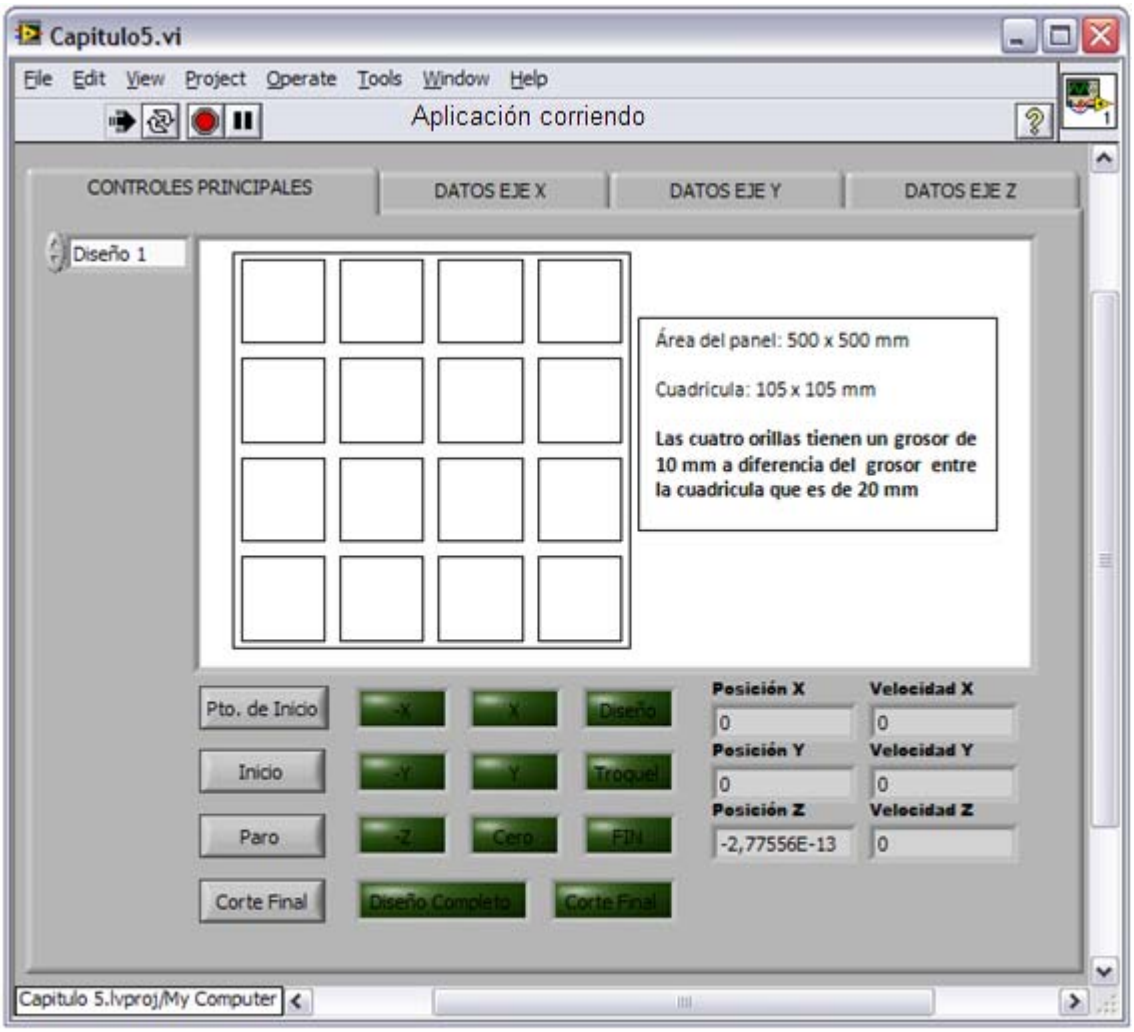
Se sabe que la aplicación está en ejecución, porque la flecha de *run* cambia de blanca a negra, el botón de stop se habilita y además la cuadrícula que sirve de guía para colocar los indicadores y controles dentro del *front panel* desaparece y queda todo absolutamente sin cuadrícula, como lo muestra la figura 147. En este punto se ha cumplido con todos los requisitos para poder iniciar con la ejecución de la aplicación de control para la troqueladora.

En los próximos párrafos se explicará cómo funciona la aplicación, mostrando imágenes de ambos programas corriendo al mismo tiempo, de tal manera que cuando se ordene al mecanismo en *Solidworks*, realizar un movimiento controlado desde *LabVIEW*, éste lo haga y tanto el panel de control como el mecanismo reflejen los resultados de ese movimiento.

Figura 147. **Aplicación de control de *LabVIEW* en ejecución**



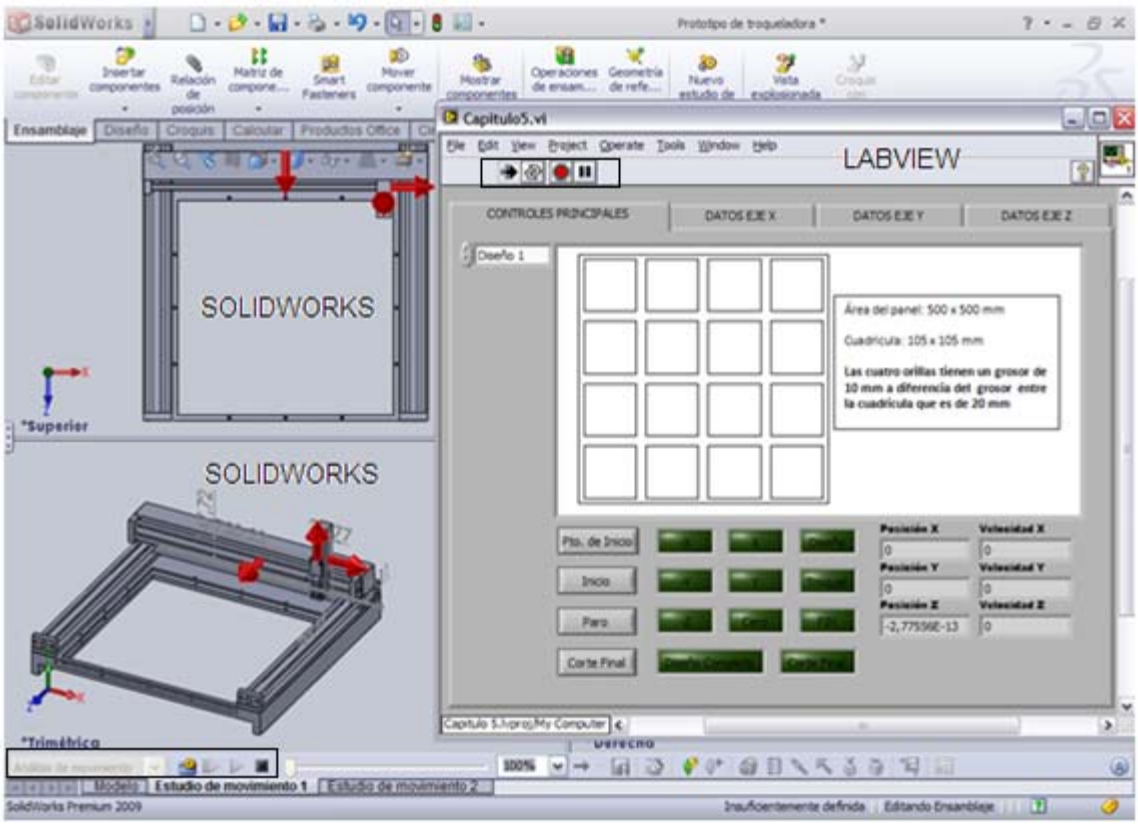
El botón de Run aún tiene la flecha de color blanco, la cuadrícula se vé en el fondo y el botón de stop está inhabilitado



La flecha de Run ha cambiado de color, ahora es negra. La cuadrícula ya no se observa en el fondo y el botón de stop ya se encuentra habilitado

En la figura 148, se ilustrarán ambos programas corriendo en simultáneo y con la comunicación establecida entre ambos. Por primera vez verá el mecanismo y el control en pantalla al mismo tiempo.

Figura 148. **Mecanismo y panel de control en pantalla al mismo tiempo**



Como se puede ver en la imagen, *Solidworks* ha establecido comunicación con *LabVIEW,* ya que en los controles para simulación de *Solidworks,* el botón stop, es el único que se encuentra habilitado. Por su parte la aplicación de control de *LabVIEW*, presenta la flecha del botón *run* en color negro, un fondo sin cuadrícula y el botón de stop habilitado. Todos los indicios son correctos, la aplicación está lista para ejecutar el diseño seleccionado en el control por el usuario.

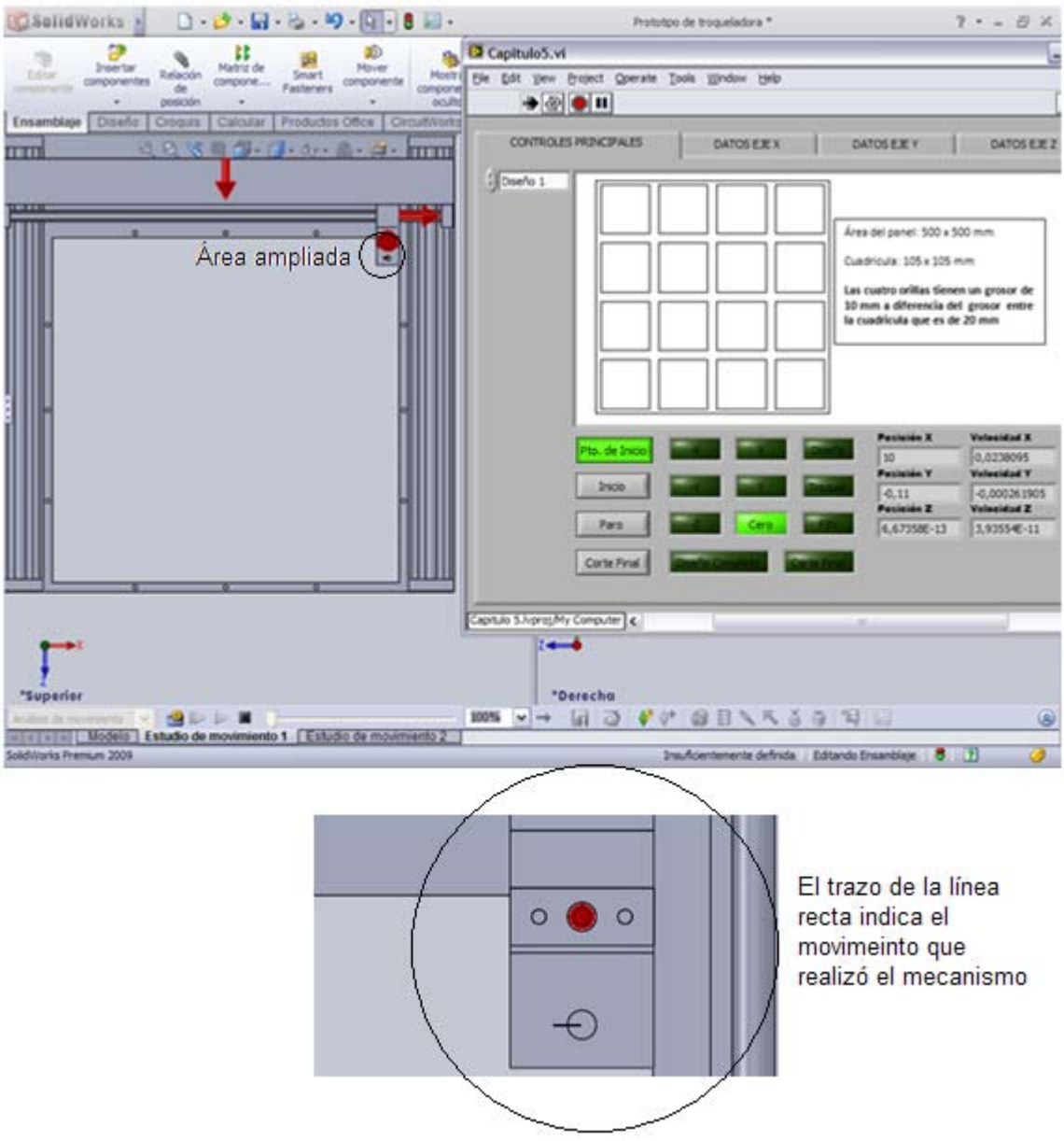
Ahora, se describirá paso a paso cómo se comporta la aplicación de *LabVIEW*, controlando los movimientos de simulación de la troqueladora. El primer paso para poner en marcha la aplicación es llevarla al punto de inicio, para lo cual se debe accionar el botón llamado Pto. de inicio en la pestaña controles principales del panel de control.

Las imágenes que verán serán de la vista superior del mecanismo y del panel de control, se omitirá la segunda vista del mecanismo que aparece en la figura 148 para poder observar con mayor detalle cada paso que ejecuta el mecanismo.

Presionando el botón Pto. de inicio el usuario ordenará al mecanismo moverse de su posición actual a la posición 10 milímetros sobre el eje X y -0.11 milímetros sobre el eje Y. En el panel de control verá reflejada dicha acción, con dos eventos. El primero es notar cómo se activan los indicadores numéricos para posición y velocidad de cada uno de los ejes involucrados en dicho movimiento.

El segundo evento que tendrá lugar en el panel de control al mover el mecanismo al punto de inicio será ver como se activa el indicador, presente en la pestaña controles principales, llamado “cero” mientras la aplicación traslada el troquel hasta el punto (10, -0.11). Una vez alcanzada dicha posición el indicador se desactivará y el usuario sabrá que el ajuste del mecanismo en el punto de inicio estará listo, la figura 149 ilustra.

Figura 149. **Ejecución de rutina para posicionamiento del mecanismo en el punto de inicio**

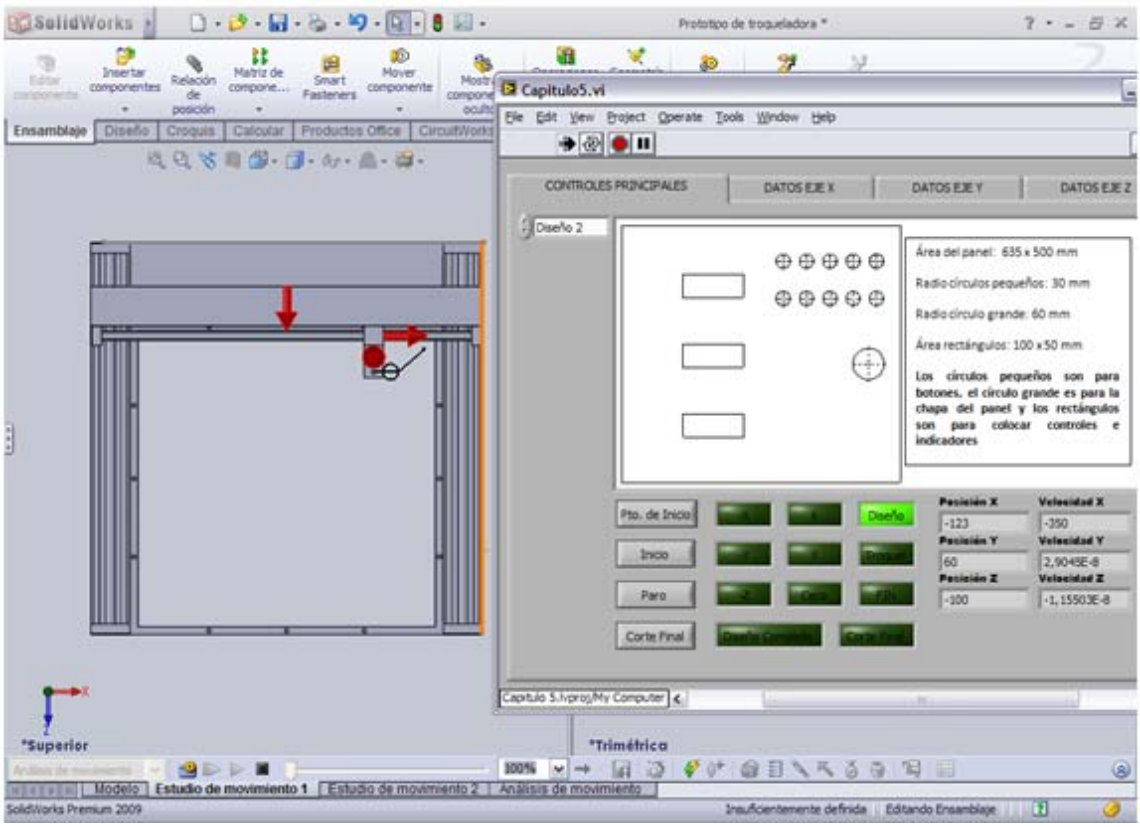


Luego de haber llevado el mecanismo al punto de inicio, está listo para realizar el diseño que ha sido previamente seleccionado por el usuario, para este caso se ha escogido el diseño 2. Entonces para iniciar con la acción de troquelado se debe presionar el botón con el nombre diseño, dicha acción hará, no solo que el mecanismo inicie a moverse basado en el diseño, sino que los indicadores de diseño, troquel, posición y velocidad, presentes en la pestaña controles principales, cumplan con su función y mantengan informando constantemente al usuario de cómo se está llevando a cabo el proceso.

Inmediatamente después de haber presionado el botón, se activará el indicador de diseño mientras la máquina se mueve al punto donde debe iniciar a troquelar, luego se desactivará éste y se activará el indicador de troquel, que indica que el troquel está descendiendo para perforar el material, y permanecerá activo hasta que éste haya descendido hasta el punto necesario para realizar un corte limpio del material.

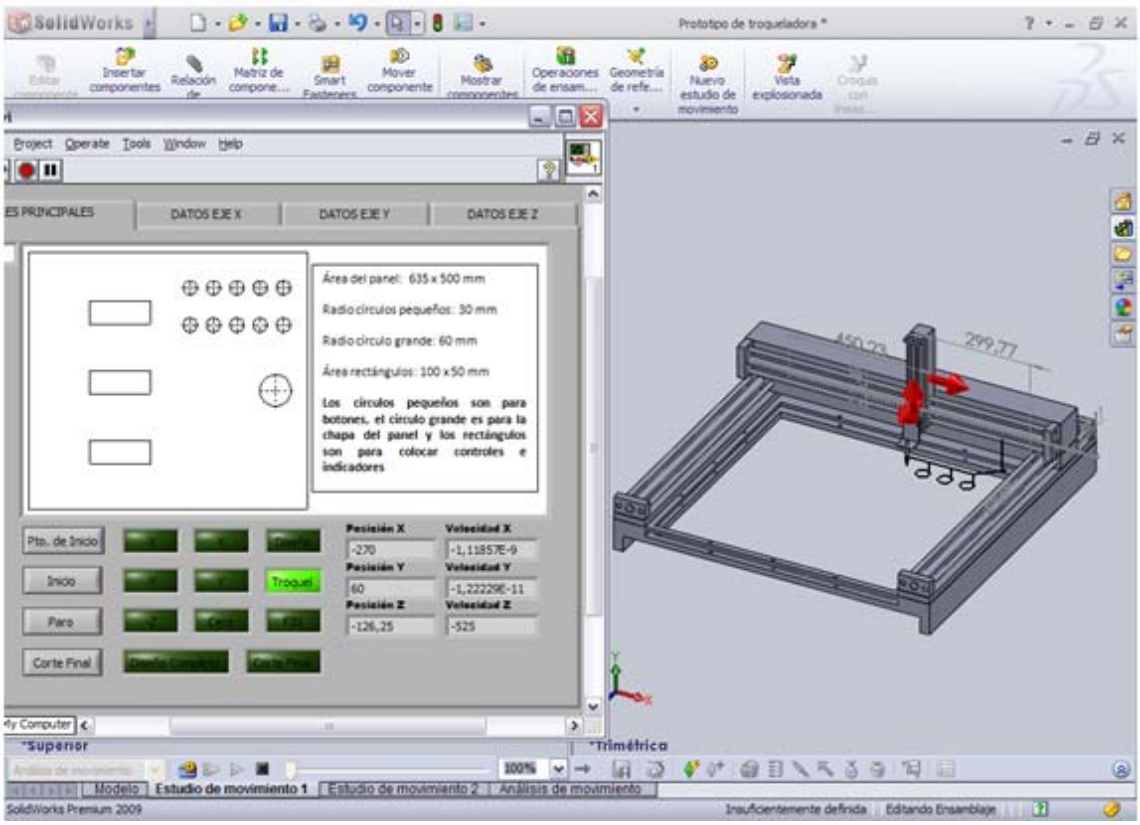
Entonces, el indicador de troquel se desactivará y una vez más el indicador de diseño estará activo mientras se traza un círculo, cuando se ha terminado de trazar el círculo, el indicador de diseño se apagará por segunda vez y el indicador de troquel se activará de nuevo, esta vez indicando que el troquel está ascendiendo hasta un punto determinado para brindar movilidad al mecanismo cuando se mueva para buscar la nueva posición y descender para troquelar un nuevo círculo. Este procedimiento intermitente, se llevará a cabo una y otra vez hasta que el diseño esté completo. La figuras 150, 151 y 152 lo ilustran.

Figura 150. **Indicador de diseño activado, mientras el mecanismo se mueve de acuerdo al diseño seleccionado**



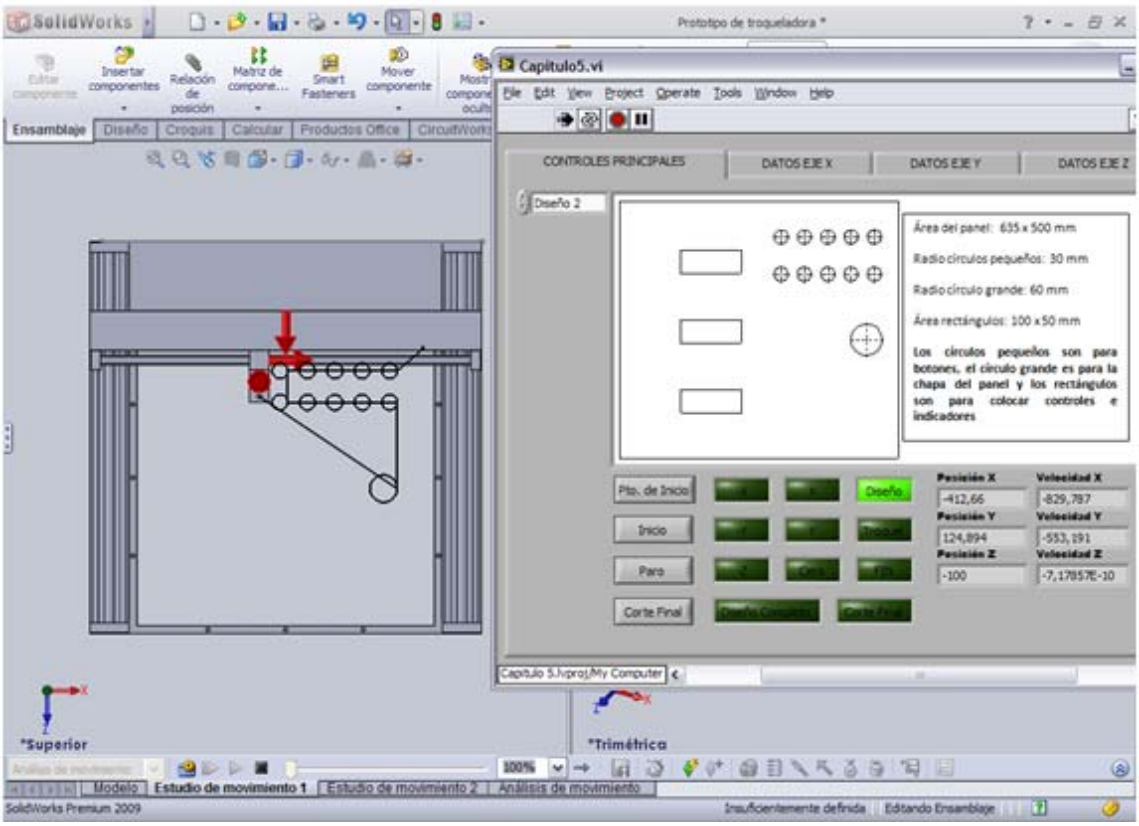
Como puede observar el indicador de posición para el eje Z marca -100, el valor necesario para que el mecanismo se pueda mover con libertad sin dañar la pieza. El indicador de diseño está activo ya que el mecanismo está llevando el troquel hasta el nuevo punto de troquelación.

Figura 151. **Indicador de troquel activado, mientras el troquel desciende para perforar el material**



En esta figura se ve como el indicador de troquel está descendiendo hasta alcanzar el punto necesario para perforar el material, el indicador para la posición del eje Z marca -126.25 y en el mecanismo se ve como el troquel dibuja una silueta hacia abajo.

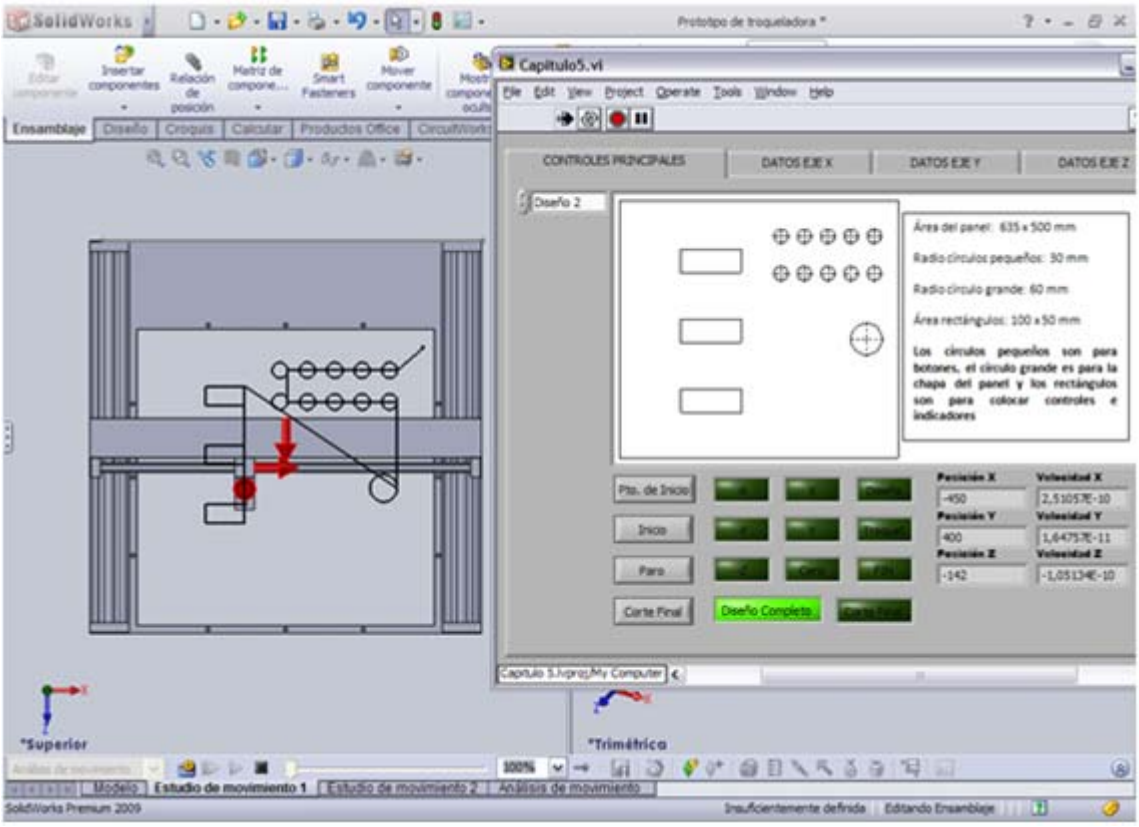
Figura 152. **Indicador de diseño activo una vez más, lectura del indicador de posición del eje Z en -100 milímetros**



De nuevo se repite el ciclo y el indicador de diseño está activado, mientras el mecanismo se dirige hacia el punto de perforación para el primero de tres rectángulos. El indicador de posición para el eje Z, una vez más entrega una lectura de -100, indicando que el mecanismo tiene libertad para moverse con el troquel funcionando sin dañar el diseño.

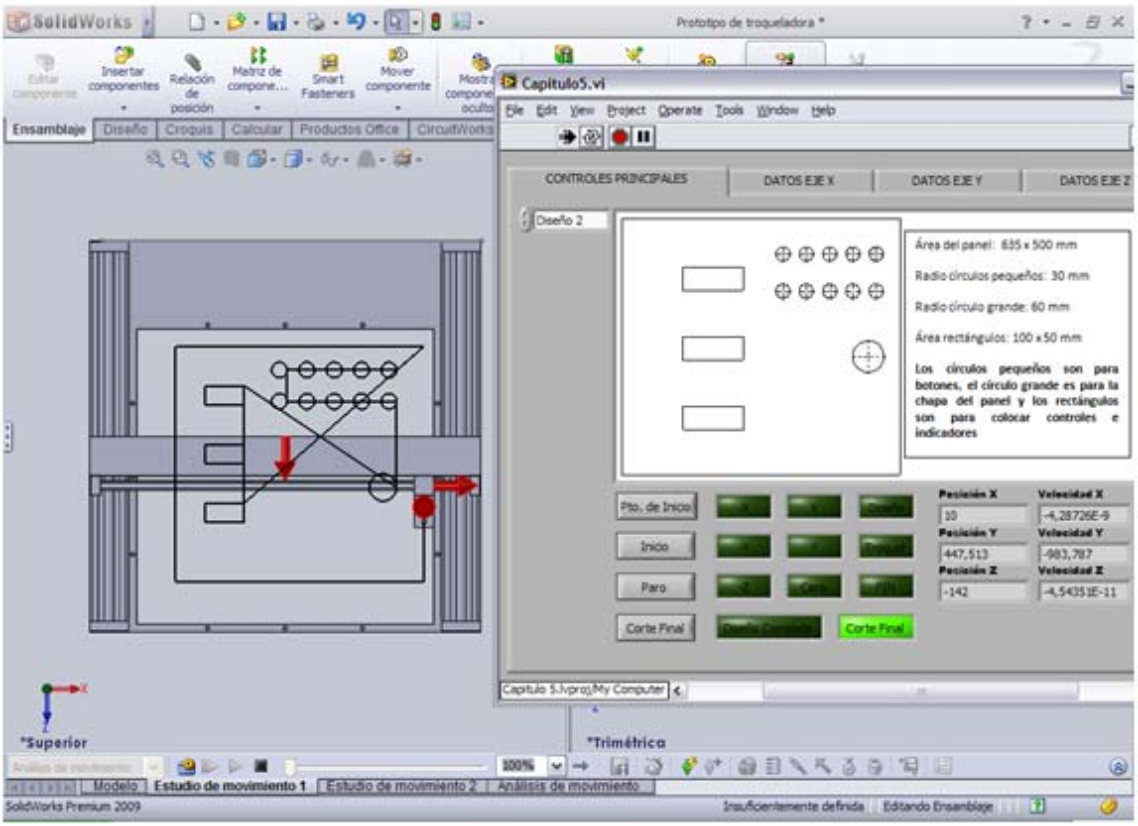
Ahora que se ha concluido con el proceso del diseño interno de la pieza, el indicador llamado diseño final se activará tal y como lo muestra la figura 153 y será hora del siguiente paso en la secuencia de creación de la pieza.

Figura 153. **Indicador de diseño completo activado**



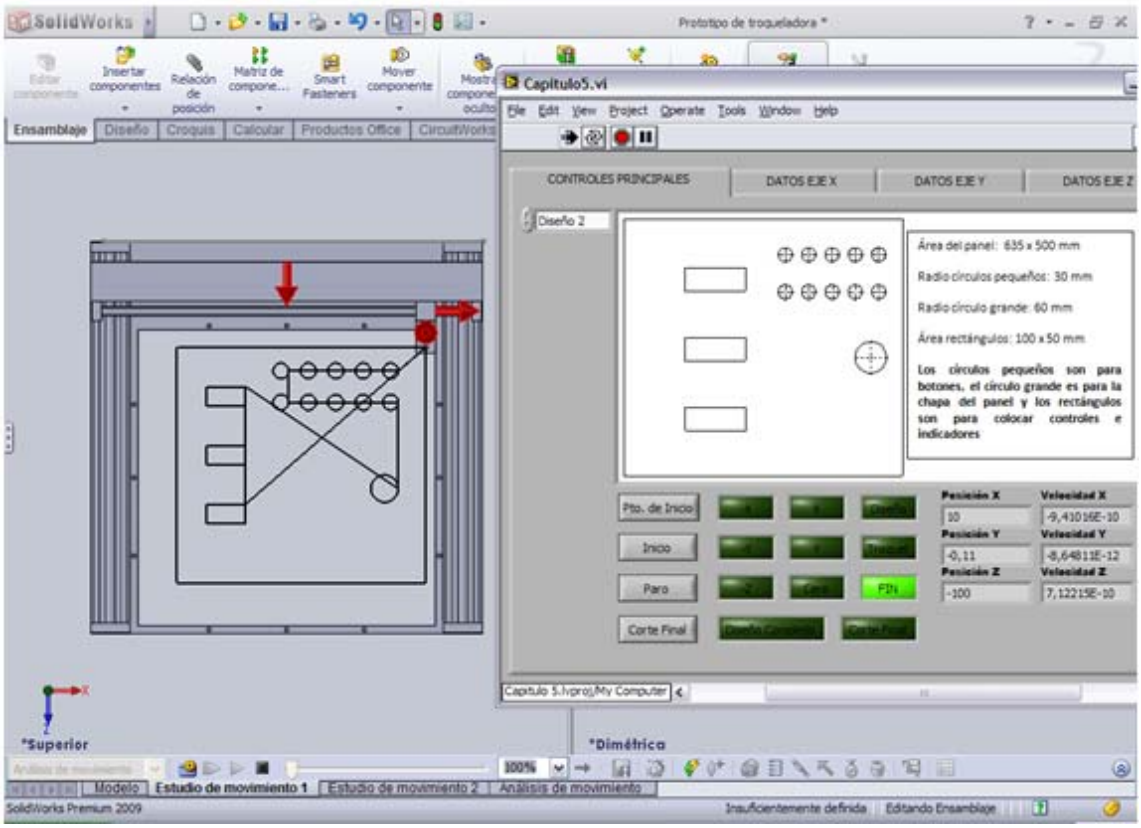
Con el diseño completo es hora que brindar un tiempo de espera, una especie de *stand by* dentro del proceso, para que el usuario coloque algo que detenga la pieza, de manera que cuando el contorno de la misma sea cortado, ésta no caiga al suelo. Para iniciar el corte final basta con presionar el botón bajo el mismo nombre, la imagen 154, ilustra el corte del contorno o corte final de la pieza.

Figura 154. **Corte del contorno de la pieza, indicador de corte final activado**



Una vez presionado el botón de corte final, el mecanismo se posicionará en el punto de inicio, descenderá el troquel, iniciará con el corte del contorno, una vez terminado dicho corte, el corte final de la pieza, el troquel de nuevo subirá a su posición de seguridad (-100 milímetros) e inmediatamente después de que el troquel ha alcanzado dicho punto el indicador de fin se activará para informar al usuario que la pieza está completa, la figura 155 ilustra.

Figura 155. **Indicador de fin activo, el proceso de la pieza está completo**



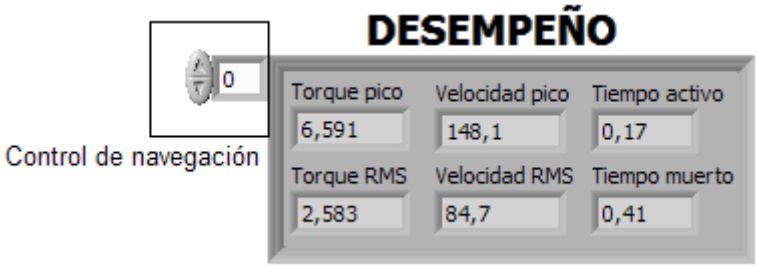
De esta manera es cómo funcionará el prototipo virtual generado en *Solidworks* y controlado desde *LabVIEW.* Exactamente el mismo orden en la ejecución de eventos se sigue para realizar el diseño 1, además el comportamiento de los controles e indicadores tanto de *Solidworks* como de *LabVIEW*, serán los mismos, obviamente ajustándose a los parámetros del diseño 1.

Ahora se hablará de las tres pestañas restantes, las pestañas de datos para el eje X, Y y Z. Estas pestañas pueden ser observadas en cualquier punto del proceso y brindaran dos tipos de datos, datos gráficos y datos numéricos. En los datos gráficos el usuario observará una gráfica correspondiente a la posición del eje bajo análisis, también una gráfica para la velocidad y una correspondiente al torque. Dichos datos hacen referencia al desempeño del motor ligado al eje.

Por el otro lado tenemos los datos numéricos, que están situados en la parte inferior de la ventana. Un dato importante acerca de este indicador, cuyos resultados son obtenidos por el *subVI motion analysis*, es que registrará y almacenará las características del motor cada vez que éste sea activado. Por ejemplo, para el motor que maneja el eje del troquel (eje Z), el *motion analysis* capturará los datos ya mencionados desde que éste se activa hasta que se apaga, y la próxima que sea activado tomará una nueva lectura y así sucesivamente.

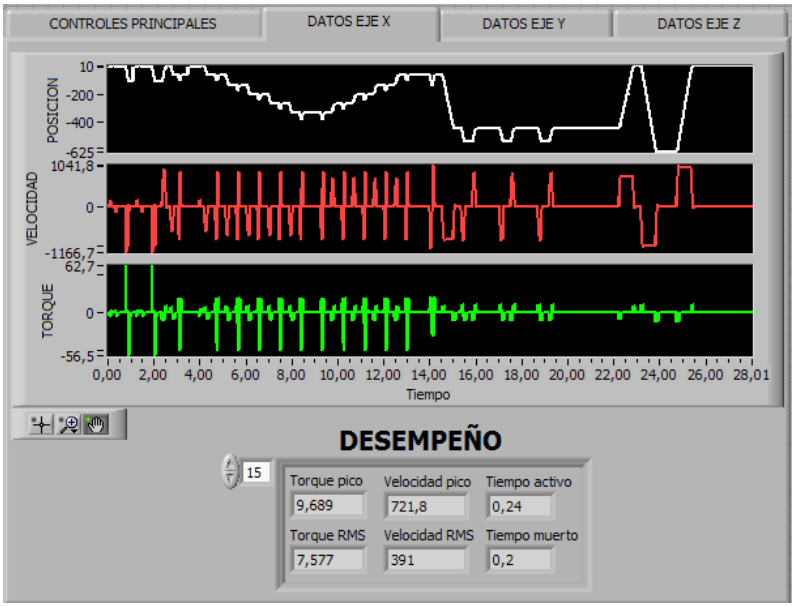
Para observar los distintos datos capturados, como torque pico, torque RMS, velocidad pico, velocidad RMS, tiempo activo en segundos y tiempo muerto en segundos en cada lectura realizada, basta con subir o bajar con el control para navegación con que cuenta dicho indicador y que es mostrado en la figura 156. En promedio para cada diseño, esta herramienta capturará 16 lecturas para cada motor ligado a cada eje de movimiento.

Figura 156. **Control de navegación para observar distintos datos capturados**



Es importante mencionar que antes de ejecutar la aplicación se debe colocar dicho control a cero, de lo contrario, por ejemplo si el usuario lo dejó en la captura 14, éste estará en dicha lectura y al inicio del proceso no verá nada, sino hasta alcanzar la lectura 14. A continuación las figuras 157, 158 y 159 mostrarán los datos capturados para cada eje durante la creación de la pieza con el diseño 2.

Figura 157. **Datos capturados eje X durante la realización del diseño 2**



En la parte inferior de la gráfica para el eje X se puede ver como la ejecución duró aproximadamente 26 minutos, además de las gráficas del motor y el indicador del desempeño del motor situado en la captura número 15.

Figura 158. **Datos capturados eje Y durante la realización del diseño 2**

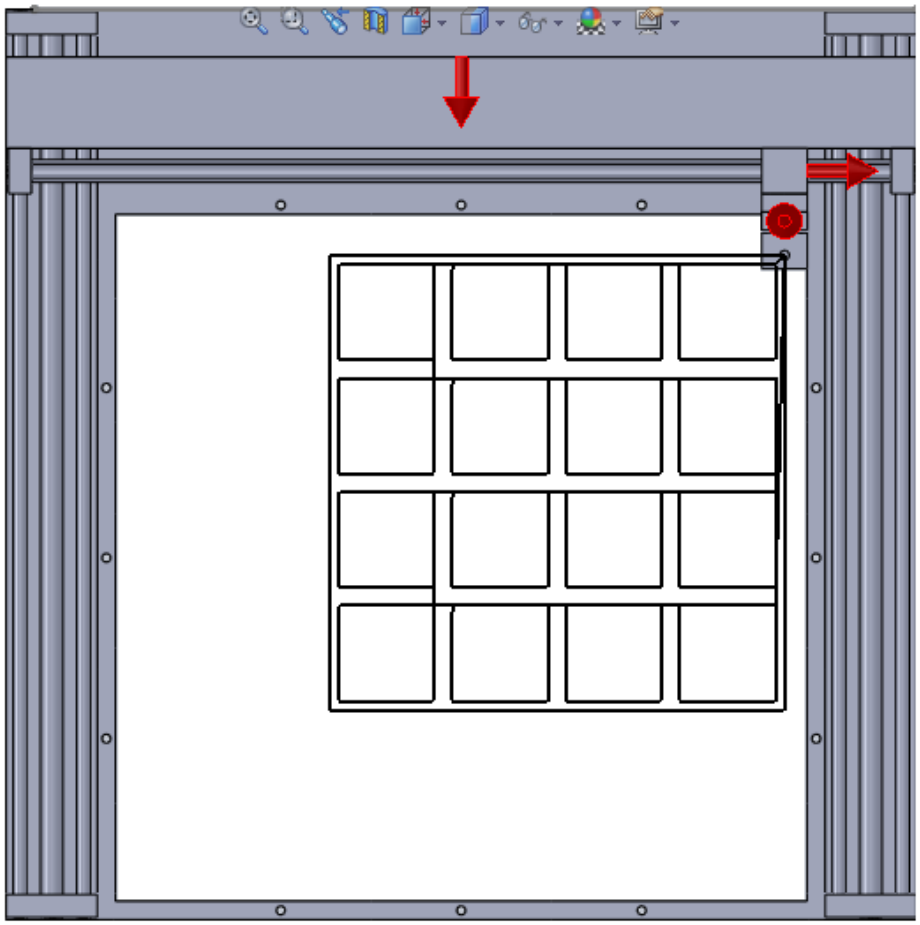
Con respecto a los datos para el motor del eje Y, podemos ver que los tiempos de ejecución concuerdan con los del eje X, 26 minutos. Sin embargo es notable la diferencia entre la gráfica de posición, cosa que no sucede en las otras dos gráficas para velocidad y torque que son bastante similares. En este caso el control de capturas está posicionado en la quinta.

Figura 159. **Datos capturados eje Z durante la realización del diseño 2**

Para el caso del eje Z las cosas son completamente diferentes, como es de esperar, ya que éste sólo ejecuta movimientos hacia arriba y hacia abajo. Incluso las gráficas para torque y velocidad son completamente diferentes, en esta ocasión el control de datos capturados indica 29. De nuevo aclaro que los datos entregados por esta herramienta son preliminares y requieren de análisis por parte de expertos en dimensionamiento de motores.

Finalmente se mostrará como lucen las trayectorias trazadas por el mecanismo una vez finalizado el diseño. Podrá observar dos vistas del mecanismo, la vista superior y la vista trimétrica, de tal manera que pueda seguir, en la medida de lo posible los movimientos realizados por la troqueladora durante la realización de ambos diseños. La figura 160 muestra el diseño 1 y la figura 161, las vistas para el diseño 2.

Figura 160. **Vistas del mecanismo con las trayectorias creadas para el diseño 1**



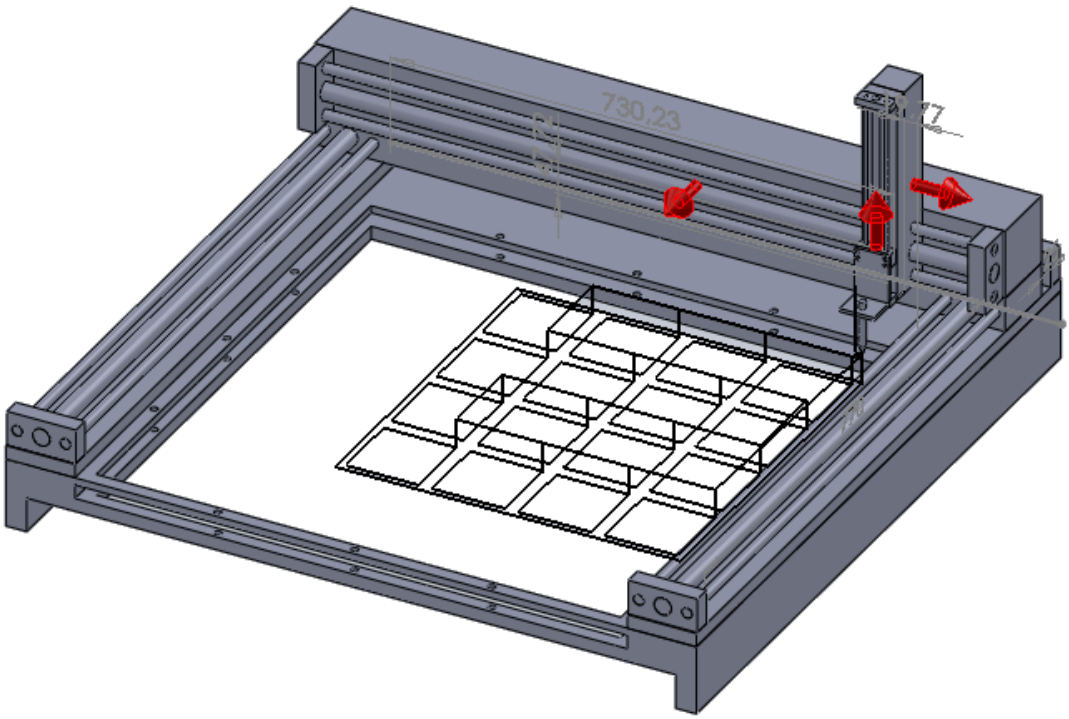
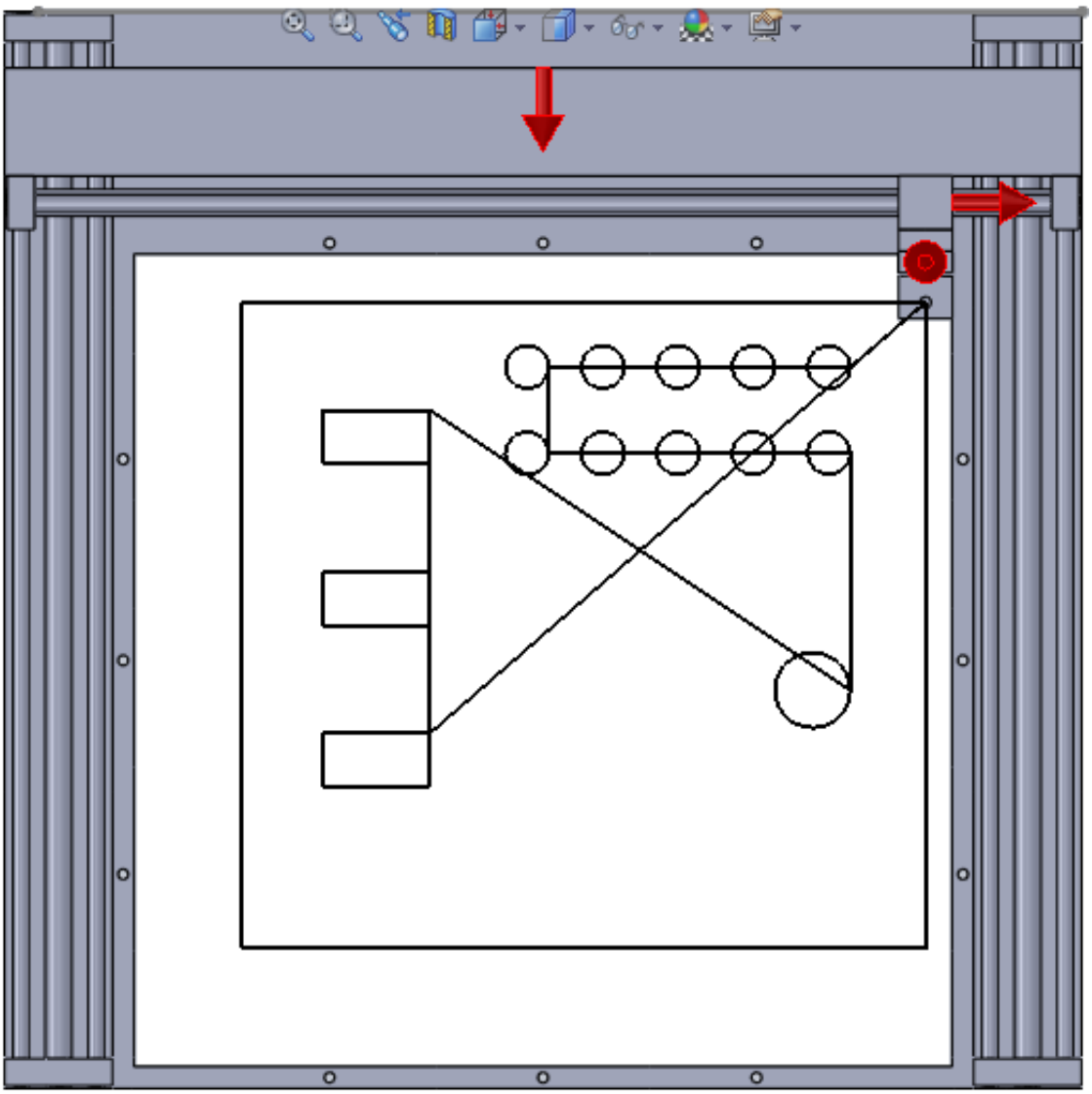
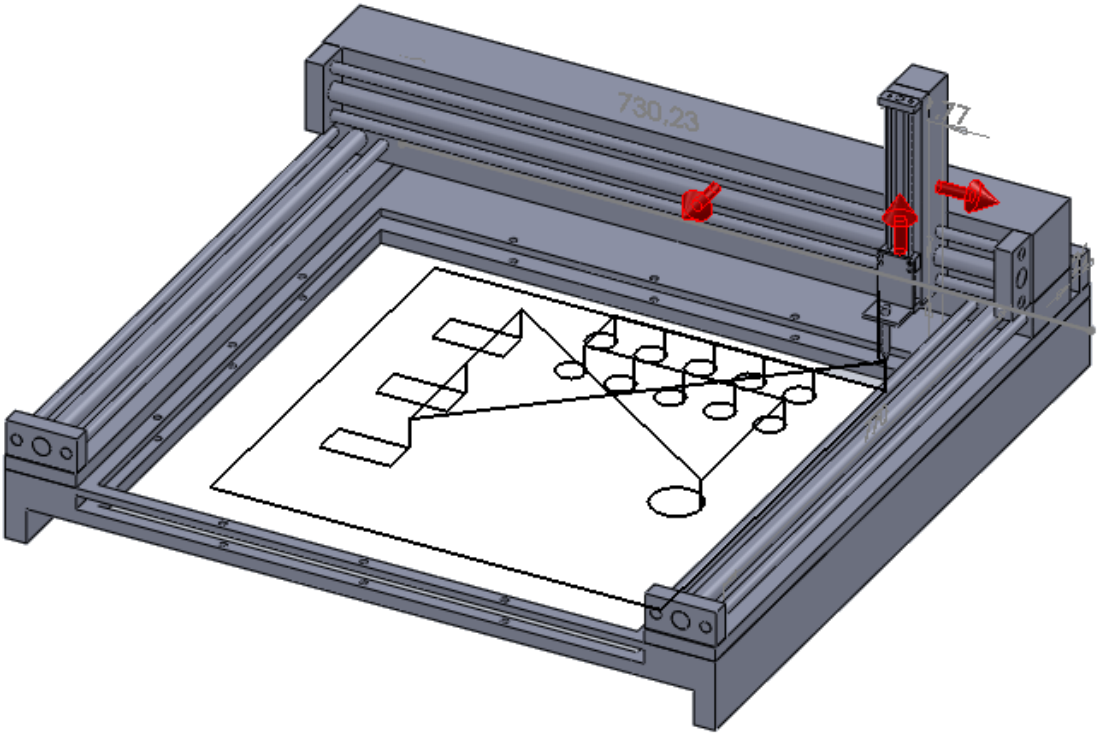

730.23

Figura 161. **Vistas del mecanismo con las trayectorias creadas para el diseño 2**




730.23

CONCLUSIONES

1. El conocimiento de la teoría básica sobre servomotores, sus partes y funcionamiento, es un tema vital dentro del área de automatización industrial.

2. Es importante tener claro que un servomecanismo es básicamente un sistema de control con retroalimentación, dentro del cual son controladas, en su mayoría, variables industriales como temperatura, presión, nivel de algún líquido, etc.

3. Dentro de la teoría de control para aplicaciones industriales en general, el control PID es una herramienta vital, razón por la cual debe conocerse y entenderse de manera adecuada.

4. El nombre troqueladora como tal, hace referencia a varios procesos industriales, cuya característica común es la perforación y/o corte de un material. Para este trabajo de graduación la troqueladora diseñada se encarga de cortar el material en una forma específica.

5. Un estudio de movimiento, dentro de *Solidworks*, representa el ambiente de simulación, el cual está basado en un solucionador de ecuaciones de cinemática que responden a las características del mecanismo bajo análisis.

6. Los NI *softmotion axis*, son las herramientas que permiten reconocer un motor creado en *Solidworks* dentro del ambiente de *LabVIEW*.

7. Una ventaja importantísima que presenta el NI *softmotion for Solidworks module*, es que no necesita mayor conocimiento sobre control numérico para iniciar con la programación de movimientos.

8. Debido a que el modulo NI *softmotion for solidworks* está basado en los estándares *PLCOpen* para control de movimiento, el conocer cómo funciona dicha herramienta, será algo aprovechable con aplicaciones de terceras marcas basadas en el mismo estándar.

RECOMENDACIONES

1. Debido al constante surgimiento de nuevas aplicaciones para el control industrial, así como la mejora de las aplicaciones ya existentes; es importante mantener una constante investigación acerca de las actualizaciones y mejoras para ambas aplicaciones.

2. Antes de intentar realizar una aplicación similar a la expuesta en este trabajo de graduación es muy importante que se conozca y se sepa utilizar las herramientas del núcleo de *LabVIEW*, ya que de esta manera será mucho más fácil entender y desarrollar aplicaciones de manera más eficiente.

3. Si la persona interesada no tiene conocimiento alguno sobre la utilización de programas para diseño CAD, es mejor investigar y empaparse un poco sobre los conceptos básicos; y aunque no es necesario poseer conocimientos avanzados, el saber un poco más allá de lo básico lo habilitará para desarrollar sus propias ideas y no depender de diseños creados por terceras personas.

4. Otro camino a tomar para obtener resultados mucho mejores que los presentados en este trabajo de graduación es utilizar el NI *motion assistant,* otra de las herramientas que presenta *LabVIEW* para control de movimiento.

BIBLIOGRAFÍA

1. ALCIATORE, David y HISTAND, Michael B. *Introduction to mechatronics and measurement systems.* 2a. ed. Estados Unidos: McGraw Hill, 1999. 502 p.

2. DASSAULT, Systemès. *Software de diseño 3D.* [en linea]. Estados Unidos: 2009 [de febrero a abril de 2010]. Disponible en web: < http://www.solidworks.es/sw/products/mechanical-engineering-cad-software.htm>

3. FALK, Dietmar. *Metalotecnia fundamental.* Barcelona: Reverté, 1986. 363 p.

4. IEEE. *El origen de los servos.* IEEE Industry Applications Magazine, 1996, vol.74. Estados Unidos.

5. INDUSTRIAS RAMFÉ. *Frenos electromagnéticos ramfé.* Colombia: Industrias ramfé. 2 p.

6. JOHNSON, Olaf A. *Diseño de máquinas herramienta.* México: Roble, 1973. 260 p.

7. KOSOW, Irving L. *Máquinas eléctricas y transfomadores.* 2a. ed. Mexico: Prentice-Hall Hispanoamérica, 1993. 502 p.

8. McCLEERY, Bryan. *Virtual prototyping training series.* [seminario web]. Estados Unidos: National Instruments, agosto 2009 [abril de 2010]. Disponible en web: http://www.ni.com/virtualprototyping/training.htm>

9. SMITH, Carlos y CORRIPIO, Armando. *Control automático de procesos teoría y práctica.* Mexico: Limusa, 1991. 585 p.

10. TRAVIS, Jeffrey y KING, Jim. *LabVIEW for everyone: graphical programming made easy and fun.* 3a. ed. Estados Unidos: Pentice-Hall, 2007. 981 p.