
# Programación de Robots Lego Mindstorms

Pablo Iván Romero de La Rosa, Leodegario G. Aguilera Hernández y Orlando Meza Zaleta

ISBN-13: 978-84-15774-15-0

Editado por la Fundación Universitaria Andaluza Inca Garcilaso para eumed.net

# PROGRAMACIÓN DE ROBOTS LEGO MINDSTORMS

**PABLO IVÁN ROMERO DE LA ROSA**

**LEODEGARIO G. AGUILERA HERNÁNDEZ**

**ORLANDO MEZA ZALETA**

**INSTITUTO TECNOLÓGICO SUPERIOR DE TANTOYUCA**

# INDICE

# AGRADECIMIENTOS

Este libro nació del esfuerzo de nuestros familiares y amigos que se privaron de nuestra presencia mientras escribíamos y realizábamos todos y cada una de los ejercicios de programación que están plasmados en este sencillo aporte a la técnica, a ellos nuestro total agradecimiento por la comprensión demostrada.

Deseamos expresar un reconocimiento y mención muy especial a la atenta y desinteresada ayuda de los integrantes del Departamento de Estudios de Posgrado e Investigación así como también a los directivos y administrativos del Instituto Tecnológico Superior de Tantoyuca.

# INTRODUCCIÓN

## *¿Qué es LEGO-Mindstorms?*

Imagina poder construir un robot completo, con sensores, motores, engranajes, reductoras, estructuras, poder programarlo y configurarlo, y todo sin soldar, taladrar, pegar o taladrar tornillos. Pues eso es LEGO-Mindstorms, una forma fácil y sencilla de aprender robótica y construir tu propio robot.

Lego Mindstorms es una plataforma para el diseño y desarrollo de robots, que sigue la filosofía de la marca LEGO, armar y construir todo tipo de objetos simplemente uniendo bloques interconectables. Pues eso es LEGO Mindstorms.

El bloque central es un microcontrolador, con forma de LEGO. La conexión de sensores y actuadores es muy sencilla, por simple presión en cualquiera de las puertas y en cualquier posición. Las piezas de Lego tienen múltiples formas y tamaños, lo que permite construir diversas estructuras, usando los bloques. Mediante un PC, se realiza la programación del controlador, usando diferentes programas y lenguajes.

## *¿Por qué usar LEGO-Mindstorms?*

En este apartado se indicaran las principales ventajas y desventajas de utilizar LEGO - Mindstorms.

**Ventajas:**

-Fácil de montar y desmontar, no es necesario usar soldadura, ni tornillos. Todo lo que se arma se puede desarmar rápidamente. Además, eso permite usar las piezas en múltiples diseños.

-Muy extendido por todo el mundo, lo que permite encontrar gran cantidad de información e ideas por Internet, diseños, soluciones, participar en foros, competiciones.

-No es un pack cerrado, es decir, se puede comprar más ampliaciones de LEGO, adquirir piezas deterioradas o perdidas, o añadir piezas echas manualmente, como por ejemplo, sensores o motores, e incluso circuitos neumáticos.

-Múltiples posibilidades y lenguajes de programación, desde el nivel más básico e intuitivo, hasta el uso de lenguajes conocidos como C o Java, e incluso la utilización de Linux…

-Que sea escalable, es decir, que a partir de un material básico haya opciones de ampliación.

-Muy indicado para entornos educativos, desde colegios a universidades, pues se puede aprender de forma fácil tanto mecánica como electrónica.

**Desventajas**

-La principal desventaja de LEGO es su estructura. Está formada por bloques de LEGO, que se unen por simple presión. Cierto que se pueden añadir elementos de refuerzo y sujeción, pero para diseños exigentes, no es recomendable. Golpes, caídas, pueden debilitar rápidamente la estructura, llegando a desarmar el robot.

-No se pueden construir estructuras circulares, pues todas las piezas y ladridos de LEGO son rectangulares.

-Colocación de las baterías. Tanto en el NXT cómo en el RCX de LEGO, se alimentan mediante seis pilas AA R6, que deben ser colocadas dentro del controlador. Esto obliga a diseñar el robot con la necesidad de acceder directamente al bloque, para poder cambiar las pilas, limitando la construcción del robot.

-Relación masa-volumen. Las piezas LEGO no son útiles en diseños donde la relación masa-volumen se hace crítica. Por ejemplo, para construir un robot de SUMO, no sería eficiente, pues la estructura LEGO es demasiado liviana, y se deberían añadir pesos para hacer el robot más robusto, o el caso contrario, para construir robot pequeños, ligeros, y resistentes, las piezas LEGO son mucho peores que los materiales cómo la fibra de carbono.

-Precio. Obviamente, comprar un robot “prefabricado”, resulta más caro que construirte tu propio robot.

## *El NXT*

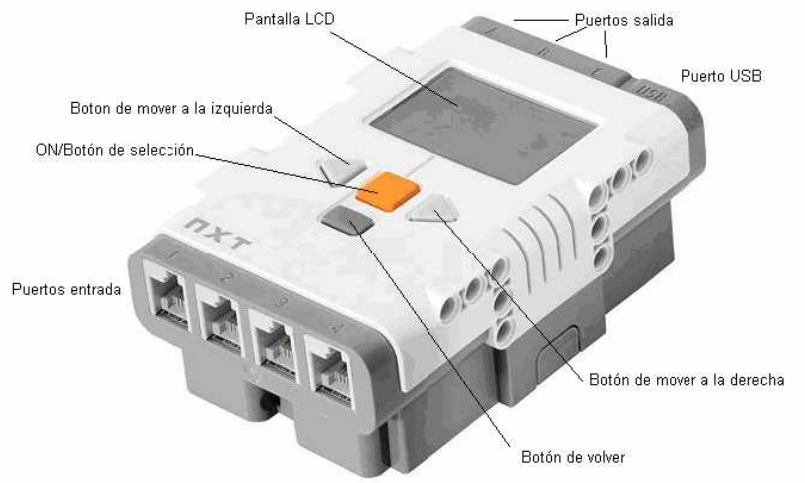



**Figura 1:** Controlador NXT.

En la Figura 1 se muestra el controlador NXT, es el objeto donde reside todo el control del robot. Para ello está compuesto por un microprocesador ARM-7 de 32 bits, el AT91SAM7S256 de Atmel. Este microprocesador de arquitectura RISC incluye 256 de memoria flash (no volátil), 64 kB (volátil) y una velocidad de funcionamiento de 48 MHz.

Tiene cuatro botones en la parte superior para utilizar los programas que tenga instalados, configurarlos y ejecutarlos. También puede visualizar en la pantalla el estado de los sensores o crear pequeños programas sin necesidad de utilizar el ordenador y programas adicionales. El botón de color naranja tiene como funciones el encendido (ON), la confirmación de acciones (Enter) y el comenzar (Start). Las flechas de color gris son para moverse por los menús y el rectángulo gris oscuro es para limpiar la pantalla (Clear) y volver atrás (Go back).

## *Transductores y Sensores:*

**Sensor:**

Dispositivo sensible que utiliza un fenómeno físico o químico dependiente de la naturaleza y el valor de la magnitud físico química a medir, lo cual permite la transducción del estímulo a una señal utilizada directa o indirectamente como medida.

Como sabemos un sensor es un dispositivo capaz de detectar diferentes tipos de materiales, con el objetivo de mandar una señal y permitir que continúe un proceso.

**Transductor:**

Un transductor es un dispositivo que transforma un tipo de variable física (por ejemplo, fuerza, presión, temperatura, velocidad, etc.) en otro.

Un sensor es un transductor que se utiliza para medir una variable física de interés. Algunos de los sensores y transductores utilizados con más frecuencia son los calibradores de tensión (utilizados para medir la fuerza y la presión), los termopares (temperaturas), los velocímetros (velocidad).

Cualquier sensor o transductor necesita estar calibrado para ser útil como dispositivo de medida. La calibración es el procedimiento mediante el cual se establece la relación entre la variable medida y la señal de salida convertida.

Los transductores y los sensores pueden clasificarse en dos tipos básicos, dependiendo de la forma de la señal convertida.

Los dos tipos son:

**Transductores analógicos:** Proporcionan una señal analógica continua, por ejemplo voltaje o corriente eléctrica. Esta señal puede ser tomada como el valor de la variable física que se mide.

**Transductores digitales:** Producen una señal de salida digital, en la forma de un conjunto de bits de estado en paralelo o formando una serie de pulsaciones que pueden ser contadas. En una u otra forma, las señales digitales representan el valor de la variable medida. Los transductores digitales suelen ofrecer la ventaja de ser más compatibles con las computadoras digitales que los sensores analógicos en la automatización y en el control de procesos.

## *Sensores LEGO*

### Sensor táctil

El sensor táctil es un interruptor: puede presionarse o liberarse.

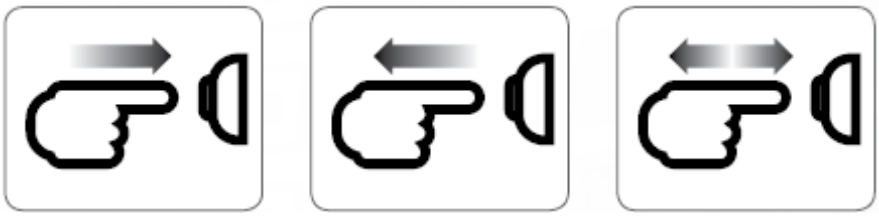


**Figura 2:** Opciones de programación del sensor táctil.

### Sensor acústico

El sensor acústico detecta el nivel de decibeles: la suavidad o intensidad de un sonido. El sensor acústico detecta dB y dBA. dBA: sonidos que el oído humano es capaz de oír. dB: todos los sonidos existentes, incluyendo los sonidos demasiado altos o bajos para el oído humano.

El sensor acústico puede medir los niveles de presión acústica hasta 90 dB, cerca del nivel de una máquina cortacésped. Las lecturas del sensor acústico en el LEGO® MINDSTORMS® NXT se muestran en el porcentaje [%] de sonido que el sensor es capaz de leer. A modo de comparación, entre 4 y 5 % es similar a una sala en silencio y entre 5 y 10 % es cerca del nivel de alguien hablando a cierta distancia. De 10 a 30 % es una conversación normal cerca del sensor o música que se reproduce a un nivel normal y

un nivel entre 30 y 100 % representa un intervalo desde personas gritando hasta música reproduciéndose a volúmenes altos. Estos intervalos se asumen a una distancia de 1 metro aproximadamente entre la fuente del sonido y el sensor acústico.

**Sensor fotosensible**

El sensor fotosensible le permite al robot distinguir entre luminosidad y oscuridad, para obtener la lectura de la intensidad de luminosidad en una habitación y para medir la intensidad de luminosidad sobre superficies de colores.

**Sensor ultrasónico**

El sensor ultrasónico le permite al robot ver y reconocer objetos, evitar obstáculos, medir distancias y detectar movimiento.

El sensor ultrasónico utiliza el mismo principio científico que los murciélagos: mide la distancia calculando el tiempo que demora una onda de sonido en golpear un objeto y volver, al igual que un eco.

El sensor ultrasónico mide la distancia en centímetros y pulgadas. Es capaz de medir distancias de 0 a 2,5 metros con una precisión de +/- 3 cm.

Objetos de gran tamaño con superficies duras proporcionan las mejores lecturas. Objetos hechos con telas suaves, con objetos curvados (por ejemplo una pelota) o con objetos muy delgados y pequeños pueden dificultar la obtención de lecturas del sensor.

## *Motores:*

**Motores ¿Para qué?**

En numerosas ocasiones es necesario convertir la energía eléctrica en energía mecánica, esto se puede lograr, por ejemplo, usando los motores de corriente continua. Los usos más habituales pueden ser:

- ✓ Tracción y dirección.
- ✓ Orientación de sistemas sensoriales.

**Principio de funcionamiento**

Los motores eléctricos, en general, basan su funcionamiento en las fuerzas ejercidas por un campo electromagnético y creadas al hacer circular una corriente eléctrica a través de una o varias bobinas. Si dicha bobina, generalmente circular y denominada estator, se mantiene en una posición mecánica fija y en su interior, bajo la influencia del campo electromagnético, se coloca otra bobina, llamada rotor, recorrida por una corriente y capaz de girar sobre su eje, esta última tenderá a buscas la posición de equilibrio magnético, es decir, orientará sus polos NORTE-SUR hacia los polos SUR-NORTE del estator, respectivamente. Cuando el rotor alcanza esta posición de equilibrio, el estator cambia la orientación de sus polos, aquel tratará de buscar la nueva posición de equilibrio; manteniendo dicha situación de manera continuada, se conseguirá un movimiento giratorio y continuo del rotor y a la vez la transformación de una energía eléctrica en otra mecánica en forma de movimiento circular.

**Motor de Corriente Continua**

Motor que funciona con corriente eléctrica continúa. El campo magnético se crea en el inducido (rotor) y en el inductor (estator). Necesitan un colector en el rotor y escobillas para su alimentación eléctrica. Este tipo de motor fue el primero que se utilizó en la tracción de los vehículos eléctricos por la simplicidad de los sistemas de control de revoluciones. Tiene un elevado mantenimiento por el desgaste de las escobillas y de los colectores por el alto consumo de corriente que tienen. En los motores de alta potencia, su tamaño llega a ser muy voluminoso.

En lugar de un armazón con un núcleo de hierro y muchos bobinados, hay una única espira conductora cuadrada girando alrededor de un eje, el cual no se dibuja.

**Servomotor LEGO**

Los tres servomotores interactivos le proporcionan al robot la capacidad de moverse. El bloque Desplazar [Move] automáticamente alinea sus velocidades para que el robot se mueva suavemente.

**Sensor de rotación integrado**

Todos los servomotores interactivos disponen de un sensor de rotación integrado. La retroacción rotacional le permite al NXT controlar los movimientos de forma muy

precisa. El sensor de rotación integrado mide las rotaciones del motor en grados (precisión de +/- un grado) o en rotaciones completas.

Una rotación son 360 grados, por lo tanto si configura el motor para que gire 180 grados, la pieza central de la rueda realizará medio giro.

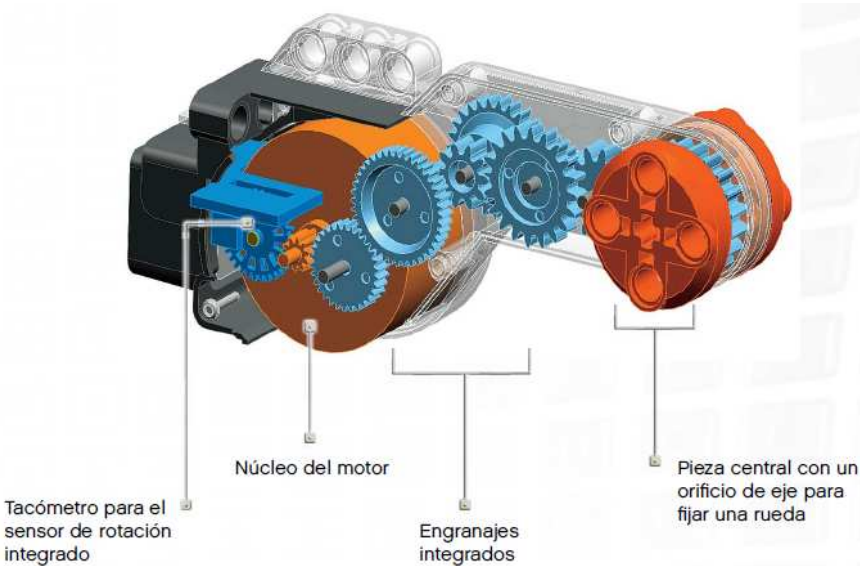



**Figura 3:** Estructura interna del servomotor LEGO.

# PROGRAMACIÓN CON LabVIEW ™

El propósito de esta unidad es introducir en LabVIEW ™. En esta sección se introducirá a través del proceso de navegación de LabVIEW™ y aprenderá a utilizar sus funciones. También se aprenderá las técnicas de ayuda, que va a construir una gran base para la programación en LabVIEW™ a medida que avanza en la programación específica para el NXT y robots LEGO ® MINDSTORMS ®.

En esta lección, aprenderá a utilizar LabVIEW ™. Se familiarizará con el ambiente de trabajo y aprenderá a hacer el trabajo con los controladores NXT y el robot LEGO ® MINDSTORMS ®.

LabVIEW™ es un lenguaje de programación como C + +, Java, Visual Basic o C del robot, es un poco diferente a los lenguajes de programación tradicionales basados en texto, ya que es totalmente gráfica. LabVIEW™ se compone de bloques en la pantalla conectados con cables. Los bloques son funciones, entradas y salidas.

Aplicaciones útiles de LabVIEW™:

- Adquisición de Datos y procesamiento de señales.
- Control de instrumentos.
- Automatización.
- Medición y Control de Sistemas Industriales.
- Diseño de Sistemas Embebidos.
- Investigación y Docencia.

Cuando comienza LabVIEW™, se encuentra con la pantalla de inicio. Puede utilizar este acceso a varios recursos, incluyendo la ayuda en línea, programas de ejemplo y foros de discusión. También puede utilizar la pantalla de introducción para abrir todos los programas que ha trabajado recientemente. Utilizará esta pantalla de introducción para empezar a trabajar en un nuevo programa. En el menú Nuevo, verá las opciones de VI en blanco, como se ve en la Figura 4.

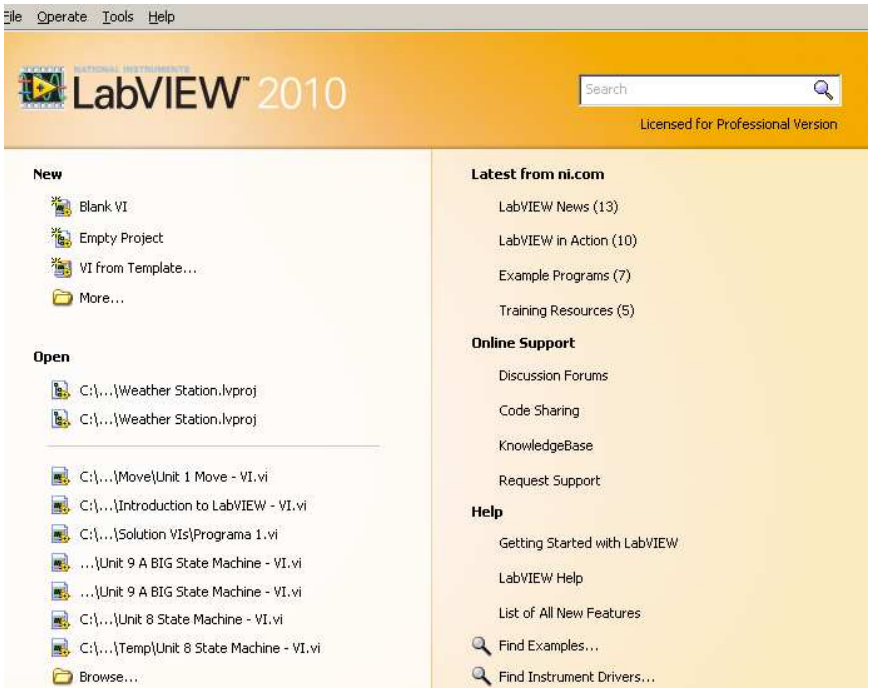



**Figura 4:** Pantalla de inicio de LabVIEW™.

Un VI es un acrónimo de instrumentos virtuales. Básicamente, un VI es un programa de LabVIEW™. ". Vi" es también la extensión de su programa cuando se guarda. Para iniciar la programación, seleccionará VI en blanco de la pantalla de inicio. Está seleccionando VI en blanco por el momento porque está creando un programa en la instancia principal de la aplicación. Un programa en la instancia de aplicación principal es cuando quiere crear un programa y hacer que se ejecute en su ordenador.

Verá dos ventanas. Una se llama panel frontal y la otra se llama diagrama de bloques. Estas se muestran en la Figura 5.

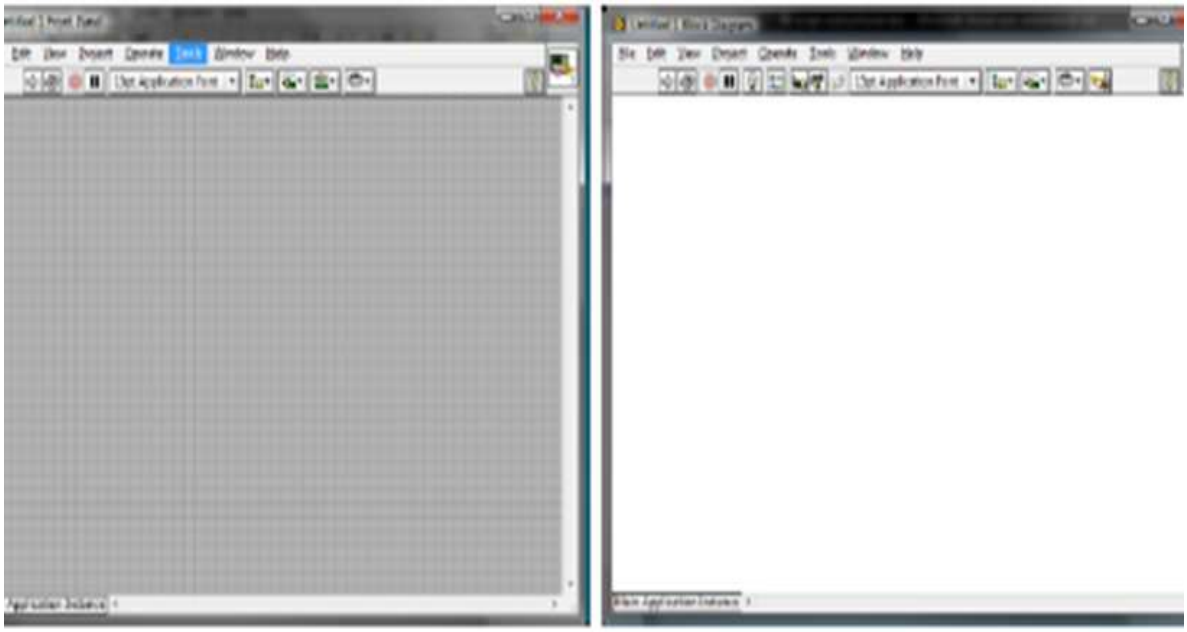


**Figura 5:** Panel frontal y Diagrama de bloques.

El panel frontal es como el panel frontal del controlador NXT. Aquí es donde puede ver los botones, las pantallas y los indicadores. Se trata de la interfaz de usuario de su programa. El diagrama de bloques es donde lleva a cabo su programación y aquí es donde su código irá. Su código le dice al hardware qué hacer y cuándo hacerlo. Los artículos que pone en el diagrama de bloques se llaman terminales y funciones. Los artículos que pone en el panel frontal se llaman controles e indicadores. Una cosa interesante a observar es que siempre puede cerrar el diagrama de bloques y tener acceso de nuevo desde el panel frontal. Puede hacerlo pulsando Ctrl + E. Este acceso directo le permite alternar entre el panel frontal y el diagrama de bloques. Cada vez que cierre el panel frontal, que es esencialmente cerrar el VI, le preguntará si desea guardar los cambios.

Explorará la barra de herramientas en el diagrama de bloques y el panel frontal como se muestra en la Figura 6.

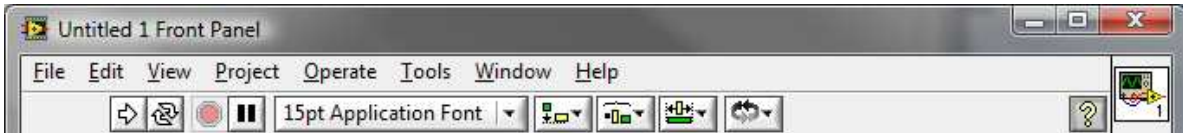



**Figura 6:** Barra de herramientas en el Panel frontal y en el Diagrama de bloques.

El botón de encendido se utiliza para ejecutar un programa. Esto ejecuta el código una sola vez. El botón de ejecución continua ejecuta un programa varias veces hasta que el

botón Cancelar ejecución se presiona. El botón de pausa puede ser utilizado para pausar un programa en el modo de funcionamiento continuo, o si tiene un programa de gran tamaño que está tomando mucho tiempo para ejecutar, puede hacer una pausa de su ejecución.

Con esto en mente, es importante mencionar que LabVIEW es un lenguaje de programación que se puede utilizar en la programación de aplicaciones para equipos industriales o para un controlador NXT. Las características y capacidades de ambos son diferentes. Para asegurarse de que el programa que está escribiendo está dirigido al sistema correcto, use la barra en la parte inferior izquierda de la ventana LabVIEW. Tiene que hacer clic derecho, verá que hay opciones de instancia de aplicación principal y NXT. El objetivo, es habilitar el NXT que tiene actualmente conectado. Si su programación está dirigida al NXT, seleccionará NXT: Target. Para todos los demás casos, se utiliza la instancia principal de la aplicación. Si tuviera que seleccionar la opción de NXT, esto abrirá un nuevo conjunto de ventanas, un diagrama de bloques y un panel frontal con una barra naranja en la parte inferior izquierda de cada ventana lo que significa que su código tiene como objetivo la programación del controlador NXT y tendrá opciones específicamente para la programación NXT. Ya no verá los menús y herramientas que no se aplican a la programación NXT. Por ahora se quedará en la instancia principal de la aplicación y creará un programa.

Al hacer clic derecho en cualquier parte del diagrama de bloques, verá una paleta de funciones, como se muestra en la Figura 7.

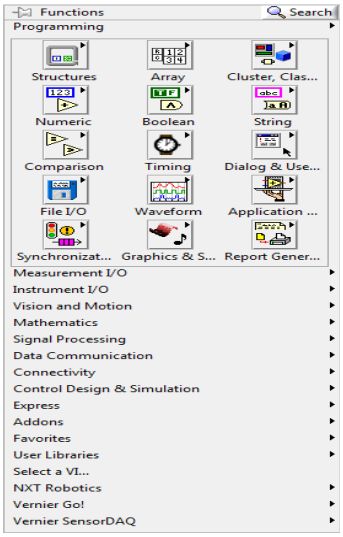



**Figura 7:** Paleta de funciones.

La paleta de funciones contiene todas las herramientas que necesita para escribir el código. Podrá ver que hay estructuras, numéricos y booleanos. Todas estas opciones se pondrán de manifiesto muy pronto. Si explora la I/O NXT, encontrará que tiene acceso a los controles de motor, sensores y otros instrumentos importantes específicos del NXT.

Al hacer clic derecho sobre el panel frontal, la Paleta de Controles aparece como se muestra en la Figura 8.

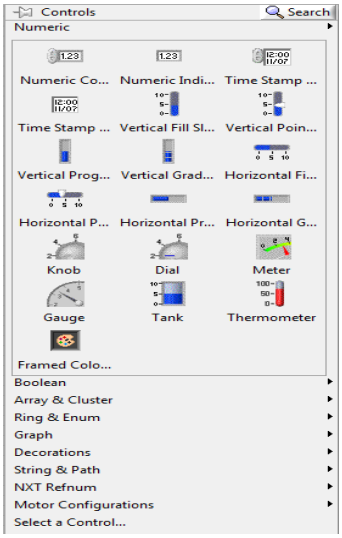



**Figura 8:** Paleta de controles.

La paleta de controles es donde se encuentran las opciones relacionadas con la interacción del usuario. Aquí, hay una variedad de controles, indicadores e incluso decoraciones.

Tanto la paleta de funciones como la paleta de controles pueden ser "inmovilizadas" por lo que rápidamente se puede acceder a ellas sin tener siempre que hacer clic derecho. Esto se puede lograr haciendo clic en el icono del alfiler en la parte superior izquierda de las paletas.

Para seleccionar un objeto, simplemente haga clic sobre él. Luego se muestra un esquema del objeto que ha seleccionado. Dé clic de nuevo para colocarlo en el panel frontal o diagrama de bloques. Recuerde que las funciones deben ir en el diagrama de bloques y los controles en el panel frontal. Seleccionar el botón y colocarlo en el panel frontal como se muestra en la Figura 9 y 10.

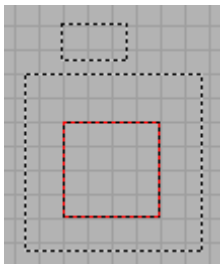


**Figura 9:** Esquema de la perilla antes de que se coloque en el panel frontal.


**Figura 10:** Perilla en el panel frontal.

Una característica interesante de LabVIEW ™ que se debe mencionar es el cursor. Puede haber notado que a medida que mueve el cursor, cambia el selector, para el puntero, a la selección. Esta es una característica de la herramienta de selección automática.

Una cosa a notar sobre la perilla y otros selectores numéricos similares a ella es que puede cambiar su escala. Para ello, basta con hacer doble clic en el valor máximo o mínimo, como el número 10 en la figura 10 y reemplazarlo con el máximo deseado o el valor mínimo.

Con el mando que ha puesto, si quiere cambiar el valor de ese botón, mueve el cursor a la perilla de manejar. Si hace clic y arrastra, es capaz de cambiar el valor de la perilla. Si quiere cambiar el tamaño de la perilla, pone el cursor en cualquier lugar por encima de la perilla y verá cuatro controladores de tamaño que aparecen. Puede utilizar estos para ajustar las dimensiones de la perilla a su preferencia. Si simplemente desea mover el mando, puede posicionarse por encima del borde del objeto y luego dar clic para mover el mando. Si quiere modificar la etiqueta, hacer doble clic en la etiqueta y puede modificarla. El comportamiento del cursor con el mando es similar a su comportamiento con la mayoría de los objetos que tiene en el panel frontal o el diagrama de bloques. Si quiere desactivar la herramienta de selección automática y seleccionar manualmente la funcionalidad del cursor, seleccione el menú Ver en la barra de herramientas y haga clic en la paleta de Herramientas. Con ello se abre la barra de herramientas tal como se muestra en la Figura 11.


**Figura 11:** Paleta de herramientas.

El botón en la parte superior con la llave y un destornillador junto a la barra verde que está seleccionada por defecto, significa que el selector automático de herramientas está activado. Para desactivarlo, haga clic en él, por lo que ya no es destacado, y a partir de ese momento, se debe seleccionar manualmente qué forma quiere que el cursor tome y la funcionalidad que quiere que tenga. El selector automático de herramientas es una preferencia personal de la mayoría de los programadores, y es muy recomendable, ya que ahorra el tiempo valioso del programador.

Una característica muy útil de LabVIEW es la ayuda contextual. Para acceder a la ventana de ayuda contextual, se selecciona el menú Ayuda de la barra de herramientas y haga clic en Ayuda de Contexto. Esto también se puede hacer pulsando Ctrl + H o haciendo clic en el icono de signo de interrogación en la parte superior derecha de su

panel frontal o diagrama de bloques. Esto mostrará una pequeña ventana en la pantalla, como la que se muestra en la Figura 12.

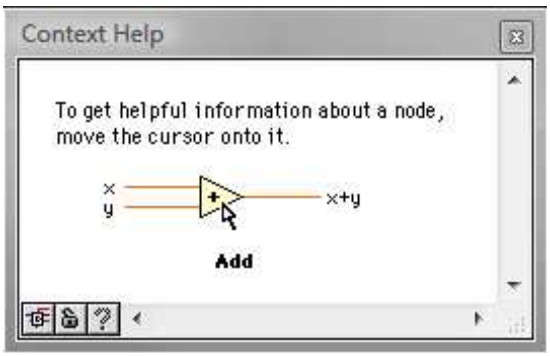



**Figura 12:** Ayuda de contexto.

La ayuda contextual da toda la información sobre cualquier cosa sobre la que el cursor se posicione. La ventana de ayuda contextual también cuenta con un enlace de Ayuda de detalle, puede hacer clic para obtener una explicación detallada de todo sobre lo cual el cursor se posicione. La ayuda más detallada también proporciona ejemplos de cómo utilizar las funciones y otros objetos.

Hasta ahora ha colocado un control en el panel frontal, también colocará un indicador. Básicamente, los controles son las entradas y los indicadores son las salidas. Para colocar un indicador, haga clic derecho en cualquier parte del panel frontal y elija cualquiera de los indicadores. En este caso elegirá el medidor.

Ahora bien, si se dirige al diagrama de bloques, de manera interesante, podrá ver el botón y el indicador que ha colocado en el panel frontal, como se ve en la Figura 13 o 14. Esto muestra que el panel frontal y diagrama de bloques están unidos entre sí de forma permanente, en el sentido de que todos los controles en el panel frontal también se encuentran en el diagrama de bloques.


**Figura 13:** Perilla y medidor en el diagrama de bloques.


**Figura 14:** Iconos alternativos para la perilla y el medidor.

Los dos iconos que tiene en el diagrama de bloques, el control y el medidor, se llaman terminales. Verá que hay un triángulo negro en la terminal de mando, y se encuentra ubicado en la derecha. En el medidor, el triángulo negro se encuentra en la izquierda. Esto puede ayudar a identificar cuáles de estos terminales son los controles y cuáles son indicadores. Controles, con datos que salen de ellos, por lo tanto, el triángulo negro sale de la terminal. Los indicadores tienen datos escritos en ellos, como el medidor, por lo tanto, el triángulo negro apunta hacia el terminal. Otra forma de diferenciar entre los controles y los indicadores en el diagrama de bloques es buscar en la frontera de la terminal. Un borde grueso significa que es un control y un borde fino indica que es un indicador.
Si se posiciona sobre el triángulo negro en la terminal de mando, se dará cuenta de que el cursor se convierte en un rollo de alambre, como se muestra en la Figura 15.

**Figura 15:** Herramienta de cableado.

Si hace clic una vez para iniciar el cableado entre la perilla y el medidor, verá que conecta las terminales como se muestra en la Figura 16.


**Figura 16:** Perilla y medidor conectados.

Para borrar un hilo, se posiciona sobre él hasta que el cursor se convierte en un puntero. Luego, hace clic una vez y pulse Suprimir en el teclado. Si alguna vez por error, hace clic en el triángulo negro en una terminal y un cable empieza a dejarse arrastrar, se puede dar clic derecho o pulse Esc en el teclado y dejará de dibujar ese cable. Si está en la mitad de un hilo, se puede pulsar la barra espaciadora para alternar entre los diferentes caminos que el cable puede tomar, como se muestra en la Figura 17.


**Figura 17:** Diferentes caminos que puede tomar el cableado.

También se puede forzar el cable para tomar un camino específico después de haber iniciado el cable en el triángulo negro de un terminal. Cuando se pulsa una vez en

cualquier parte del diagrama de bloques y se va a colocar el alambre. Puede repetir este procedimiento tantas veces como quiera y definir cualquier camino que quiera que tome el cable antes de llegar a su destino. Un ejemplo se muestra en la Figura 18.

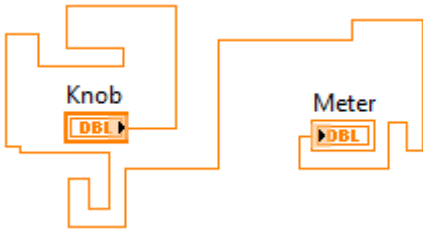



**Figura 18:** Forzando el camino del cableado.

Después de colocar un cable, si no está satisfecho con su ubicación, puede usar la opción Limpiar Cableado. Para hacer esto, haga clic derecho en el cable y seleccionar Limpiar Cableado, y LabVIEW automáticamente elige el camino más corto para el cable. Esto se muestra en la Figura 19.

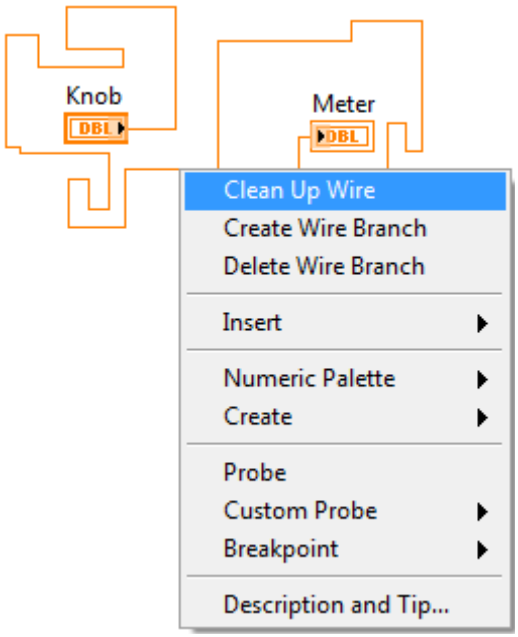



**Figura 19:** Herramienta de limpieza de cableado.

Si no le gusta la ruta que LabVIEW ha elegido para su cableado, puede deshacer la acción para restaurar la forma del cableado original. La opción Deshacer también se puede acceder utilizando el atajo Ctrl + Z.

Otra forma de organizar no sólo nuestro cableado, sino el resto del código es utilizando la herramienta Limpiar diagrama. Se encuentra sólo en la ventana del diagrama de bloques en la barra de herramientas del diagrama de bloques de la parte superior derecha representada por una escoba, como se muestra en la Figura 20.

**Figura 20:** Herramienta Limpiar diagrama.

En cualquier momento, si hace clic en el botón Limpiar diagrama, automáticamente comprime su código en la forma más legible y compacta, así como organiza los objetos en el diagrama de bloques. Esto se muestra en la Figura 21.

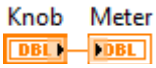



**Figura 21:** El código limpio se logra utilizando la herramienta Limpiar diagrama.

Una vez más, si no le gusta la forma en que LabVIEW ha reorganizado el código, puede deshacer la acción y regresar el código a su formato original.

Si tiene dos o más terminales conectados entre sí a través de un cable, y uno de ellos es eliminado, se obtendrá un cable roto como resultado. Esto deja un cable con una "X" roja, como se muestra en la Figura 22. Los cables rotos pueden también ser resultado de cableado de dos objetos con tipos de datos incompatibles. Hay muchas maneras de eliminar los cables rotos, pero el método más fácil es presionar Ctrl + B en el teclado.


**Figura 22:** Cable roto.

Ahora tiene su primer programa de LabVIEW. Esta es una buena oportunidad para guardar el programa. Guarda un VI es como guardar un archivo en cualquier otra aplicación, como por ejemplo un documento de Word o una presentación de PowerPoint. Seleccionar el menú Archivo de la barra de herramientas y hacer clic en

Guardar como. A continuación, selecciona el directorio de destino y el nombre del archivo. Al hacer clic en Guardar guardará su VI en su disco duro.

**Taller:** Utilice una perilla para mostrar su posición numérica con un medidor, un termómetro y un manómetro y medir al mismo tiempo. Editar el valor máximo para ajustar los datos con claridad en el medidor, termómetro y manómetro.

# ¡MUÉVETE!

Comienza en LabVIEW seleccionando un VI en blanco dirigido a NXT desde la pantalla de inicio, como se muestra en la Figura 23. Hace esto porque quiere crear un programa que se ejecute en el controlador NXT y que controla el robot LEGO MINDSTORMS.

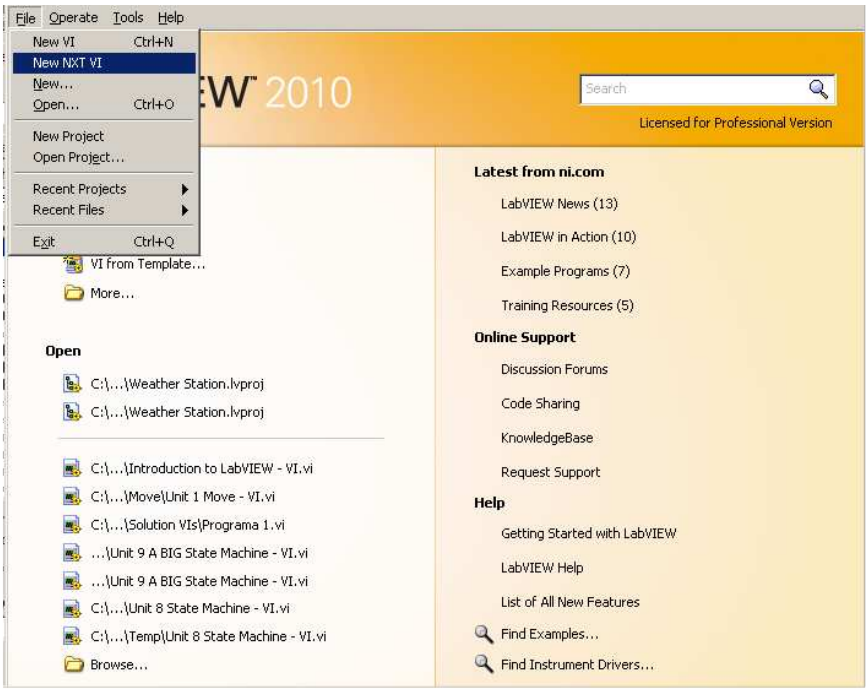



**Figura 23:** Seleccionando un VI en blanco para el NXT.

Para crear un programa para el robot LEGO MINDSTORMS ha seleccionado un VI en blanco o si está en la instancia principal de la aplicación, puede hacer clic derecho en el rectángulo en la parte inferior izquierda del panel frontal o diagrama de bloques que dice instancia de aplicación principal. Esto se muestra en la Figura 24.

Después de hacer clic derecho, debería ver un menú como el que se muestra en la Figura 25.

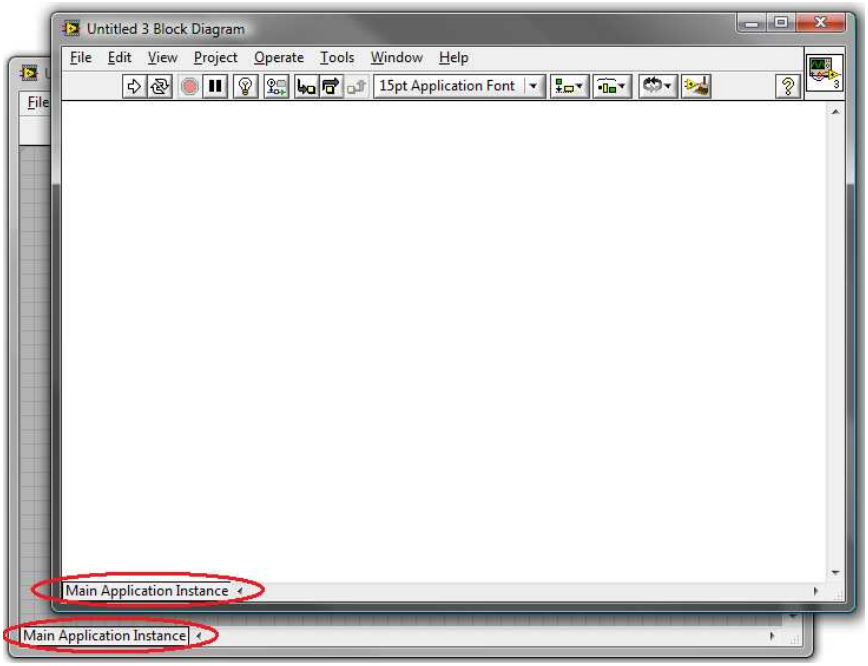



**Figura 24:** Intancia de aplicación principal.

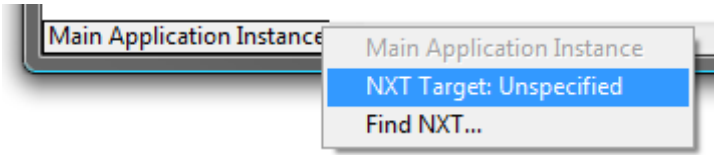



**Figura 25:** Menú para cambiar entre instancias.

Puede hacer clic en NXT Target: Unspecified, para cambiar a la instancia que se dirige al controlador NXT y ahora puede empezar a programar específicamente el controlador NXT.

La diferencia entre la instancia principal de la aplicación y la instancia NXT es que son diferentes las funciones que están disponibles en la instancia NXT y por lo tanto las paletas son ligeramente diferentes. Algunas funciones disponibles en la instancia principal de la aplicación no son compatibles con el NXT por lo que no aparecen en la paleta en la instancia NXT.

Comenzará a explorar la instancia dirigida al controlador NXT. Hacer clic derecho en el diagrama de bloques para que aparezca la paleta de funciones y luego seleccione el menú I / O NXT y verá una sub-paleta que se utiliza muy a menudo en la programación de los robots LEGO MINDSTORMS. Esto se muestra en la Figura 26.

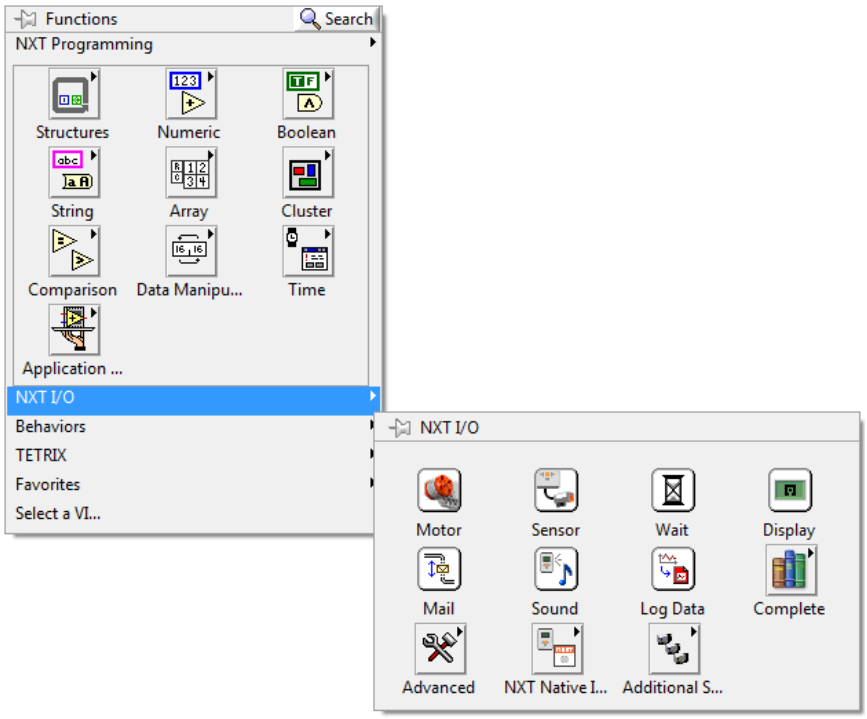



**Figura 26:** Menú I/O NXT.

Explorará algunos de los elementos de la sub-paleta de I / O NXT. Si se posiciona sobre estos iconos, puede utilizar la ayuda contextual para que le ayude en la comprensión de ellos.

Si se posiciona sobre el icono de control de motores, puede ver que la ayuda contextual le muestra lo que los puertos de entrada y de salida de esa función hacen. Esto se muestra en la Figura 27.

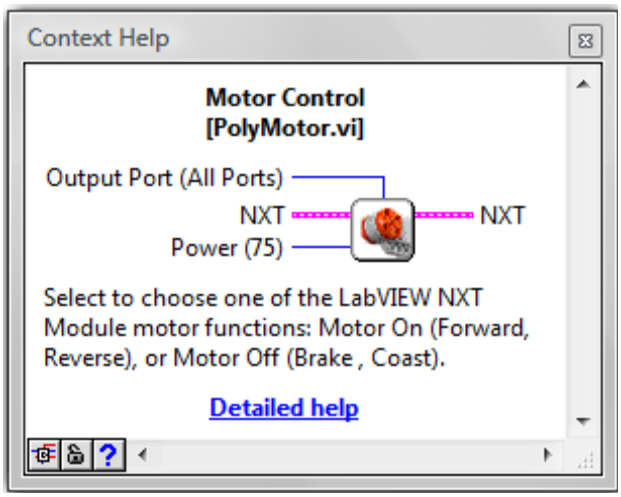



**Figura 27:** Ayuda del Control del motor.

Verá que hay dos terminales en este bloque, ambos llamados NXT. Hay una entrada y una salida NXT. En las unidades siguientes, aprenderá qué es esto, pero en pocas palabras, se utiliza para la secuencia de elementos juntos. También verá que hay una

terminal de alimentación y una para el puerto de salida. El puerto de salida es donde le dirá al programa en qué puerto se encuentra el motor. Cuenta con un puerto de alimentación que le dice al programa la velocidad a la que el motor debe girar. Seleccione el icono de control de motores y colóquelo en el diagrama de bloques. Esto se muestra en la Figura 28.


**Figura 28:** Control del motor.

Cuando se selecciona, el control del motor por defecto aparece seleccionado la opción de moverse en la dirección de avance. Donde dice: Fwd, también hay un selector para abrir un menú desplegable. Si da clic en el menú desplegable, puede ver que hay dos opciones, como se muestra en la Figura 29.


**Figura 29:** Selector de opciones del control del motor.

El menú del Control del Motor tiene las opciones para hacer que el motor vaya en la dirección hacia adelante o hacia atrás. Esto puede ser útil si quiere hacer que el robot se mueva en diferentes direcciones. También está el menú de paro del motor, donde puede elegir que el motor se apague por inercia hasta detenerse o puede optar por detener inmediatamente el motor mediante la selección de freno. Utilizará el menú de paro del motor ampliamente en las unidades posteriores.

La mayoría de las funciones que va a utilizar van a tener este tipo de menú desplegable. Se le llama Selector polimórficos. Esto es sólo una forma elegante de decir que esta función se puede trabajar de manera diferente en diferentes situaciones. También puede notar que a medida que se posiciona sobre las diferentes opciones en el menú polimórfico, el contexto de ayuda cambia las descripciones.

En este caso, sólo quiere que el robot se mueva hacia adelante, así que dejará su estado predeterminado. Ahora configurará el motor para hacer que el robot se mueva hacia adelante. Va a empezar por la creación de una constante para especificar la velocidad del motor. Como ya ha aprendido, hacer clic derecho sobre el terminal,

seleccionar Crear y hacer clic en Constante. Ahora tiene la posibilidad de introducir el número de la velocidad del motor, que va desde 0 hasta 100. Es importante tener en cuenta que la velocidad óptima es de 75, por lo que si decide no seleccionar una velocidad usted mismo, LabVIEW selecciona la velocidad de 75 automáticamente. Esto se muestra en la Figura 30.


**Figura 30:** Una velocidad seleccionada.

Lo siguiente que tiene que hacer es especificar el puerto donde está ubicado el motor. Para ello, hacer clic derecho en la terminal del puerto de salida, y una vez más, seleccionar Crear y hacer clic en Constante. Esto crea un menú desplegable conectado al puerto de salida con un cable como se muestra en la Figura 31.

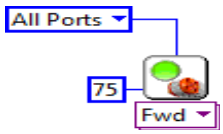



**Figura 31:** Selector de puerto.

El menú por defecto selecciona todos los puertos. Si quiere especificar exactamente en qué puerto se encuentra el motor, hacer clic para abrir el menú desplegable y seleccionar el puerto. Este puerto es el mismo puerto al que está conectado el motor en el controlador NXT, como se muestra en la Figura 32.

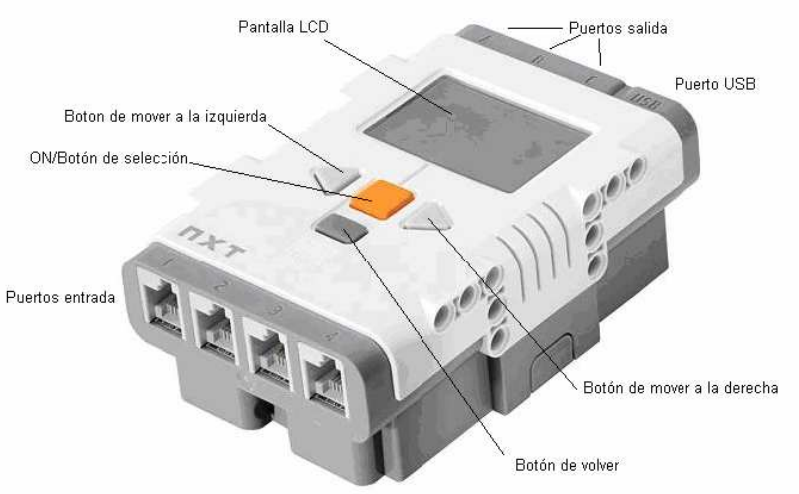



**Figura 32:** El controlador NXT.

Si quiere que el robot se mueva hacia adelante, necesita controlar dos motores: uno para mover el motor unido a la rueda derecha, y otro para mover el motor unido a la

rueda izquierda. Para ello, necesita tener dos funciones de control de motores, uno para cada motor.

Hay una forma rápida de duplicar una función en el diagrama de bloques: posicionarse sobre el objeto, pulsar Ctrl en el teclado, y hacer clic y arrastrar a donde quiera colocar el objeto. Esto se muestra en la Figura 33.

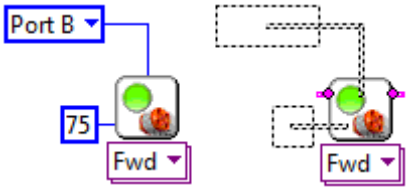



**Figura 33:** Duplicando objetos en el diagrama de bloques.

Una vez que tiene las dos funciones de control de motores en el diagrama de bloques, tiene que asegurarse de que ha seleccionado los puertos adecuados para cada uno, como se muestra en la Figura 34.

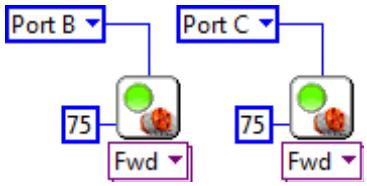



**Figura 34:** Configuración para controlar dos motores.

Una vez que ha conectado los motores a los mismos puertos que ha seleccionado en las terminales de control de motores, estará listo para ejecutar el programa y ver que el robot se mueva. El robot utilizado en los capítulos de programación con LabVIEW se muestra en la Figura 35.

**Figura 35:** Robot utilizado.

Ahora que tiene un programa funcional, debe trasladarlo a su controlador NXT y luego ejecutarlo. Ha examinado brevemente el controlador NXT al principio de esta unidad, así que va a aprovechar esta oportunidad para aprender sobre el controlador NXT en detalle.

El controlador NXT es el cerebro de los robots LEGO ® MINDSTORMS ®. Tiene un potente microprocesador de 32 bits y memoria Flash. Se muestra en la Figura 36.

El NXT puede obtener información de hasta cuatro sensores y tener el control de hasta tres servo motores, todo a través de sus 7 puertos. También tiene una pantalla LCD y cuatro botones. Los botones se utilizan para navegar por los menús del controlador NXT. El controlador NXT está alimentado por 6 baterías AA o una batería recargable.

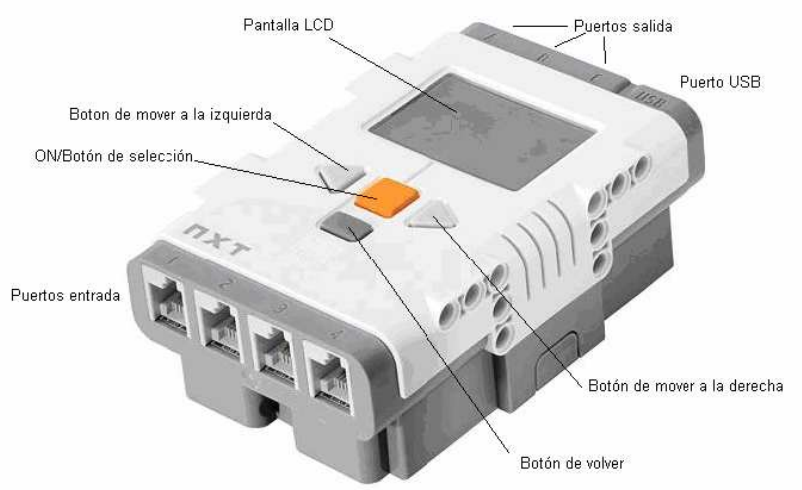



**Figura 36:** El controlador NXT.

Los dos botones de flecha gris se utilizan para navegar por los menús del controlador NXT. El botón de color naranja se utiliza para seleccionar objetos en el menú y el botón rectangular gris oscuro se utiliza para finalizar un programa o navegar en el menú principal del controlador NXT.

Puede transferir los programas del ordenador al controlador NXT mediante USB o Bluetooth. Comenzará por conectar al controlador NXT mediante un cable USB. Simplemente conecte un extremo del cable USB al controlador NXT y el otro al puerto USB de la PC. La primera vez que conecte un NXT en el PC, es importante ser paciente ya que el sistema operativo puede tomar hasta unos pocos minutos para instalar los controladores. Una vez que está conectado, necesita decirle a LabVIEW ™ dónde encontrar el controlador NXT. Hacer clic derecho sobre el cuadro de color naranja en la parte inferior izquierda de la ventana, y hacer clic en Buscar NXT. Con ello se abre una ventana similar a la que se muestra en la Figura 37.

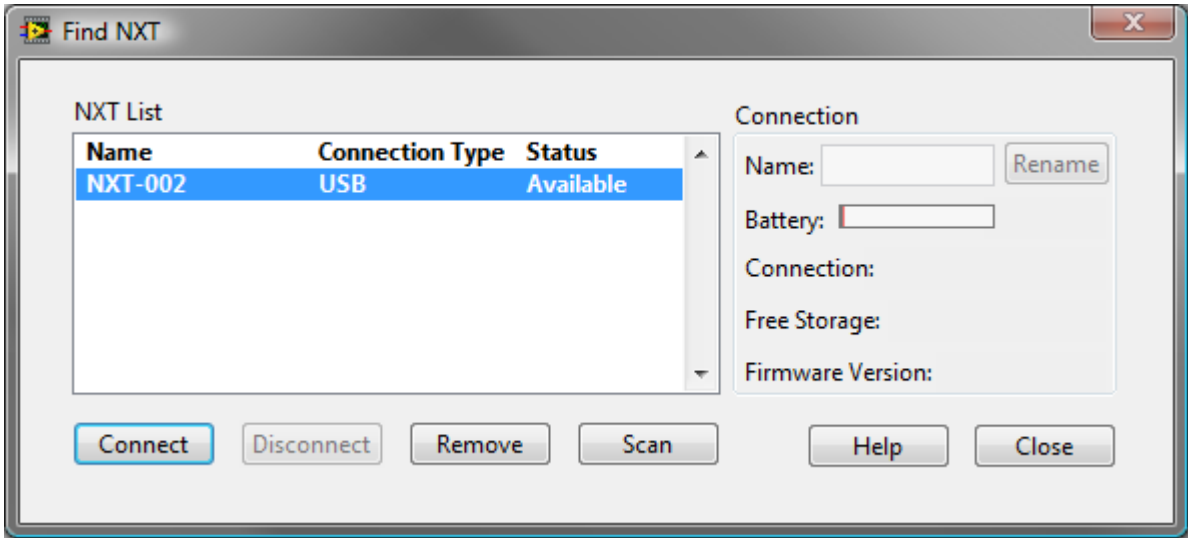



**Figura 37:** Encontrando el controlador NXT.

Dado que el controlador NXT está conectado vía USB, puede ver que está en la lista. Hacer clic una vez para resaltarlo y hacer clic en Conectar para vincularlo con LabVIEW. Verá entonces que la ventana se actualiza para mostrar el nombre, nivel de batería, y otra información sobre el controlador que está conectado. Esto se muestra en la Figura 38.

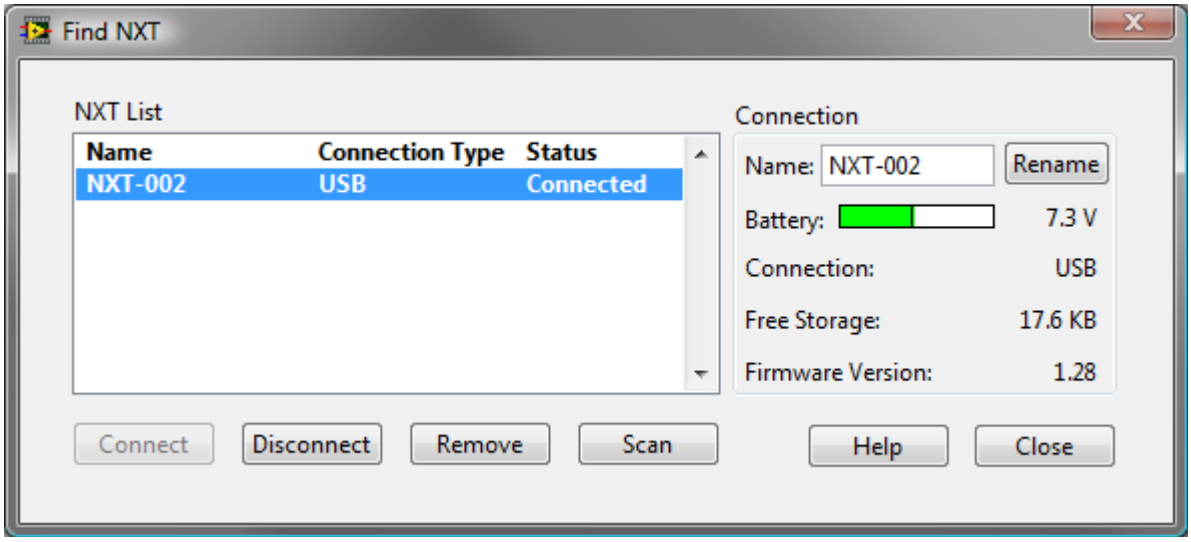



**Figura 38:** Estableciendo conexión con el NXT.

Conectar el controlador NXT al PC a través de Bluetooth es mucho más conveniente que la conexión mediante un cable USB, ya que elimina el uso de cables. Para conectarse al controlador NXT a través de Bluetooth, debe asegurarse que la PC tiene incorporado las capacidades de Bluetooth, o que tiene un adaptador Bluetooth USB instalado. A continuación, debe habilitar el Bluetooth de la PC o el adaptador Bluetooth.

Además, debe asegurarse de que el Bluetooth está activado en el controlador NXT. Para ello, tiene que seleccionar Bluetooth en el menú principal del controlador NXT como se muestra en la Figura 39.


**Figura 39:** Seleccionando Bluetooth del menú principal.

Asegúrese de que Bluetooth está activado, seleccione el icono On / Off y seleccionar On. Una vez que Bluetooth está activado, el NXT lo llevará de vuelta al menú principal donde debe seleccionar Bluetooth. También debe asegurarse de que la visibilidad del NXT está habilitada para que su PC sea capaz de buscar. Para ello, entra en el menú Bluetooth de nuevo, seleccionar Visibilidad y seleccionar Visible.

Ahora que el Bluetooth está activo en el PC y en el controlador NXT, puede conectar con el controlador NXT a través de Bluetooth. Para conectar el controlador NXT, debe seguir un procedimiento similar a cuando se conecta a través de USB. Hacer clic derecho en el rectángulo naranja en la parte inferior izquierda de nuestra pantalla y hacer clic en Buscar NXT. Verá la misma ventana que vio anteriormente (Figura 34). Esta vez, hacer clic en Buscar para buscar dispositivos Bluetooth, y verá que LabVIEW realiza la búsqueda de dispositivos Bluetooth, como se muestra en la Figura 40.


**Figura 40:** Buscando dispositivos Bluetooth.

Después de unos 10-15 segundos, el NXT se encuentra, como se muestra en la Figura 38.


**Figura 41:** NXT encontrado vía Bluetooth.

Hacer clic en el nombre del NXT para resaltarlo y, a continuación, hacer clic en Conectar. Si esta es la primera vez que el NXT se conecta al PC, el programa le pedirá que introduzca una clave de acceso antes de que el NXT pueda ser sincronizado con el PC. Esto se muestra en la Figura 42.


**Figura 42:** Clave de acceso del sistema.

Tiene que introducir "1234" como clave de acceso y seleccionar Aceptar. A continuación, se escucha un pitido y le pedirá que introduzca una clave en el controlador NXT. Una vez más, seleccionar "1234" mediante la navegación con los botones de flecha gris en el controlador y la selección con el botón naranja. Una vez que ha escrito "1234", se puede navegar hasta el icono de la marca de verificación y seleccionar con el botón naranja. Esto se muestra en la figura 43.


**Figura 43:** Introduciendo la clave.

Una vez hecho esto, el sistema operativo se toma unos minutos para formar una conexión con el controlador NXT a través de Bluetooth. Luego tendrá la capacidad para compilar sus programas en el controlador NXT de forma inalámbrica.

Podrá explorar otras opciones del controlador NXT con LabVIEW mediante la selección de las herramientas del menú desplegable, seleccionar Herramientas NXT, y hacer clic en la Terminal del NXT. Esto abre una ventana similar a la que se muestra en la Figura 44.

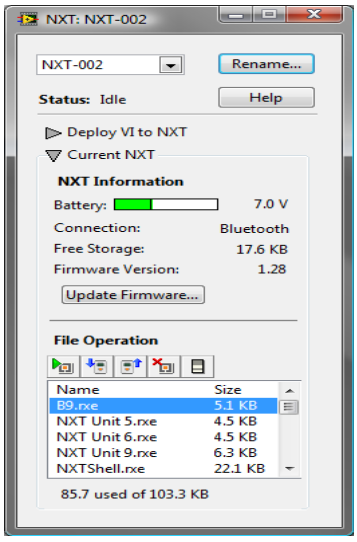



**Figura 44:** Terminal NXT.

Si hay múltiples NXT's conectados, podrá seleccionar el controlador que necesite utilizando el menú desplegable. Esta ventana muestra el nivel de batería del NXT, la cantidad de espacio libre en su memoria, y su versión del firmware. La primera vez que ejecute un controlador NXT con LabVIEW, hay que actualizar el firmware, hacer clic en Actualización de firmware. Esto sólo debe hacerse cuando se conecta a través de USB. En la parte inferior o en la ventana, verá las operaciones de archivo. Aquí podrá ver todos los archivos existentes en el NXT. Es una buena idea para eliminar los programas anteriores de los controladores que ya no utiliza. Todos los archivos de software se encuentran en formato". RXE". Puede hacer clic en el archivo una vez para resaltarlo y luego haga clic en la eliminación de archivo (s) para eliminarlo. Además, puede hacer clic en el botón Desfragmentar para borrar parte de la memoria.

Ahora puede ejecutar programas en el NXT. Si se fija en la barra de herramientas de LabVIEW, verá que hay tres botones para ejecutar un programa, cada uno de los cuales ejecuta el programa de forma ligeramente diferente. Estos tres botones son Ejecutar, Implementar y Depurar. Estos botones se muestran en la Figura 45.

**Figura 45:** Modos de ejecución de programas.

El botón Ejecutar compila el programa en el controlador NXT e inmediatamente se ejecuta. Si hace esto con el código que ha creado para hacer que el robot se mueva hacia adelante, verá que LabVIEW compila el software en el controlador NXT y el robot se moverá hacia adelante.

El botón Implementar compila y descarga el programa en el controlador NXT, pero no funciona hasta que se opte por ejecutarlo con los botones del controlador NXT. Esto es muy útil si quiere compilar el programa a través de USB y no hacer que se ejecute de inmediato, porque si el programa implica el movimiento del robot, el botón de ejecución compila y ejecuta el programa, haciendo que el robot empiece a moverse con el cable USB todavía conectado.

El tercer método para ejecutar un programa es la opción de Depuración. Se a hablará más sobre esto en los últimos capítulos, pero en esencia, la opción de Depuración mantiene el vínculo entre el programa LabVIEW en nuestro PC y el controlador NXT. Podrá usar esto para depurar el programa e interrogar para asegurarse de que funciona correctamente.
Seleccionando el botón Ejecutar o el botón de Implementa, podrá ejecutar el programa y ver que ha hecho que el robot se mueva hacia adelante. Ha creado con éxito su primer programa para el robot LEGO MIDSTORMS.

**Taller:** Hacer que el robot LEGO MINDSTORMS se mueva hacia atrás.

# ¡MANTENGASE EN MOVIMIENTO!

En esta unidad se estudiará la forma de hacer que un programa se repita, pero sobre todo, cómo conseguir que un robot se mueva hacia adelante durante un largo periodo de tiempo, frente a tan sólo un segundo o dos. También se analizará cómo utilizar un ordenador para controlar el robot.

Es importante tener en cuenta que el código en el ejemplo de la lección anterior se ejecuta sólo una vez. Envió comandos al motor para viajar a una velocidad determinada. Tan pronto como los comandos fueron enviados a los motores, el programa se llevó a cabo. Sin embargo, cuando se está programando, encontrará que hay muchos procesos que tienen que ser ejecutados varias veces. La forma en que puede lograr esto es utilizar una estructura de código de LabVIEW ™ llamada ciclo While. Un ciclo While permite que el código dentro de él se ejecute hasta que cierta condición se cumple. Los ciclos While de LabVIEW se muestran en la Figura 46.

**Figura 46:** Ciclo While.

Un ciclo While tiene dos terminales. La de la izquierda se llama terminal de iteración. Se trata de la caja azul con la "i" en el mismo. La segunda es la llamada caja verde de la terminal condicional. Se cablea un valor booleano en la terminal condicional para saber cuándo el programa debe salir. Un valor booleano es un valor verdadero o falso, y dependiendo de la configuración del ciclo While, o bien continuará el ciclo cuando es verdadera, o detendrá el bucle cuando el valor es falso. Para crear el ciclo While, haga clic derecho en el diagrama de bloques, se posiciona sobre el icono de Estructuras y haga clic en el ciclo While. Esto se muestra en la Figura 47.

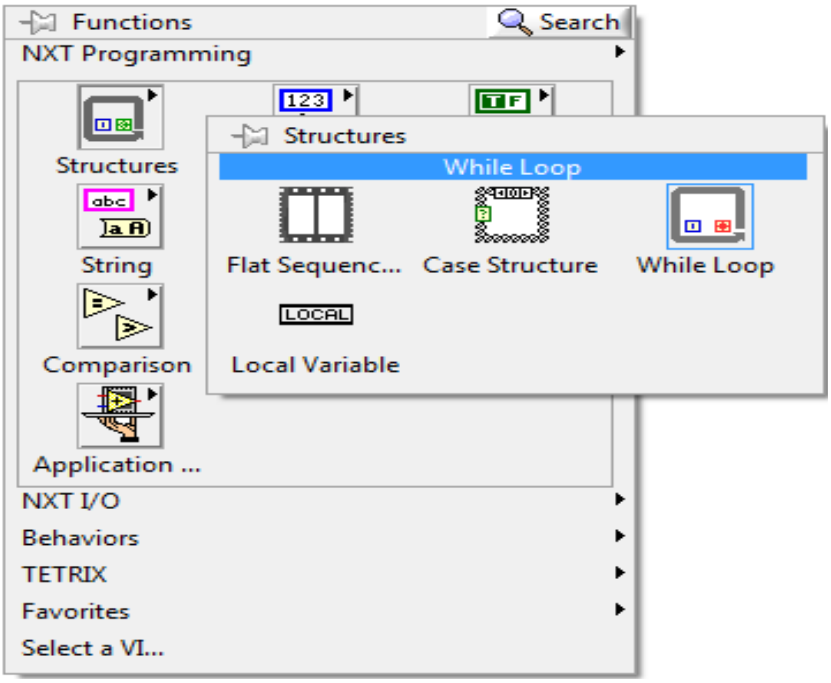



**Figura 47:** Seleccionando un ciclo While.

Se dará cuenta de que el cursor ha cambiado. Una vez que haga clic, podrá empezar a dibujar la forma del ciclo While, como se muestra en la Figura 48.

**Figura 48:** Dibujando el ciclo While.

Si pulsa una vez más, verá que se ha creado la estructura del ciclo While, como se mostró anteriormente en la Figura 46.

Podrá modificar el tamaño del ciclo While colocando el cursor sobre el borde de él, esperando a que las asas aparezcan, a continuación, hacer clic y arrastrarlo de la forma deseada. También puede dibujar el ciclo, alrededor del código ya existente, o podrá colocar el código dentro del ciclo While.
Verá lo que la ayuda de contexto tiene que decir sobre el ciclo While. Puede abrir la ayuda contextual pulsando Ctrl + H, y luego se posiciona sobre el ciclo para obtener una descripción. Esto se muestra en la Figura 49.

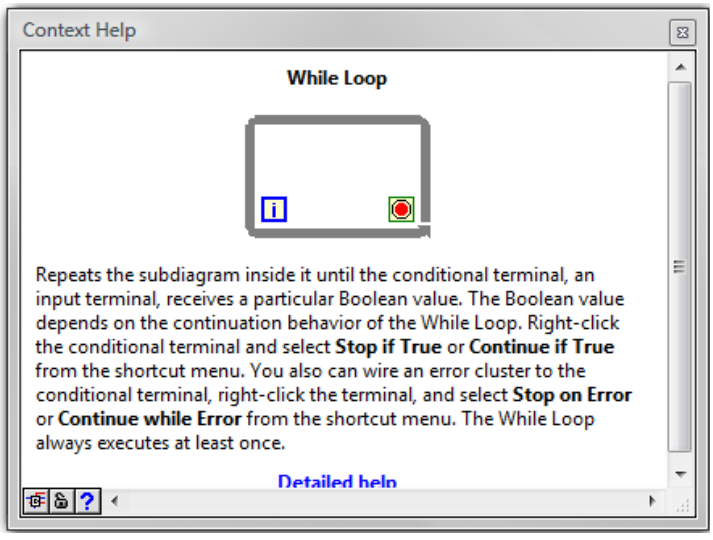



**Figura 49:** Ayuda contextual del ciclo While.

Como se ha dicho antes, la terminal de iteración es la caja azul con la "i" en el ciclo While. La terminal de iteración muestra el número de veces que el ciclo ha acabado, pero tiene un comportamiento inusual, ya que la primera vez que se ejecuta, el

resultado es cero. La primera vez que pasa a través del ciclo, el valor de "i" es 0. La segunda vez, el valor es 1, y así sucesivamente.

El terminal condicional en la parte inferior derecha del ciclo While, se indica como una señal de alto, es un valor lógico. Aquí es donde pone un valor verdadero o un valor falso para continuar o bien ejecutar el ciclo o para detenerlo. Si se posiciona sobre el centro de la terminal, se dará cuenta de que el cursor cambia al cursor de dedo. Si pulsa una vez, el icono dentro de la caja verde cambia. Si pulsa de nuevo, va a cambiar de nuevo a la señal de Stop. Esto cambia el valor lógico a detener el ciclo. Una buena manera de recordar es que la señal de alto significa una parada, por lo que un valor verdadero a una señal de alto significa que se detendrá. Debe tener cuidado ya que dependiendo del modelo elegido para la terminal de condición del ciclo, pone el valor booleano para decidir si desea continuar o detener el ciclo.
Si quiere crear un ciclo que se ejecuta siempre, y está en el modo de señal de alto, deberá introducir un valor falso en el terminal de forma constante. Para crear el ciclo que se ejecutará siempre, debe tener siempre un falso en la terminal de Stop. Para ello, puede crear una constante. Ahora tiene que crear una constante booleana.

Para crear una constante booleana, puede hacer clic en el terminal condicional y haga clic en Crear constante o haga clic derecho en el diagrama de bloques para que aparezca la paleta de funciones, vaya a la sub-paleta booleana y seleccione la constante apropiada. Esto se muestra en la Figura 50.

Quiere que este ciclo se ejecute siempre, así que va a seleccionar la constante falsa y cableará a la terminal de señal de Stop. Esto se muestra en la Figura 51.

La alternativa a esto sería tener una constante verdadera con el icono de siga de la terminal condicional, como se muestra en la Figura 52.

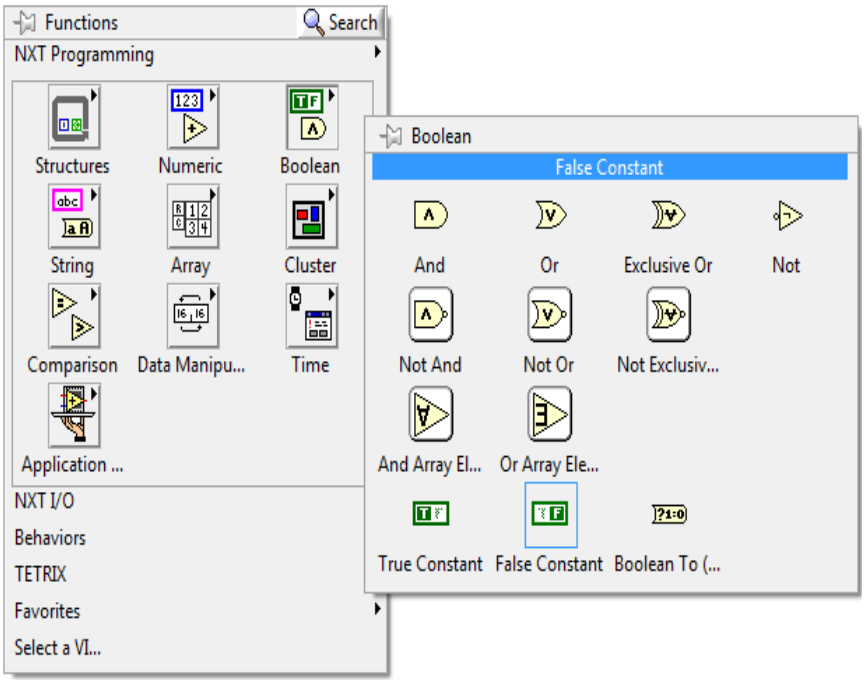



**Figura 50:** Constantes booleanas.

**Figura 51:** Ciclo infinito.

**Figura 52:** Ciclo infinito.

En la unidad anterior, ha programado el robot para que se mueva hacia adelante, utilizando el botón Ejecutar o en el botón Implementar. Compilado el código en el ordenador, y luego enviado al controlador NXT. Sin embargo, se desconecta el controlador NXT del ordenador inmediatamente después de eso. En esta lección, investigará cómo mantener el enlace entre el ordenador y el controlador NXT de forma constante. Para ello, utilizará el botón de Depuración, que es el tercer icono en la barra de herramientas, en el diagrama de bloques o del panel frontal, como se muestra en la Figura 53.

**Figura 53:** Botón de Depuración.

Siempre que se corre el código en modo de depuración, y tiene un control en el panel frontal, como un botón, puede ajustar el valor de ese control y observar el cambio en el robot. Del mismo modo, si tiene un indicador en el panel frontal, puede ver la actualización del indicador en la pantalla de la computadora, a pesar de que el código se está ejecutando en el controlador.

Se recomienda que utilice el modo de Depuración cuando se conecta vía USB, ya que permite la capacidad de respuesta óptima. A pesar de que funciona si se conecta de forma inalámbrica a través de Bluetooth, se encontrará con que a veces, no es tan sensible.

Hay una alternativa al uso del modo de Depuración, que consiste en ejecutar el programa en la instancia principal de la aplicación. Para pasar a la instancia principal de la aplicación de la instancia de selección del NXT, hacer clic derecho en el rectángulo naranja, que se encuentra en la parte inferior izquierda del panel frontal y diagrama de bloques, como se muestra en la Figura 54.

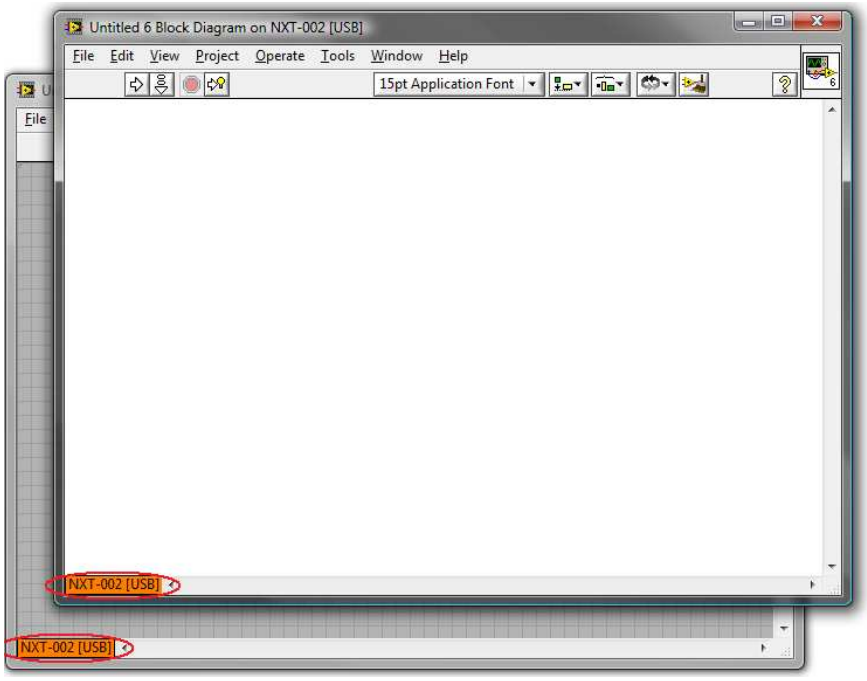



**Figura 54:** Instancia NXT.

Una vez que haga clic derecho en el rectángulo naranja, verá un menú similar al que se muestra en la Figura 55.

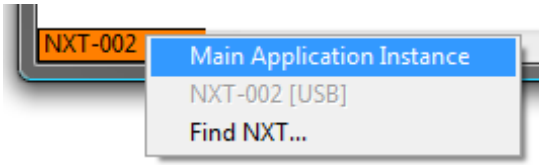



**Figura 55:** Cambiando a la instancia principal.

Cuando se selecciona la instancia de aplicación principal, se dará cuenta de que otro panel frontal y diagrama de bloques aparecerá. Esta es ahora la instancia de la aplicación principal. Si hay algún código que ha creado, mientras estaba en la instancia del controlador NXT, automáticamente aparece aquí en la instancia de la aplicación principal. Si ejecuta el programa aquí en la instancia de la aplicación principal, se dará cuenta de que actúa igual que el modo de Depuración. Mantiene un vínculo entre el ordenador y el controlador, y que son capaces de controlar el robot en tiempo real.

Mediante la instancia de la aplicación principal creará programas para los robots pero no se implementa el programa en el controlador NXT. Anteriormente, cuando ha creado programas dirigidos a la instancia el NXT y cuando hace clic en el botón Ejecutar, se compiló, descargó y ejecutó el programa en el NXT. Esto significa que el programa se guarda en el NXT, y utilizó el poder de procesamiento del controlador NXT para ejecutar el programa. Sin embargo, cuando ejecute el programa en la instancia principal de la aplicación, no es la implementación y la descarga del programa al NXT. Se trata simplemente de utilizar el NXT para controlar los motores y sensores. El programa está utilizando el poder de procesamiento del PC, que es mucho mayor que la potencia de procesamiento de un controlador NXT. Este enfoque puede ser utilizado para ejecutar programas muy grandes y complejos con el NXT.

Debe recordar que siempre hay que escribir el código en la instancia dirigida al controlador NXT, porque esa instancia sólo da acceso a funciones con las que el NXT es compatible. Algunas de las funciones que están disponibles en la instancia de la aplicación principal no pueden trabajar con el NXT cuando se ejecute el programa desde la instancia de NXT. Todas las funciones disponibles en la instancia NXT trabajarán en la instancia de las aplicaciones principales. Para permitir la flexibilidad de usar un programa, tanto en la instancia NXT y la instancia de aplicación principal, no hay que utilizar todas las funciones exclusivas de la instancia de la aplicación principal.

Creará un programa que utiliza el modo de Depuración, y que puede cambiar la velocidad de los motores mientras se está ejecutando. Comenzará por la colocación de dos controles de motor en el diagrama de bloques, como se muestra en la Figura 56.

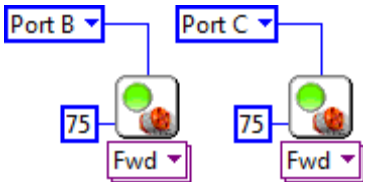



**Figura 56:** Control de motores.

Si quiere introducir la velocidad a la que quiere que los motores giren, coloque un indicador, que tendrá que ir al panel frontal. Hacer clic derecho en el panel frontal y seleccione Control Numérico. Esto se muestra en la Figura 57.

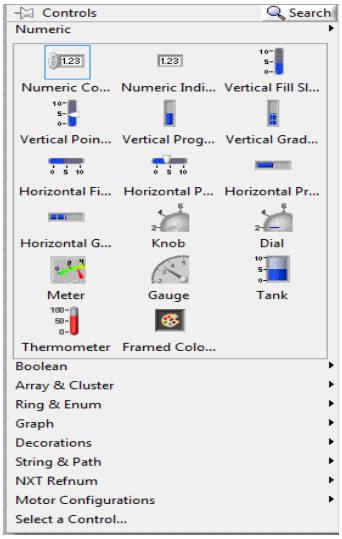



**Figura 57:** Seleccionando un control numérico.

Una vez que coloque el control numérico en el panel frontal, se verá como la Figura 58.

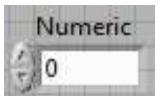



**Figura 58:** Control numérico.

Este control numérico nos permite introducir un número o incrementar / decrementar en saltos de 1. Esto se puede hacer haciendo clic en los botones arriba o abajo situados a la izquierda del campo de número.

Si va al diagrama de bloques, se observa que el indicador numérico se ha aparecido aquí y el código se muestra en la Figura 59.

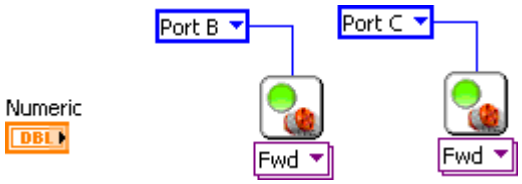



**Figura 59:** Código del diagrama de bloques.

Como quiere que el programa actualice la velocidad de los motores, ya que se cambia desde el control numérico, tiene que poner este código en un ciclo de tiempo, por lo que continuamente se puede actualizar el valor del control. Va a colocar este código dentro de un ciclo While y luego conectar el indicador numérico a los terminales de alimentación de las funciones motoras. El código se muestra en la Figura 60.

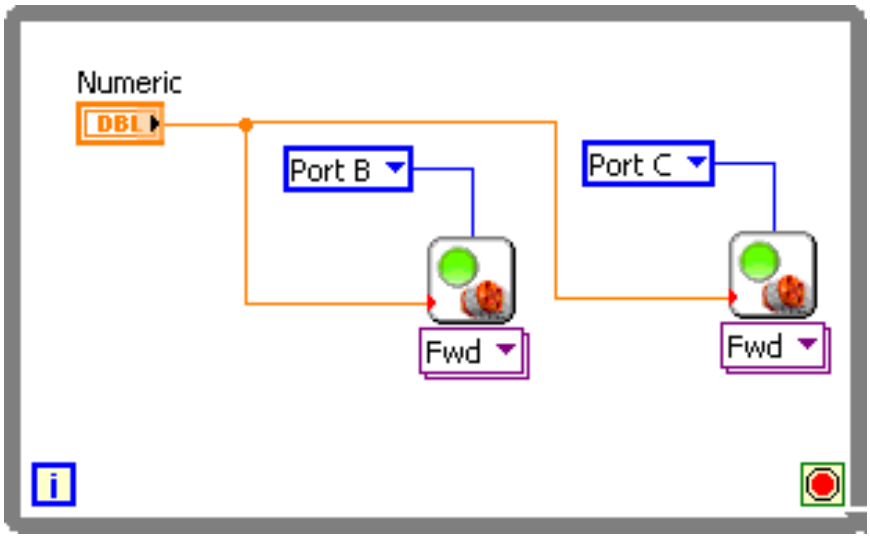



**Figura 60:** Código con el ciclo While.

La última cosa que necesita hacer es que el ciclo While sea infinito. Para ello, es necesario conectar una constante falsa a la terminal condicional, como ha aprendido en páginas anteriores. Después de lo que se hace, el código debe ser similar a la Figura 61 y estará listo para ejecutar su programa.

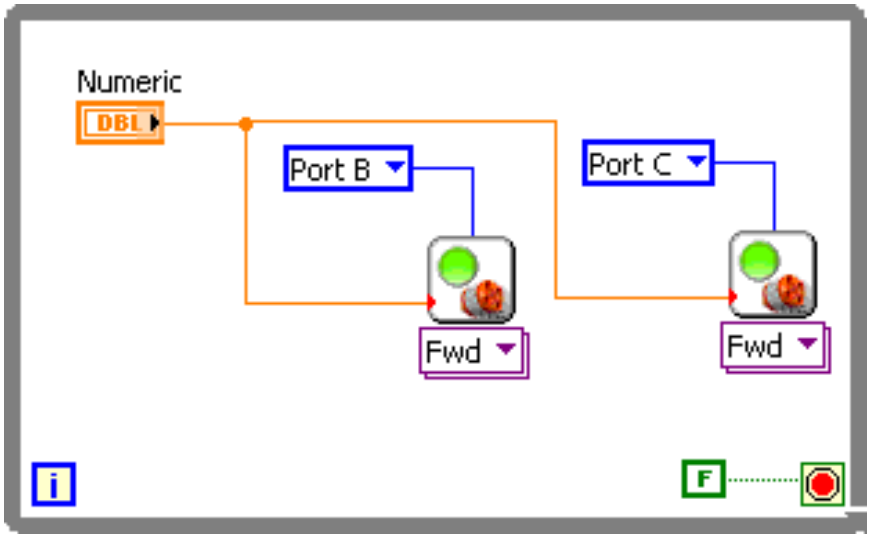



**Figura 61:** Código final.

Ejecutará este programa en modo de Depuración para que haya un vínculo constante entre el ordenador y el NXT. Esto le permitirá cambiar la velocidad del motor a diferentes valores a medida que el programa se está ejecutando.

Conectará el controlador NXT y haga clic en el icono de la barra de herramientas de Depuración. El programa se iniciará, pero los motores no girarán porque el control tiene en la salida un valor de 0. Si escribe un número, como 25 y presiona Enter, se dará cuenta de que la velocidad cambia de inmediato a 25. También puede hacer clic en las flechas arriba y abajo para aumentar o disminuir la velocidad. Para finalizar el programa, debe hacer clic en el botón Cancelar ejecución que se muestra en la Figura 62.

**Figura 62:** Botón cancelar ejecución.

Va a cambiarse a la instancia de la aplicación principal y ejecutar el programa desde allí. Haga clic derecho en el rectángulo naranja en la parte inferior izquierda de la pantalla y seleccione la instancia de aplicación principal. Si hace clic en el botón Ejecutar, se dará cuenta de que cambiará en el modo de ejecución.

**Taller:** Utilice una perilla para controlar la velocidad del robot en tiempo real.

# ¡PARE CON EL SENSOR DE CONTACTO!

En esta unidad, aprenderá a tomar la entrada de los sensores del robot, en particular, el sensor de contacto.

La función de lectura del sensor es utilizada para comunicarse con un sensor conectado al controlador NXT. La función lee el sensor en el robot y devuelve un valor. Puede acceder a la función de leer sensor haciendo clic derecho sobre el diagrama de bloques, seleccionando I / O NXT, y luego haciendo clic en Leer sensor como se muestra en la Figura 63.

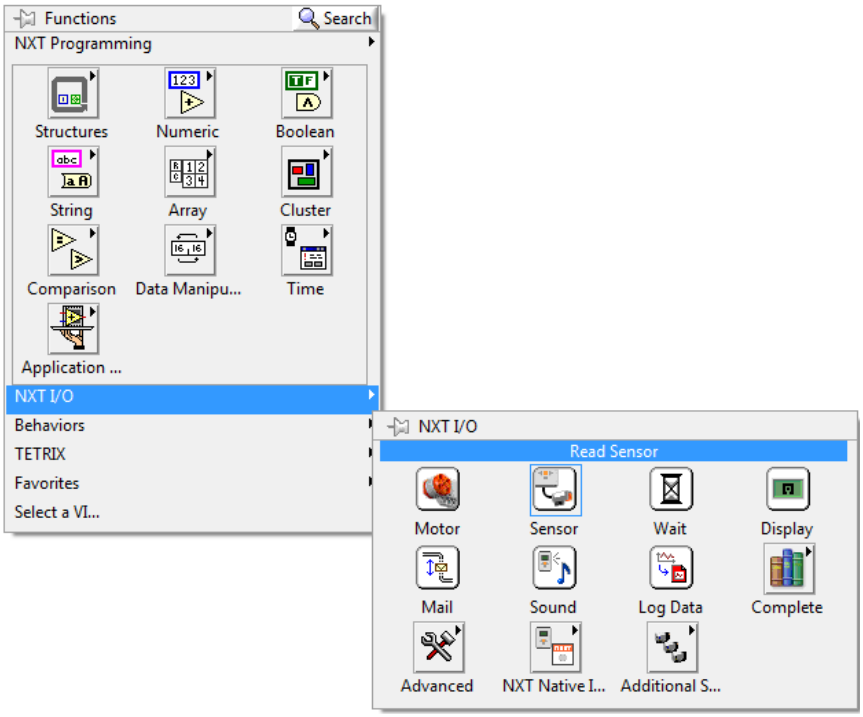



**Figura 63:** Localización de Leer sensor.

Cuando coloca el bloque de Leer sensor en el diagrama de bloques, se dará cuenta de que también tiene un selector polimórfico, lo que nos permite seleccionar el tipo de sensor que quiere leer. En este selector polimórfico se tienen todos los diferentes sensores compatibles con el controlador NXT, como se muestra en la Figura 64.

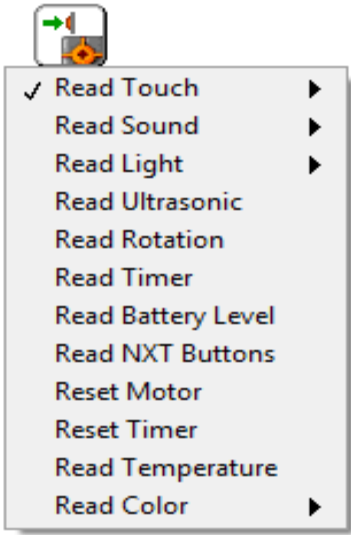



**Figura 64:** Selector polimórfico Leer sensor.

En este caso, ya que quiere comunicarse con el sensor de contacto. Selecciona Leer sensor de contacto. A partir de ahí, tiene tres opciones diferentes, como se muestra en la Figura 65.

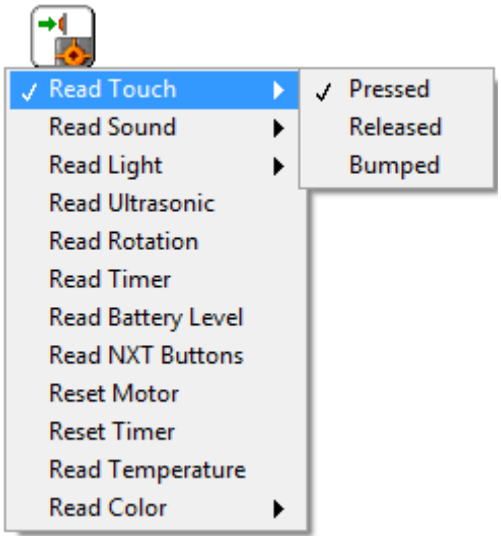



**Figura 65:** Tres opciones del sensor de contacto.

Las tres opciones son: presionado, liberado, y chocado. Cuando se selecciona el modo presionado, el sensor da una salida solamente cuando el sensor está presionado. En el modo liberado, el sensor da una salida cuando el sensor se libera, por lo que en otras palabras, el programa sigue funcionando hasta que el sensor se libera. En el modo chocado, el sensor produce una salida solo cuando al sensor se le dio unos golpecitos.

Al abrir la ayuda contextual, verá lo que dice al respecto de leer un sensor. La ayuda contextual se muestra en la Figura 66.

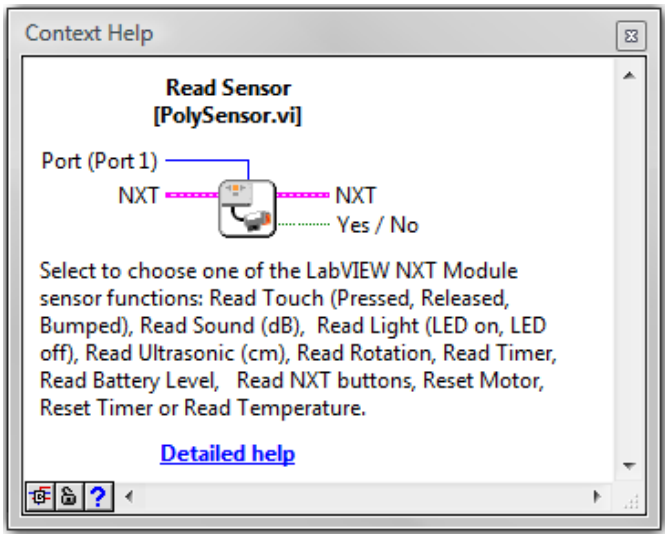



**Figura 66:** Ayuda contextual para los sensores.

La ayuda de contexto muestra que hay dos terminales importantes. Una de ellas es la entrada del puerto, y la otra es la salida booleana.

El color de estos cables indica el tipo de dato que pasa a través de ellos. El cable del puerto es de color azul, lo que significa que está llevando a un valor entero, y la salida es el cable verde que significa que es un valor booleano.

La forma en que funciona la opción Leer sensor de contacto (Presionado) es que se lee el sensor de contacto conectado al puerto de entrada especificado y devolver un Sí o un cierto si el sensor de contacto está presionado, o un No o un falso en caso contrario.

Para seleccionar el puerto en que se conecta el sensor se utiliza un método similar al que se utiliza para especificar el puerto de un motor. Hacer clic derecho en la terminal de entrada y crear una constante. Esto se muestra en la Figura 67.

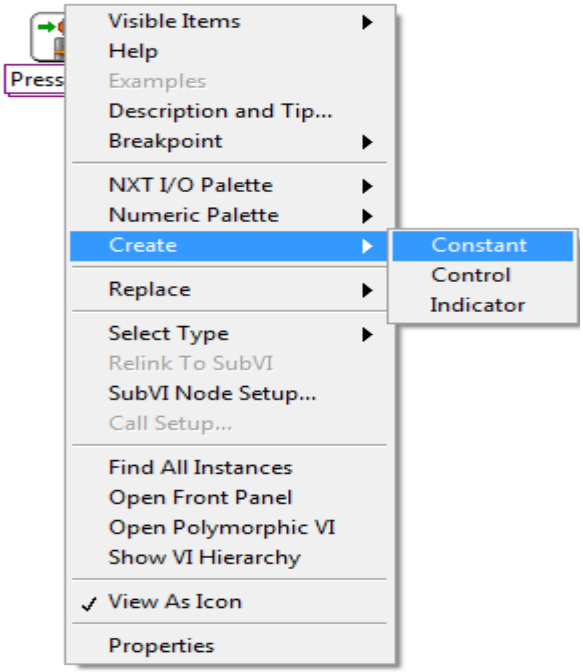



**Figura 67:** Seleccionando el puerto.

Esto crea un selector, que se llama Puerto 1 por defecto. Puede hacer clic en la flecha para abrir el menú desplegable y seleccionar el puerto al que está conectado el sensor de contacto en el NXT. Esto debe verse como la Figura 68.

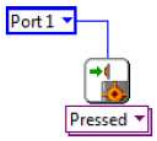



**Figura 68:** Selector de puerto.

El generador de números aleatorios tiene un propósito claro: crear números aleatorios cada vez que se invoca la función. De manera predeterminada, se crea un número aleatorio entre 0 y 100.

Para seleccionar el generador de números aleatorios, haga clic derecho sobre el diagrama de bloques, se posiciona sobre las funciones numéricas y verá el generador de números aleatorios, como se muestra en la Figura 69.

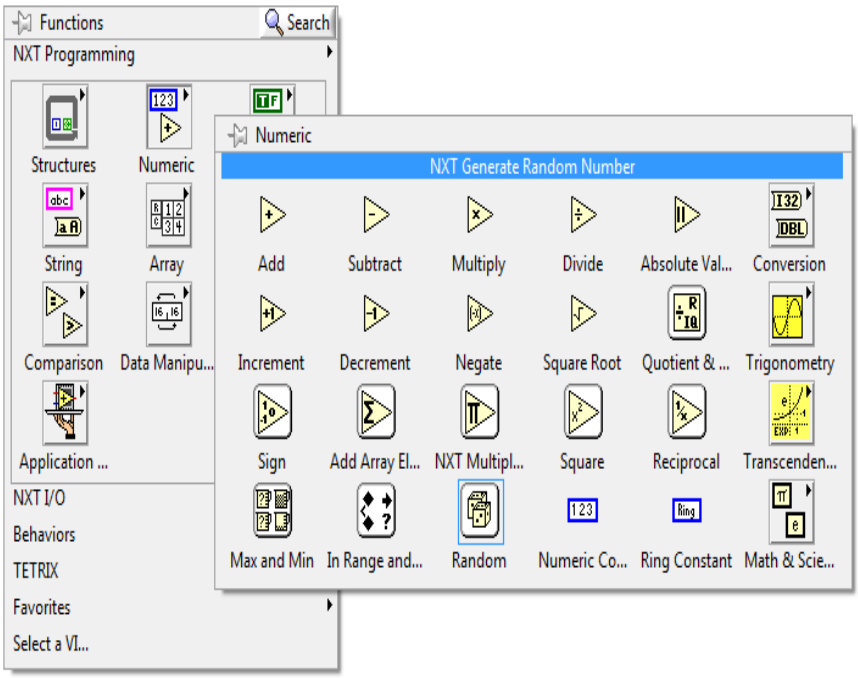



**Figura 69:** Seleccionando el generador de números aleatorios.

Una vez que haya hecho clic en él y lo colocó en el diagrama de bloques, verá un icono similar al que se muestra en la Figura 70.

**Figura 70:** Generador de números aleatorios.

Si se posiciona sobre el generador de números aleatorios, verá la ayuda contextual como se muestra en la Figura 71.

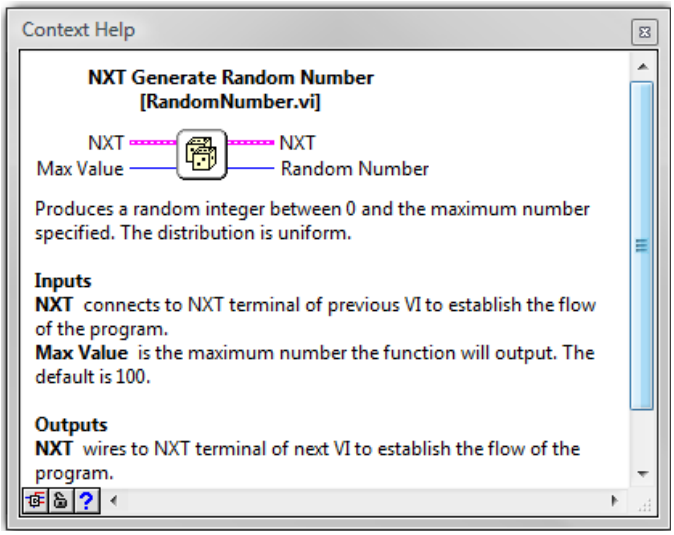



**Figura 71:** Ayuda contextual del generador de números aleatorios.

La ayuda de contexto muestra que hay dos terminales importantes en esta función, el valor máximo y el número aleatorio.

La terminal del valor máximo convenientemente le permitirá cambiar el rango de los números aleatorios que se seleccionan por defecto de 0 – 100 o de 0 – a un número que elija. Para cambiar el valor máximo, simplemente haga clic derecho en la terminal de valor máximo y agregue una constante para el valor máximo.

Para utilizar la función de números aleatorios, es necesario conectar la salida, que es la terminal de números aleatorios y utilizarla en el código, como se muestra en la Figura 72.

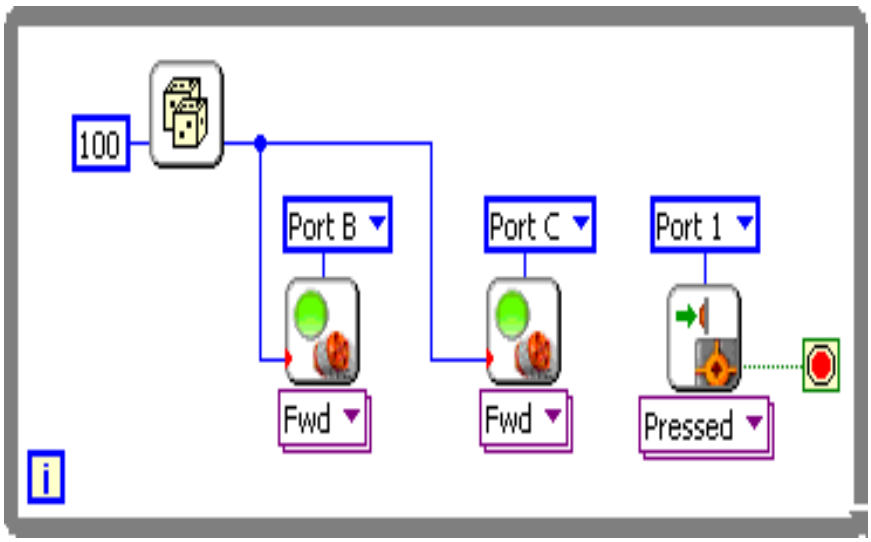



**Figura 72:** Código del programa.

El generador de números aleatorios tiene el inconveniente de que genera los números cada 10 mseg. Por consiguiente el controlador NXT no es capaz de leer esos cambios, por tal motivo es necesario agregar una espera para que el generador demore ese tiempo en enviar el siguiente dato al controlador NXT. Hay una función de LabVIEW que se llama esperar. Espera una cantidad especificada de tiempo antes de dejar que el programa comience o continúe. Para agregar esa función de espera es necesario hacer clic derecho en el diagrama de bloques, entrar al menú I / O NXT y seleccionar *Wait* como se muestra en la Figura 73.

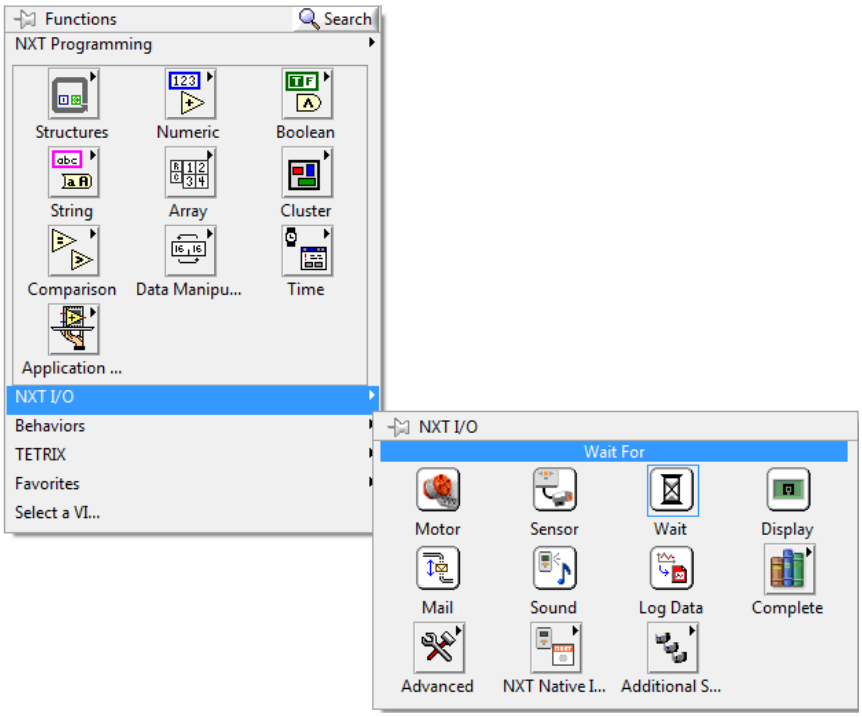



**Figura 73:** Localización de la función esperar.

Cuando coloca la función esperar dentro del ciclo While, verá que es una función polimórfica. Hay muchas opciones aquí, como se muestra en la Figura 74.

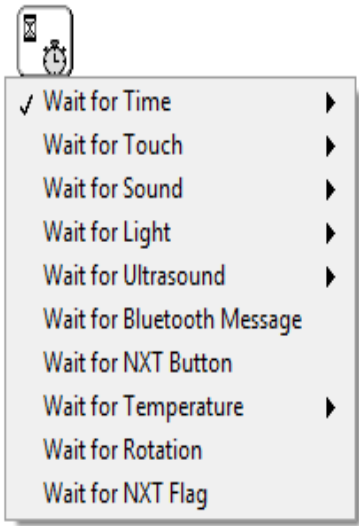



**Figura 74:** Selector polimórfico de la función esperar.

Por default selecciona  esperar un tiempo, y espera un número determinado de segundos. Si necesita ser más preciso, puede seleccionar el modo de milisegundos. Hay también una variedad de funciones de espera de otros sensores como por ejemplo el ultrasónico, el de sonido, el de colores, etc.

Agregue la función de espera dentro del ciclo While, como se muestra en la Figura 75.

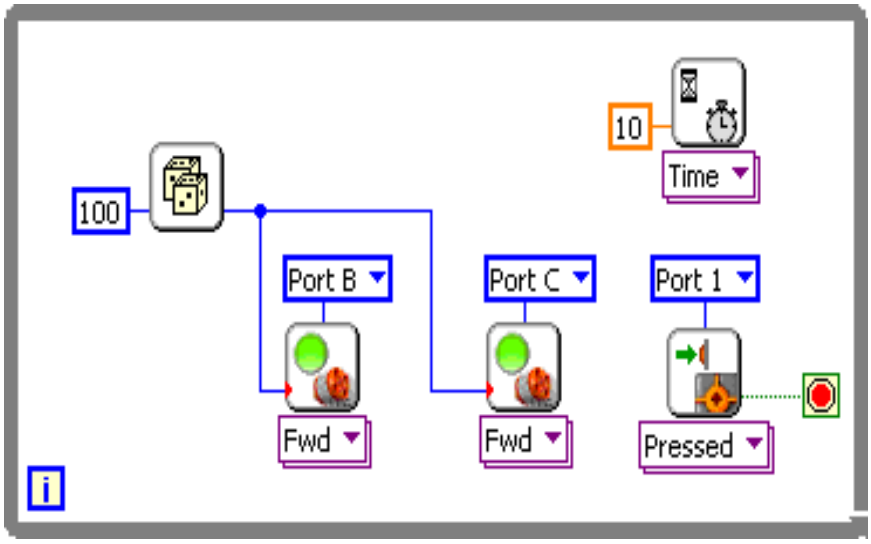



**Figura 75:** Código mejorado del programa.

Si se ejecuta este código, se dará cuenta de que el robot mueve los motores a una velocidad determinada aleatoriamente durante 1 segundo, posteriormente se ejecuta una velocidad distinta. Repite esto hasta que el sensor de contacto es presionado.

**Taller:** Hacer que el robot LEGO MONDSTORMS se mueva hacia atrás a velocidades aleatorias con un tiempo de espera de 25 mseg, hasta que el sensor de contacto este soltado y conectado al puerto 2.

# DIJE ¡PARE!

Se dará cuenta al ejecutar el código de la lección anterior en el robot, que de vez en cuando, cuando el sensor de contacto es presionado, el robot no se detiene. Por ejemplo, si pulsamos el sensor de contacto muy rápidamente, el programa no se detendrá. Hay una explicación simple y una solución simple.

La principal razón para el comportamiento que está viendo es el paralelismo. La forma en que se ejecuta el código de LabVIEW ™ y cómo se decide qué va primero y qué viene a continuación está basado en la conexión de cables entre las entradas, salidas, funciones, controles e indicadores. Si hay cables que conectan dos funciones, hay una secuencia forzada de las operaciones, pero si no hay cables que conectan las funciones individuales, significa que se desarrollará en paralelo. No importa donde se colocan en el diagrama de bloques o cuando se colocaron allí. Lo que importa es si hay cables entre las funciones.
En el código que editó en la Figura 75. Básicamente tiene tres funciones: una es la de mover el motor, otra es la lectura del sensor de contacto, y la ultima es el retraso del programa. No hay cables que conectan las tres funciones, y como resultado, se van a ejecutar en paralelo.

Vamos a analizar este fenómeno con el diagrama de flujo se muestra en la Figura 76.

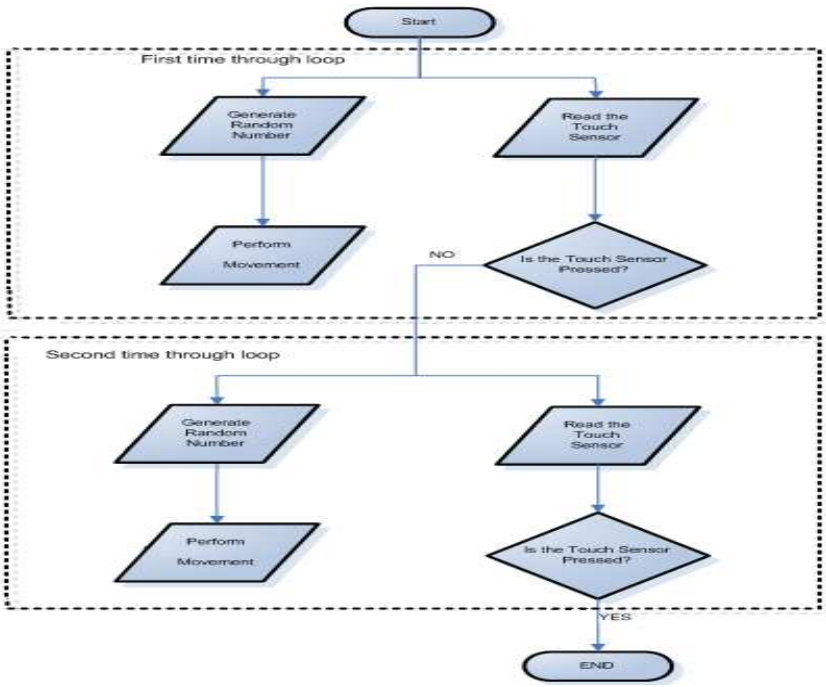



**Figura 76:** Diagrama de flujo del código de la Figura 75.

En primer lugar, un número aleatorio se genera, y un movimiento del motor se realiza. En paralelo con esto, se lee desde el sensor y decide si debe o no parar.

Una forma de definir la secuencia de operaciones que desea que el programa siga es cablear las funciones entre sí. Esto se puede hacer usando las terminales NXT en las funciones. Cada función NXT tendrá dos terminales NXT. Es el cable grueso de color rosa se muestra en la parte superior del bloque de función, como se muestra en la Figura 77.

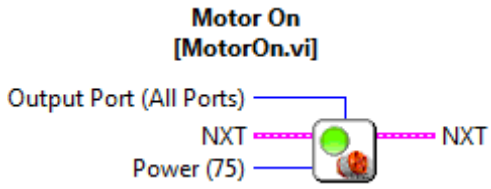



**Figura 77:** Una función con las terminales NXT.

Las terminales NXT le permiten conectar las funciones en conjunto para determinar una secuencia de operaciones. La terminal de la derecha es la salida, y la terminal de la izquierda es la entrada.

La primera función en el programa no tiene un cable conectado a la terminal de la izquierda NXT. Se ejecutará dicha función y luego ejecutará una función relacionada a través del cable en el terminal de la derecha de la primera función. Esto se muestra en la Figura 78.


**Figura 78:** Secuencia de terminales NXT.

El control del motor a la izquierda se ejecutará primero, y luego la segunda función se ejecuta. La segunda es la última función de ejecutar porque no hay cable rosa que sale de su terminal de NXT a la derecha.

Otro concepto interesante con respecto a detener los motores que debe discutir es el freno del motor. Ahora que sabe cómo hacer que el programa inmediatamente después de que el sensor de contacto es presionado se termine, puede observar que el robot todavía tiene tiempo para detenerse. Esto no es debido a la demora de tiempo, o al paralelismo. Esta es causada simplemente por la inercia de los motores y luego llega a una parada. Una forma de corregir esto y hacer que el robot inmediatamente

ponga fin al programa, es utilizar la función de freno en la función de control de motores. Utilizará el selector polimórfico para seleccionar Motor Off y haga clic en freno, como se muestra en la Figura 79.


Figura 79. Freno de motor.

Simplemente, esta función determina la condición de finalización del ciclo durante el uso de los cables NXT de color rosa. Esto hará que el robot llegue a un cese inmediato tan pronto como el bucle While termina. Esto se muestra en la Figura 80.

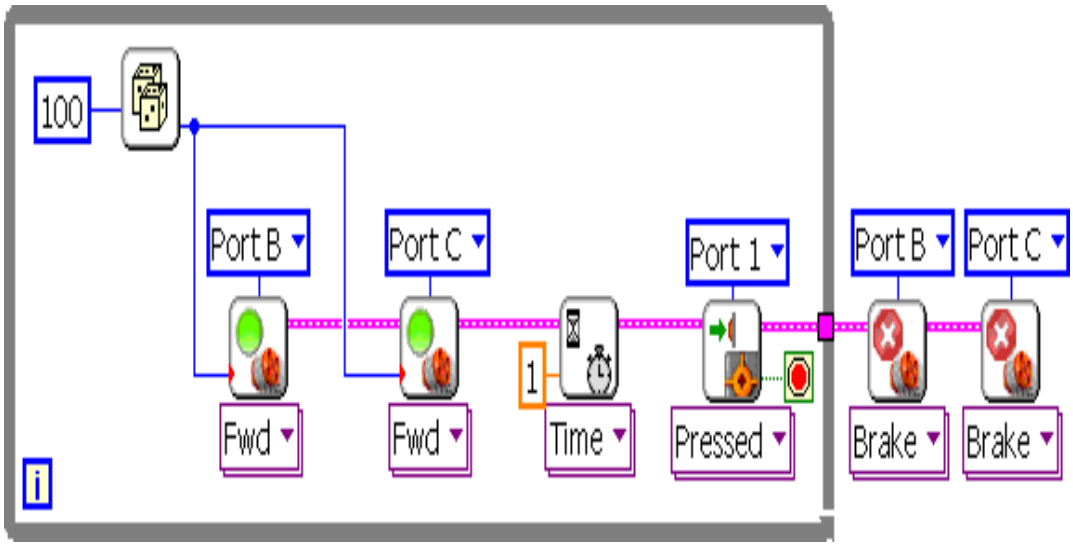



Figura 80: Freno de motor al final del programa.

**Taller:** Arreglar el código para que la unidad NXT del robot avance hacia adelante a velocidades al azar hasta que el sensor de contacto es presionado.

# ¡CUIDADO!

Esta lección aprenderá cómo utilizar las estructuras de caso. La estructura de casos en LabVIEW ™ es muy similar a la declaración if / else que encuentra en muchos otros lenguajes de programación. Una estructura de casos le permite hacer una cosa en una situación, y algo diferente en otra situación. Un ejemplo de esto es que si el robot se encuentra cerca de una pared, entonces se detenga. Si el robot no está cerca de una pared, entonces que siga adelante.

Al igual que en el ciclo While, la estructura del caso se encuentra en las estructuras. Otra similitud con el ciclo While es que se utiliza para contener el código. Lo que está dentro de la estructura de casos estará a cargo de uno u otro caso. La estructura del caso se muestra en la Figura 81.

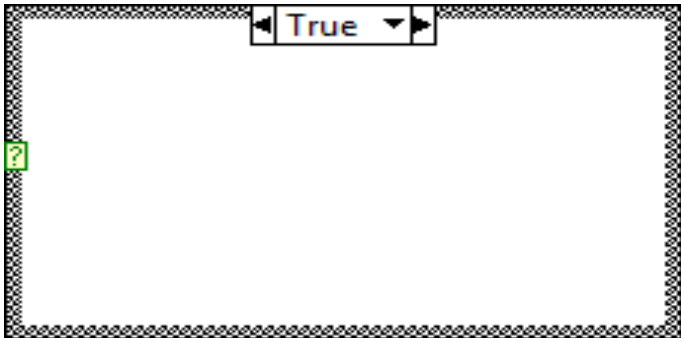



**Figura 81:** Estructura de casos.

Las características principales de la estructura de casos son la etiqueta del selector en la parte superior, que muestra en que caso se encuentra, y el selector de caso sobre la izquierda, que es un signo de interrogación en él.

Las flechas de la etiqueta del selector se pueden utilizar para alternar entre los diferentes casos o se puede utilizar la flecha desplegable junto al nombre del caso. Esto se muestra en la Figura 82.

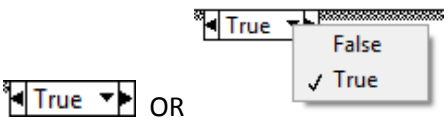



**Figura 82:** Alternar entre los casos.

Por defecto, en una estructura de casos se espera una entrada booleana. Por lo tanto, existen dos casos: un caso real y un caso falso. También puede notar que el selector

de caso es de color verde, lo que significa que se espera un valor booleano. Sin embargo, las estructuras de caso pueden ser utilizadas para tomar valores numéricos o de tipo cadena. Por ahora, se limitara a tratar con valores booleanos de entrada a la estructura de caso.

Si ponemos algo de código dentro de nuestro caso falso, como un sensor, que sólo existe en el caso falso y luego alternamos con el caso real, veremos que no está allí. Esto se muestra en la Figura 83.

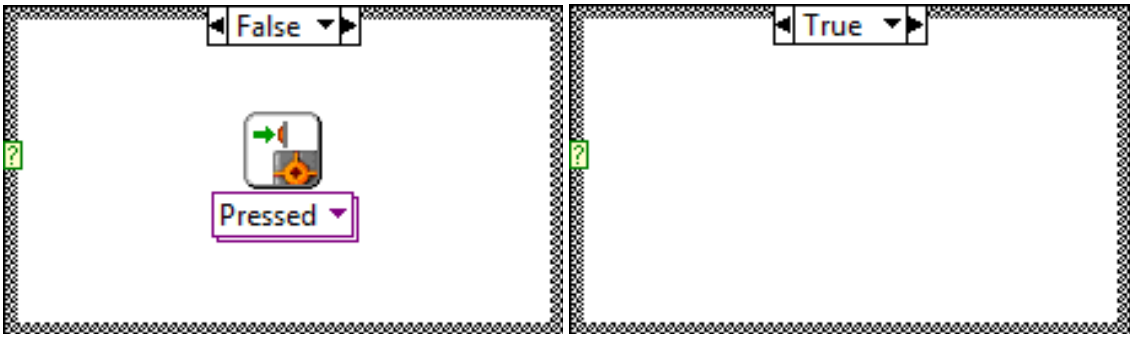



**Figura 83:** Casos falso y verdadero.

La razón por la que no ve las mismas cosas en el caso verdadero y el caso falso es porque solo ha colocado objetos en uno de los dos casos.

Para ayudarle a entender la forma en que trabajan las estructuras de caso, va a crear un programa que, en un caso, suma dos números, y en el otro caso, los resta. Este programa se crea en la instancia principal de la aplicación, por lo que es importante cambiar a esa instancia antes de ejecutar el programa. Sin embargo, puede crear el programa mientras esta en la instancia NXT, porque se ha acostumbrado a la ubicación de todas las funciones de las paletas.

Va a comenzar por la colocación de una estructura de caso en el diagrama de bloques. Tiene un caso real y un caso falso, y puede decidir sumar los dos números en el caso real, y restarlos en el caso falso. Necesita una manera de indicar si está en el modo de sumar o en el modo de restar. Para ello, se dirige al panel frontal y crea un conmutador. De clic con el botón derecho del ratón en el panel frontal, seleccione la paleta *Express >> Buttons & Switches*, y haga clic en *Vertical Toggle Switch*. Esto se muestra en la Figura 84.

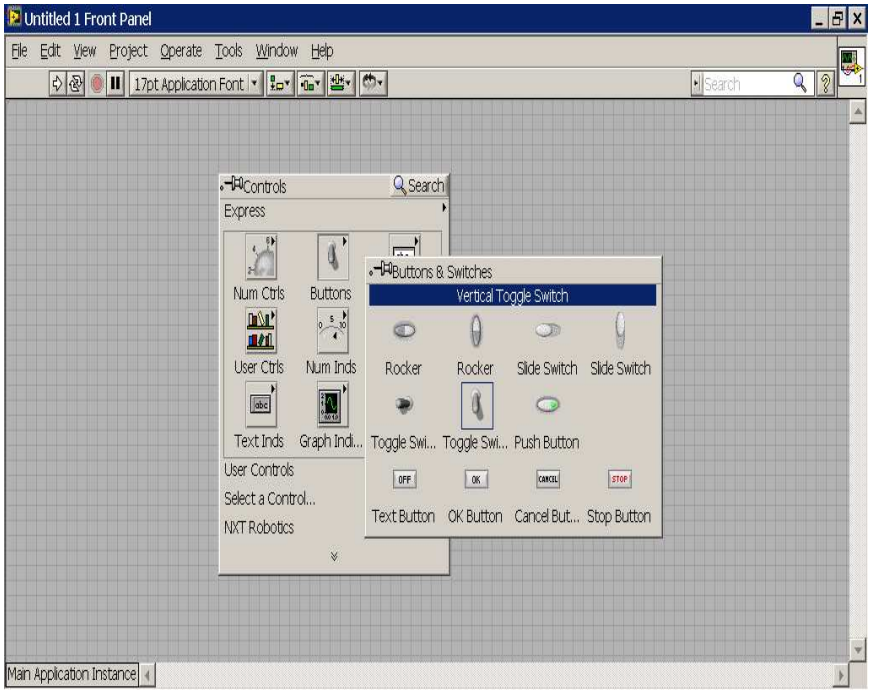



**Figura 84:** Localización del Vertical Toogle Switch.

Después de que coloque el interruptor en el panel frontal, debe verse como la Figura 85.


**Figura 85:** Vertical Toogle Switch.

Puede cambiar el tamaño de este interruptor por lo que es más fácil de usar. También de clic derecho sobre el interruptor, seleccione Visible Items y seleccione Label. Haga clic en el texto Boolean. Esto crea una etiqueta que diga si el interruptor de palanca es la salida de un valor verdadero o un valor falso. Cuando está en la posición hacia abajo, la etiqueta dice "Restar", lo que significa que el interruptor de salida es un valor falso. Cuando se pulsa el interruptor cuando el cursor se convierte en un puntero, y el interruptor se destina a la posición con la etiqueta que diga "Sumar". Esta es la posición en la que el interruptor de salida arroja un valor verdadero.

Si cambiamos el diagrama de bloques, verá que el interruptor ha aparecido. Va a cablear el interruptor a la selección de casos de la estructura de caso. Nuestro código debe ser similar a la Figura 86.

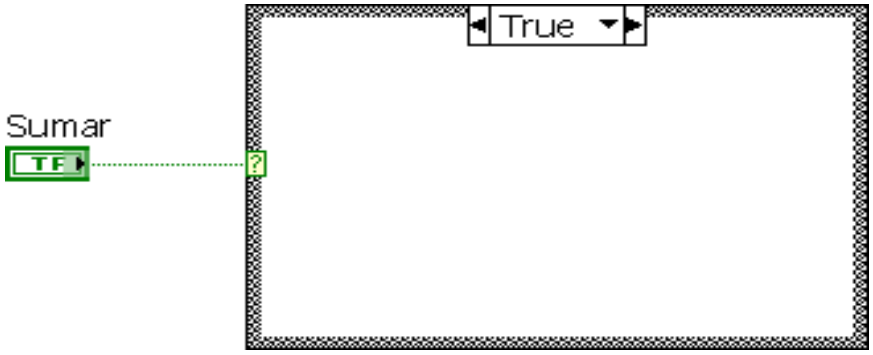



**Figura 86:** Codigo de la estructura de caso.

Dado que quiere sumar en el caso real, se colocará una función Sumar dentro del caso Verdadero. Estará restando en el caso falso, por lo que va a colocar una función Restar dentro del caso Falso. Las funciones de Restar y Sumar se encuentran en la sub-paleta numérica que se encuentra dando clic derecho sobre el diagrama de bloques en Programming >> Numeric. Su código debe ser similar a la Figura 87.

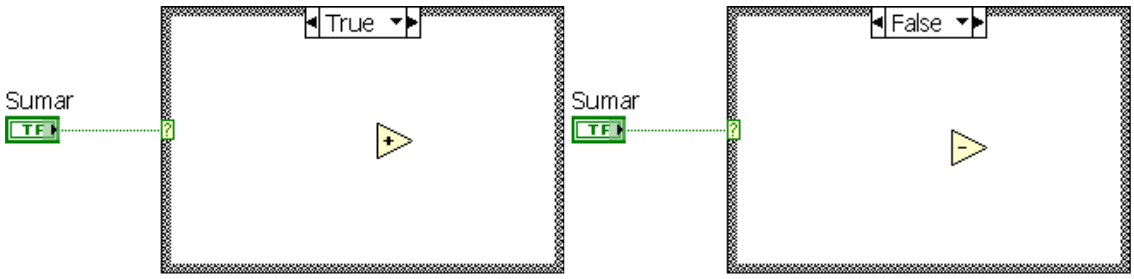



**Figura 87:** Código en los casos verdadero y falso.

Tiene que volver al panel frontal y crear dos controles numéricos para que pueda introducir los dos valores que se van a sumar / restar. También tiene un indicador numérico para mostrar el resultado. Estos controles numéricos e indicadores se pueden acceder desde la sub-paleta numérica en la paleta de controles. El panel frontal debe ser similar al de la Figura 88.

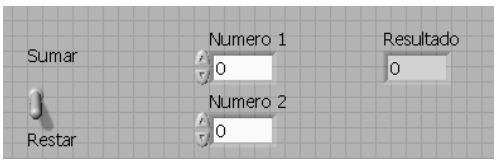



**Figura 88:** Panel frontal del programa.

Si vuelve al diagrama de bloques, tiene que poner los dos controles y el indicador en el lugar correcto. Va a colocarlos como se muestra en la Figura 89.

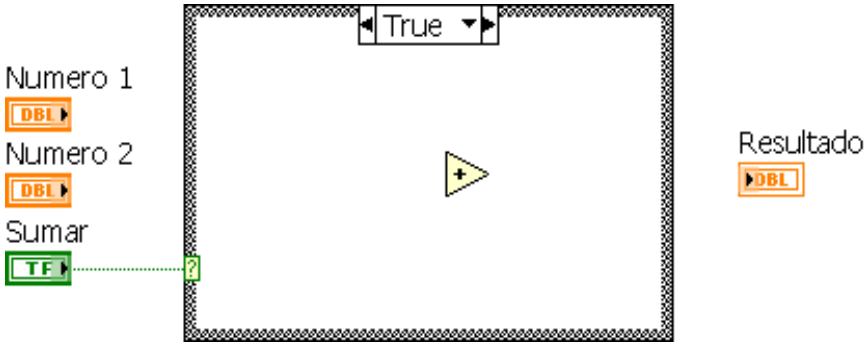



**Figura 89:** Disposición del código.

Ahora tiene que conectar los dos controles a las entradas de la función Sumar, y tiene que conectar la salida de la función Sumar al indicador. Esto se muestra en la Figura 90.

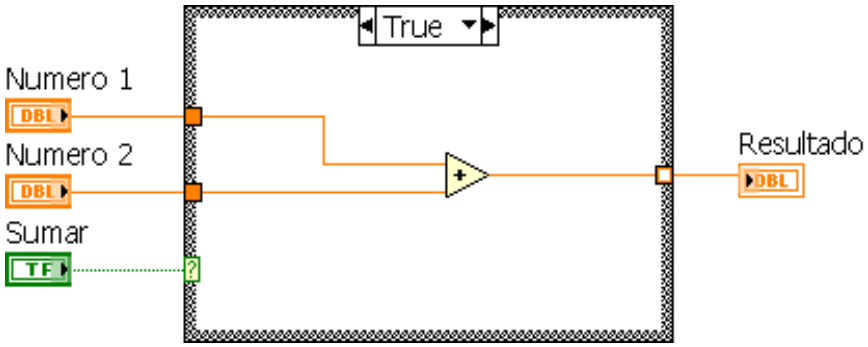



**Figura 90:** Cableado de entradas y salida.

Ahora tiene que cablear el caso falso; los dos controles de las entradas a la función Restar, y el cable de salida de la función Restar al indicador. El código debe ser similar a la Figura 91.

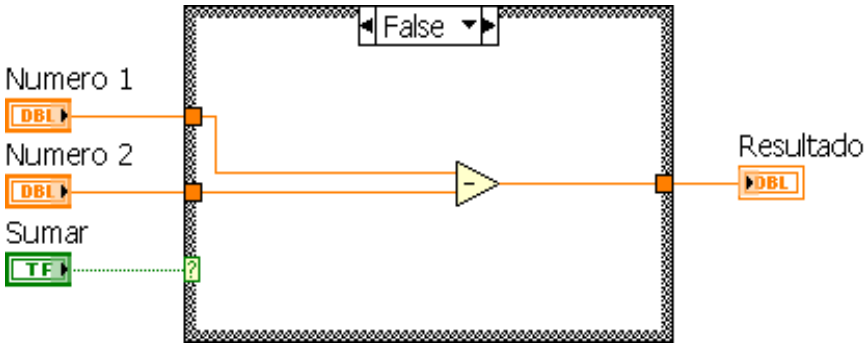



**Figura 91:** Caso Falso.

Si se dirige al panel frontal, puede introducir dos valores en los controles. Va a usar 7 y 2 y poner el interruptor en el modo Sumar. Ahora bien, si hace clic en el botón Ejecutar, el indicador mostrará el resultado de 9, como se muestra en la Figura 92.

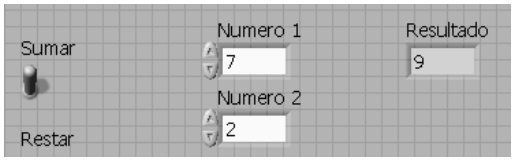



**Figura 92:** Ejecución del programa.

Ahora va a utilizar este programa para restar. Pulse el interruptor y póngalo en el modo de Restar. Puede realizar ejecutar el programa y ver que se calcula 7 - 2, y muestra el resultado, como se muestra en la Figura 93.

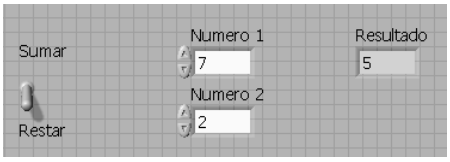



**Figura 93:** Ejecución del programa.

Ha creado con éxito un programa que funciona de manera diferente, dependiendo del valor booleano ingresado a la etiqueta del selector de la estructura de caso.

Hay muchos diferentes sensores que se pueden utilizar con el LEGO ® MINDSTORMS ® robot y el controlador NXT. Esta el sensor de contacto, el sensor de sonido, el sensor de temperatura, el sensor de ultrasonidos, y muchos más. Va a explorar el sensor ultrasónico con mayor detalle en esta lección. Este sensor se muestra en la Figura 94.

**Figura 94:** Sensor ultrasonico.

El sensor de contacto que previamente aprendió proporciona un valor booleano a la salida, pero el sensor ultrasónico genera un valor numérico. La salida del sensor ultrasónico es la distancia hasta un objeto. Si está muy cerca de un objeto, la distancia puede ser cero o un número pequeño. Si está muy lejos de un objeto, el número será grande, de hasta un máximo de 255. Es importante señalar que los datos procedentes del sensor es un valor entero de la distancia. Es por eso que hay un cable azul que sale de la terminal de salida, como se muestra en la figura 95.

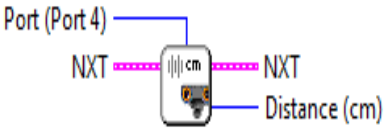



**Figura 95:** Función del sensor ultrasonico.

Como todas las otras funciones de NXT, hay dos terminales NXT y debemos especificar a qué puerto está conectado en el controlador NXT.

Puede acceder al sensor ultrasónico haciendo clic derecho sobre el diagrama de bloques, seleccionando NXT I / O y haciendo clic en el sensor. Con ello se abre la función del sensor como se muestra en la Figura 96.


**Figura 96:** Sensor por default.

Debe hacer clic en la flecha desplegable para acceder a la selección de polimórficos y haga clic en Leer ultrasónico. Esto se muestra en la Figura 97.

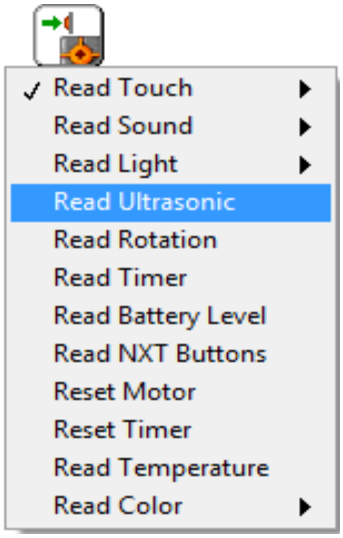



**Figura 97:** Seleccionando sensor ultrasónico.

Como siempre, debe hacer clic derecho en la terminal del puerto para crear una constante y seleccionar el puerto correcto.

Es muy poco probable que quiera hacer que el robot haga algo cuando haya una distancia exacta hasta un objeto. En su lugar, va a querer que el robot haga algo si es menor o mayor que una cierta distancia hasta ese objeto. Por ejemplo, querrá que el robot deje de moverse cuando se detecta un objeto a menos de 20 centímetros de distancia, pero sigue conduciendo cuando no hay nada delante de él.

Dado que el sensor ultrasónico devuelve un número único, que es la distancia de un objeto delante de él, tiene que determinar un umbral. Un umbral es básicamente un valor numérico con el que va a comparar la salida del sensor ultrasónico. Querrá que el

robot haga algo si el valor del sensor ultrasónico tiene una salida que está por debajo de ese valor umbral, y hacer algo más si es igual o superior a ese valor umbral. Una función de comparación, como el menor que o el mayor que, puede hacer eso. Esta función ayuda a convertir la salida numérica del sensor ultrasónico en un valor booleano que puede conectar a una estructura de caso.

Es importante señalar que las funciones de comparación toman dos valores numéricos: o enteros de doble precisión, enteros, etc, y booleanos. Esto permite formular la pregunta como "¿Estoy cerca de la pared?" Y obtener una respuesta afirmativa o no. La ubicación de las funciones de comparación se puede encontrar haciendo clic derecho en el diagrama de bloques, y seleccionando de la sub-paleta de comparación. En este caso seleccionará la función Menor que, como se muestra en la Figura 98.

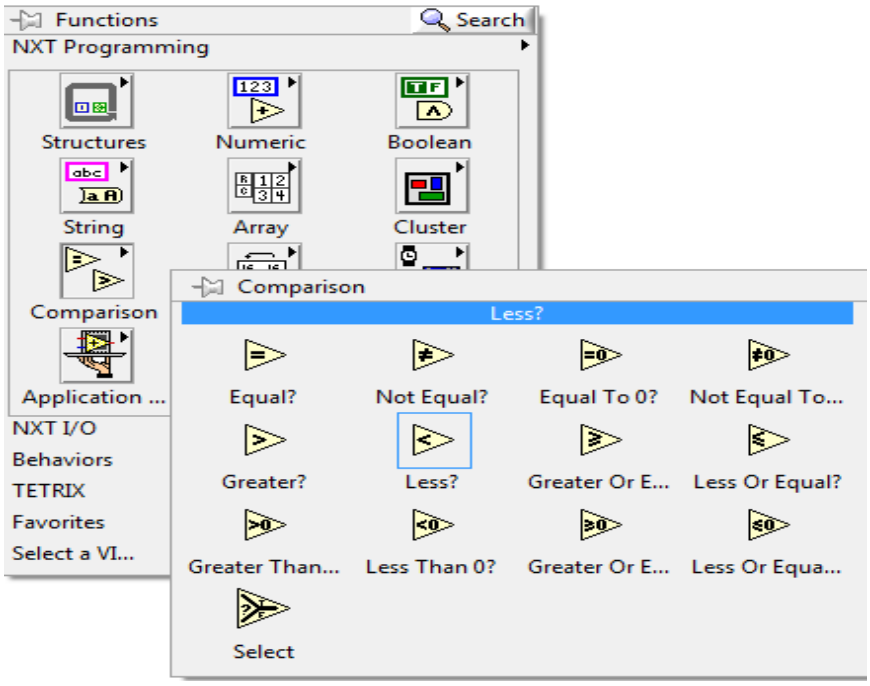



**Figura 98:** Localización de la Función Menor que.

Ahora puede colocar la función Menor que, cerca del sensor ultrasónico y cablear la terminal de salida del sensor a la terminal de entrada de la función. Esto se muestra en la Figura 99.

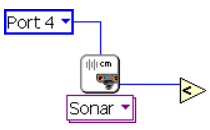



**Figura 99:** Función menor que con el sensor ultrasonico.

En la otra entrada de la función Menor que, puede crear una constante para determinar la gama. Esto se muestra en la Figura 100.

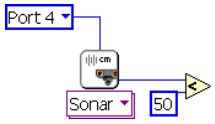



**Figura 100:** Función menor que con una constante.

El valor que asigna a la constante depende de la situación. Revisando el ejemplo cuando el robot se acerca a una pared, este número determinará qué hacer cuando el robot está a una distancia inferior a 50 cm y qué hacer cuando el robot está a una distancia superior a 50 cm.

Ahora puede conectar la salida de la función Menor que en la selección de la estructura de caso. Esto se muestra en la figura 101.

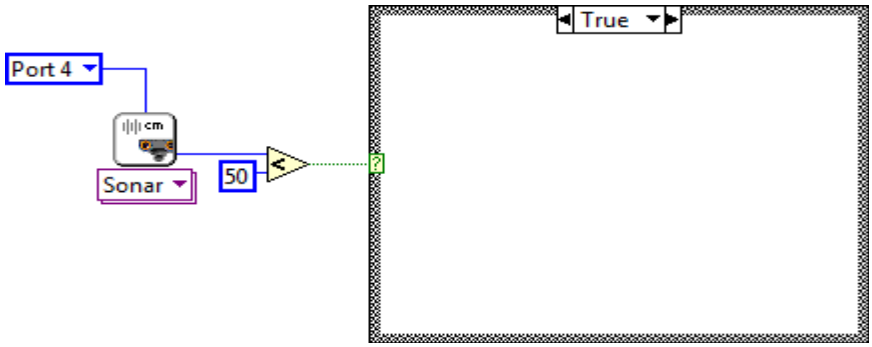



**Figura 101:** El sensor ultrasónico es utilizado para decidir cuál caso se ejecuta.

Ahora, el código del caso "verdadero" se ejecutará si el valor de comparación devuelve un “true”. Si el valor de comparación devuelve un “false”, el caso "falso" se ejecutará.

**Taller:** Hacer que el robot se mueva hacia adelante hasta que el sensor ultrasónico detecte un objeto enfrente del robot a una distancia de 20 cm., y entonces se detenga. Si el objeto es eliminado, el robot sigue avanzando. Al pulsar el sensor de contacto se detiene el programa.

# ¿HASTA DÓNDE?

El propósito del modo de depuración, como se ha visto antes, es mantener un vínculo entre el ordenador y el controlador NXT, mientras que el programa se está ejecutando. Esto también se puede hacer si el programa se ejecuta desde la instancia de aplicación principal.

La ventaja de mantener un vínculo entre el ordenador y el controlador NXT es que le permite cambiar los valores de los controles mientras se ejecuta el programa. También le permite monitorear lo que está pasando con el programa  para depurarlo. La forma de hacerlo es mediante la creación de indicadores.

Los indicadores siempre aparecen en el panel frontal y en el diagrama de bloques, como se ve en la Figura 102.

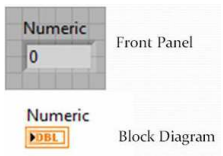



**Figura 102:** Indicador en el panel frontal y en el diagrama de bloques.

Es importante recordar que los indicadores sólo funcionarán en modo de depuración en la programación de nuestro controlador NXT.

Un tipo muy útil de indicador es un gráfico. Los gráficos toman un solo punto de datos a la vez, y recuerda una historia de valores anteriores.

Los gráficos se pueden acceder haciendo clic derecho sobre el panel frontal, seleccionando el gráfico y haciendo clic en el gráfico de forma de onda. Luego vera el diagrama como se muestra en la Figura 103.

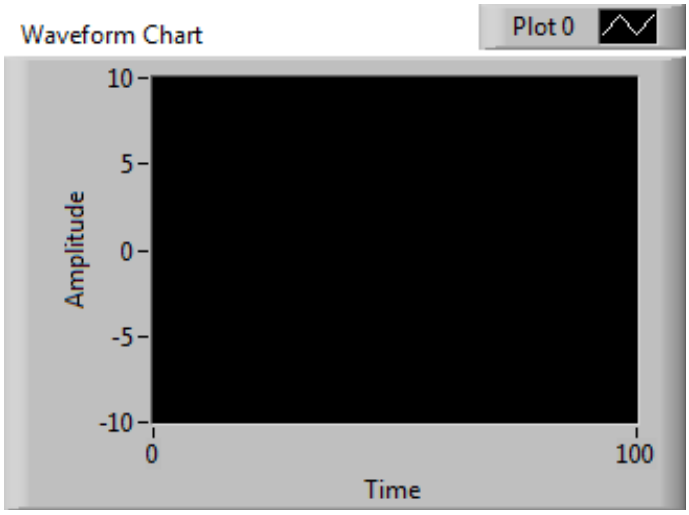



**Figura 103:** Gráfico de forma de onda.

Si da clic con el botón derecho sobre el gráfico, verá que hay opciones para cambiar muchas cosas con respecto al gráfico, como se ve en la Figura 104.

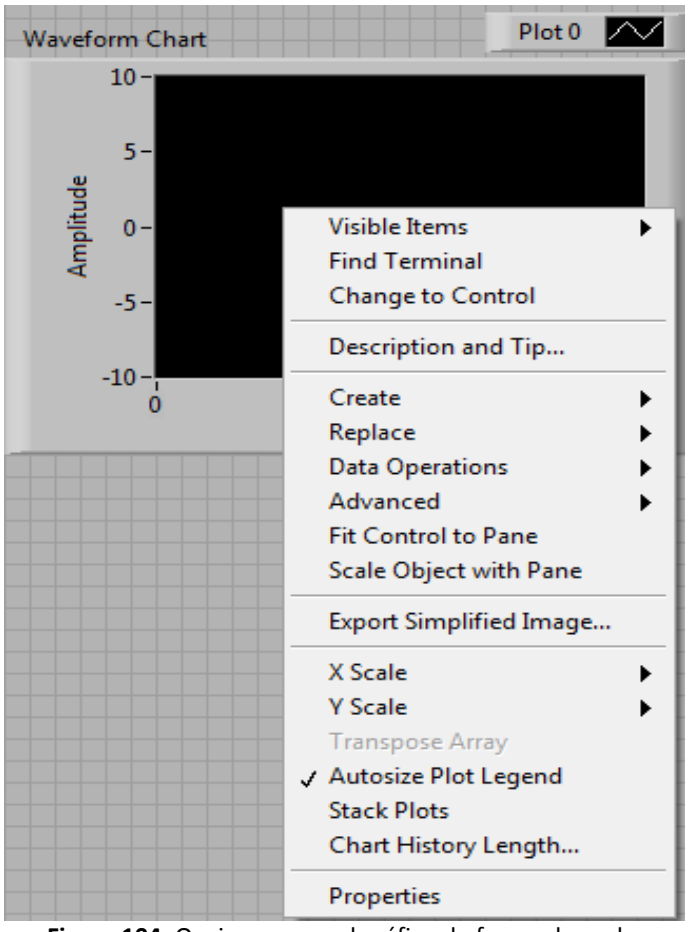



**Figura 104:** Opciones para el gráfico de forma de onda.

En este menú, puede cambiar el color, las escalas, los estilos, el formato y mucho más.

**Taller:** Hacer un programa que lea la distancia desde el sensor ultrasónico utilizando una gráfico mientras que el código del taller anterior se está ejecutando. Al mismo tiempo que permita modificar el umbral para la comparación, mientras que el programa se está ejecutando.

# ¿PASO ADELANTE O PASO ATRÁS?

Esta unidad le enseñará cómo utilizar la memoria de bucle, investigando el uso de registros de desplazamiento. Los registros de desplazamiento se unen a los bucles. Almacenan o "recuerdan" los datos de una iteración y los ponen a disposición en la siguiente iteración. Es algo así como una variable en lenguajes basados en texto.

Para crear un registro de desplazamiento, haga clic derecho en el borde del ciclo While, como se muestra en la figura 105.

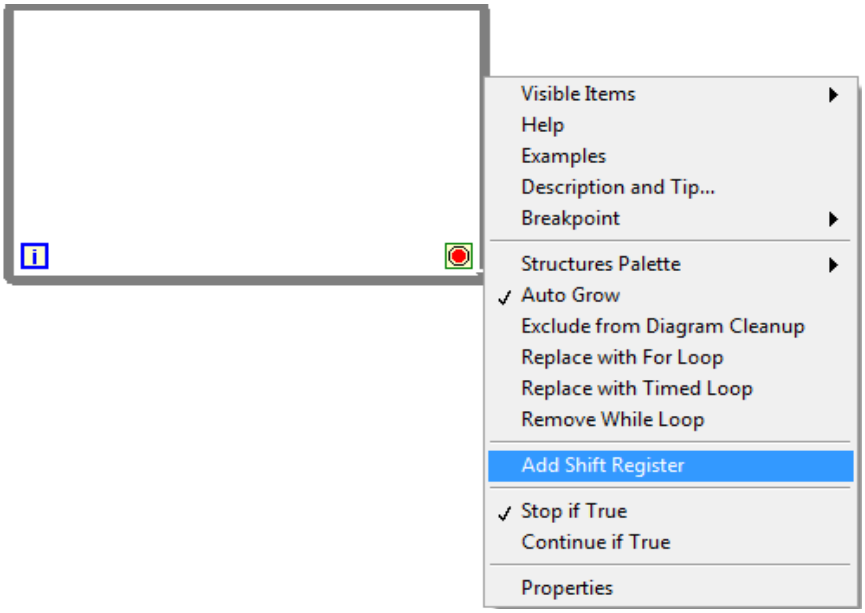



**Figura 105:** Seleccionando un Shift Register.

Si hace clic en Agregar registro de desplazamiento, se verá que va a añadir dos cajas con triángulos en los lados: izquierdo y derecho del ciclo While, como se muestra en la Figura 106.

**Figura 106:** Ciclo While con shift registers.

La razón de que son de color negro se debe a que no han dicho qué tipo de datos almacenará. Una vez que le diga el tipo de datos que almacenará los registros de desplazamiento cambiará de color para que coincida con el tipo de datos. Lo principal que tiene que recordar acerca de los registros de desplazamiento es que el lado izquierdo es una salida, es decir, el lado izquierdo da datos sobre la última vez por el ciclo. El lado derecho es un insumo, por lo que almacena la información aquí lo que quiere recordar en la siguiente iteración del ciclo.

Si hacemos clic sobre el registro de desplazamiento, nos damos cuenta de que al tratar de mover un lado de arriba a abajo, el otro se mueve con él. Los registros de desplazamiento siempre trabajan en parejas. Como se ha mencionado antes, los registros de desplazamiento son de color negro porque no hay ningún tipo de datos que se les asigne.

Ahora que sabe lo que hacen los registros de desplazamiento, puede crear un programa que muestre los números al azar a partir de la iteración actual y la anterior iteración de un ciclo While. Es importante señalar que este programa se creará en la instancia principal de la aplicación porque no va a estar en contacto con el controlador NXT.

Va a usar el ciclo While, con un registro de desplazamiento. También va a utilizar un generador de números aleatorios que pueden encontrar en la paleta de funciones de la instancia principal de la aplicación en la sub-paleta numérica, como se muestra en la Figura 107.

Va a colocar el generador de números aleatorios dentro del ciclo While y lo cableará a la parte derecha del registro de desplazamiento, como se muestra en la figura 108.

Como puede ver, los registros de desplazamiento se han vuelto de color naranja debido a que el cable que ha conectado es de tipo de datos double.

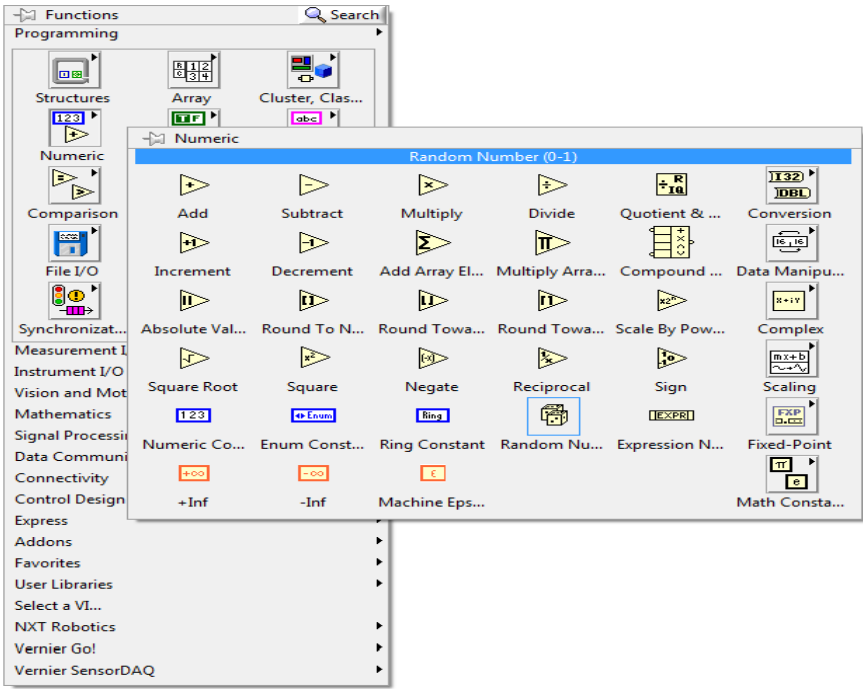



**Figura 107:** Localización del generador de números aleatorios.

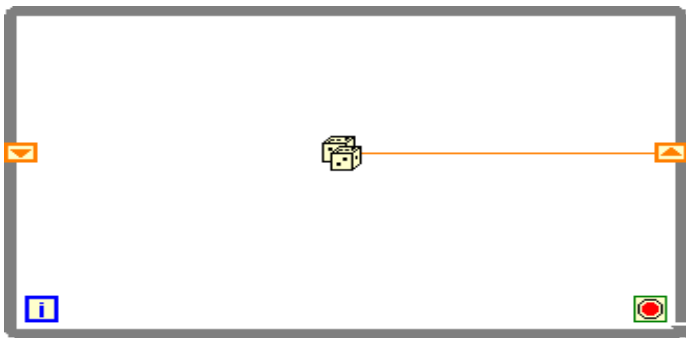


**Figura 108:** Generador de números aleatorios conectado al shift register.

En la Figura 108, el número al azar se da en el lado derecho del registro de desplazamiento. Esto significa que el lado izquierdo del registro de desplazamiento contendrá el valor de la última vez que pasa por el ciclo. Si crea un indicador del lado izquierdo del registro de desplazamiento, como se muestra en la Figura 109 podrá observar los valores de la última iteración.

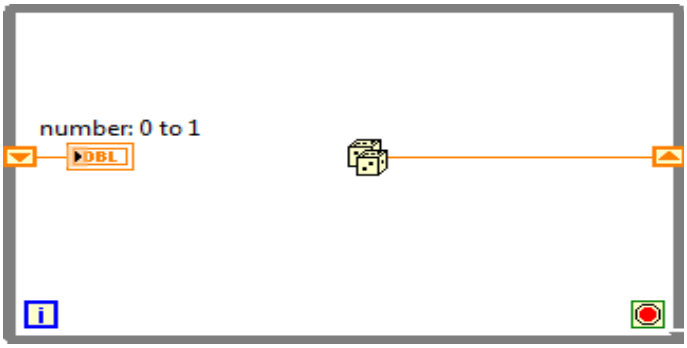



**Figura 109:** Monitoreando los números aleatorios en la última iteración.

También debe cambiar la etiqueta del indicador que acaba de agregar "Último número al azar".

También va a crear un indicador en el cable que sale de la terminal de salida del generador de números aleatorios con el fin de comparar los dos valores. Debe cambiar el nombre del indicador a "Actual número al azar". Esto se muestra en la Figura 110.

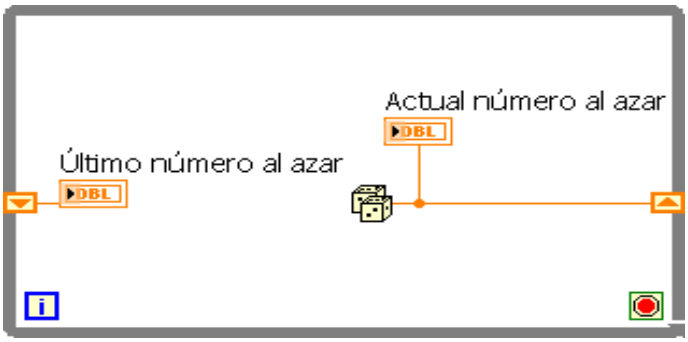



**Figura 110:** Comparando los números aleatorios de la iteración anterior.

Lo único que queda por considerar es el valor inicial. ¿Cuál es el primer valor que entra en el lado izquierdo del registro de desplazamiento? Para el resto de las iteraciones, tiene el número al azar de la iteración anterior, pero la primera vez que entra en el ciclo, no hay ningún valor en el lado izquierdo del registro de desplazamiento. Es por eso que hay que inicializar el registro de desplazamiento.

Para iniciar el registro de desplazamiento, puede crear una constante numérica y cablear en el lado izquierdo del registro de desplazamiento, como se muestra en la Figura 111. Puede dejar la constante numérica en cero.

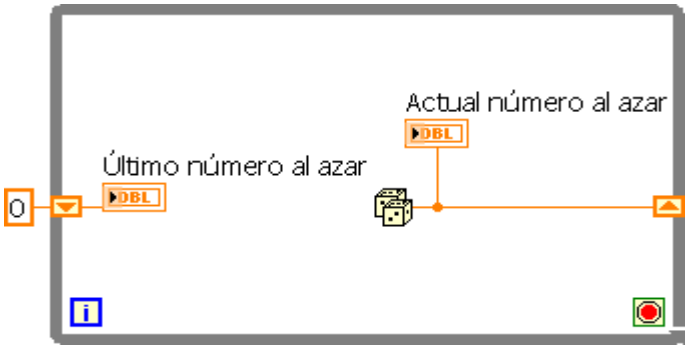



Figura 111: Inicializando el shift register.

El único propósito de cablear la constante de afuera del circuito en la parte izquierda del registro de desplazamiento es que lo inicie. Esto significa que solo se utiliza la primera vez que se ejecute el ciclo, el valor en el indicador denominado "Último número al azar" será 0. En cualquier otro momento, el valor de entrada en el indicador denominado "Último número al azar" será el del generador de números aleatorios a partir de la iteración anterior.

Para completar el ejemplo que está trabajando, tiene que añadir unas cuantas cosas más. Sólo entonces será capaz de ver exactamente cómo trabajan los registros de desplazamiento. En lugar de crear un ciclo infinito, hará un botón de parada, que terminará el ciclo y detendrá el programa cuando haga clic. Para crear un botón de paro, es necesario hacer clic derecho en el panel frontal, seleccione la paleta Buttons & Switches y haga clic en Stop Button. Esto se muestra en la Figura 112.

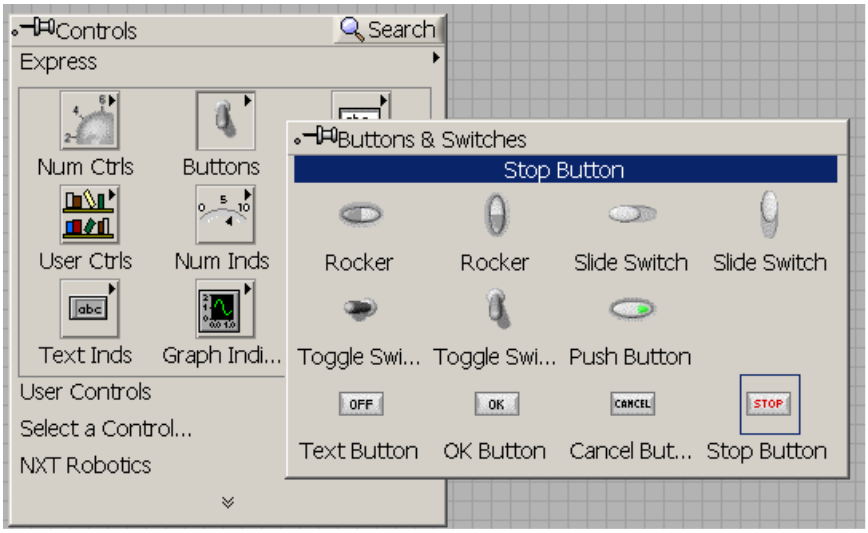



**Figura 112:** Seleccionando el Stop Button.

Cuando coloca el botón de paro en el panel frontal, se observará como la figura 113.


**Figura 113:** Botón de paro.

Un botón de paro también aparecerá en el diagrama de bloques y se verá como la Figura 114.


**Figura 114:** Botón de paro en el diagrama de bloques.

Ahora tiene que cablear este botón de paro a la terminal condicional. Cuando se pulse el botón en el panel frontal, mostrará un valor verdadero y finalizará el ciclo While. El programa debe ser similar a la Figura 115.

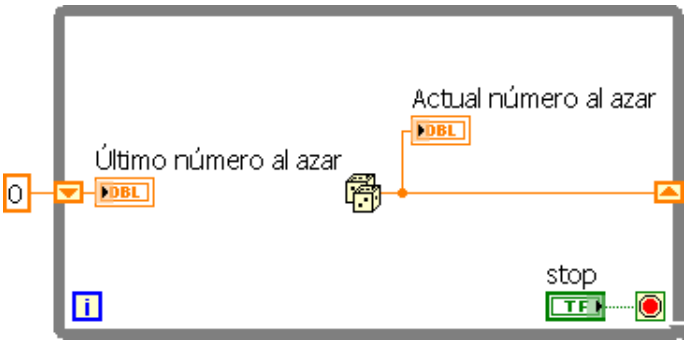



**Figura 115:** Diagrama de bloques del programa.

Para ejecutar este programa, tiene que asegurarse de que está en la instancia principal de la aplicación ya que este código no utiliza el NXT. Si ejecuta este programa, se dará cuenta de que la muestra en el panel frontal cambia muy rápido, más rápido de lo que pueda notar. Para corregir esto hay que añadir un retraso de tiempo. Esto se puede encontrar haciendo clic derecho sobre el diagrama de bloques en la instancia principal de la aplicación y la selección de sincronización, y haciendo clic en Espera. Esta función espera puede ser colocada en cualquier parte del ciclo While, y puede crear una constante de la cantidad que quiera que la demora en milisegundos. Si quiere esperar dos segundos, por ejemplo, la constante será de 2.000 milisegundos.

Esta listo para ejecutar el programa, como se muestra en la Figura 116.

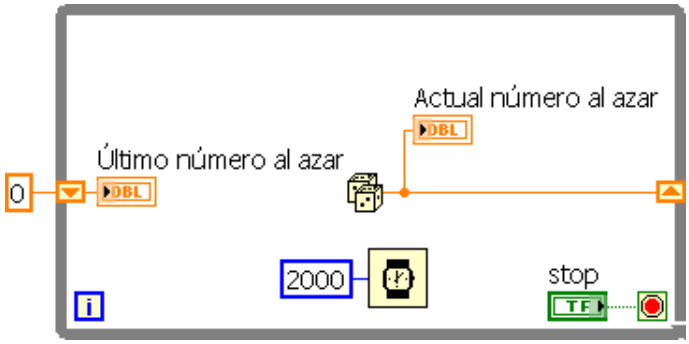



**Figura 116:** Diagrama de bloques del programa.

En el panel frontal, verá la siguiente secuencia, como se muestra en la Figura 117.

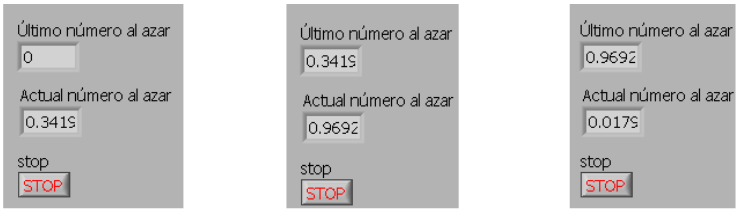



**Figura 117:** 3 iteraciones del ciclo While del programa.

Los números aleatorios serán diferentes, pero este programa es una buena manera de ver cómo funciona la acción de los registros de desplazamiento, y ver su utilidad.

Ahora se hablará de algunas funciones booleanas útiles que nos ayudarán a completar el ejercicio.

La primera de ellas es la función Not. La función no sólo tiene un valor booleano y luego la da. Si el valor de entrada es cierto, el valor de salida de la función no será falso. Si el valor de entrada es falso, el valor de salida de la función no será verdadero.

Esta función se puede encontrar haciendo clic derecho sobre el diagrama de bloques, seleccionar el operador booleano en la sub-paleta y haga clic en la función Not. La función Not se muestra en la Figura 118.

**Figura 118:** Función Not.

La siguiente función es llamada Select. Esta función permite seleccionar uno de dos valores posibles sobre la base de un valor booleano. La función Select se ve como en la Figura 119.

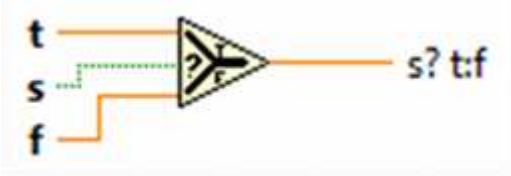



**Figura 119:** Función Select.

Hay tres terminales en el lado izquierdo: un valor booleano que va en el medio, un valor que va en la parte superior y un valor que va en la parte inferior. Si el valor booleano es cierto, el valor de la parte superior "t" llega a la salida. Si el valor booleano ingresado a la función Select es falso, el valor de la parte inferior "f" se emite.

La función Select es en realidad una versión simplificada de una estructura de casos, ya que tiene dos valores y todo lo que quiere hacer es seleccionar uno de ellos.

La función Select se puede encontrar haciendo clic derecho sobre el diagrama de bloques, la selección de la comparación de sub-paleta y haciendo clic en Selección. Esto se muestra en la Figura 120.

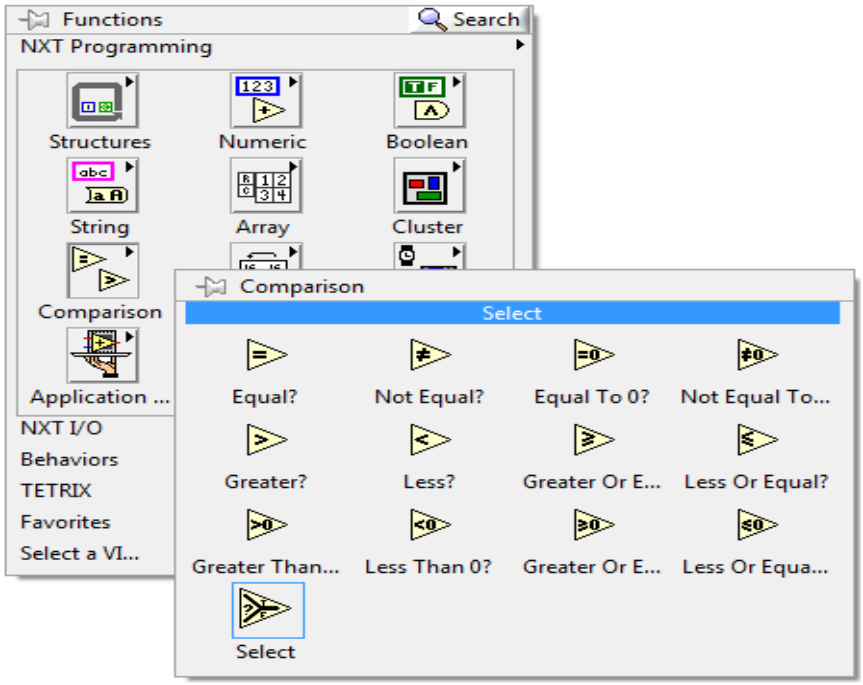



**Figura 120:** Localización de la función Selección.

Puede crear un programa sencillo como el mostrado en la Figura 121, para ver el uso de las dos funciones booleanas.

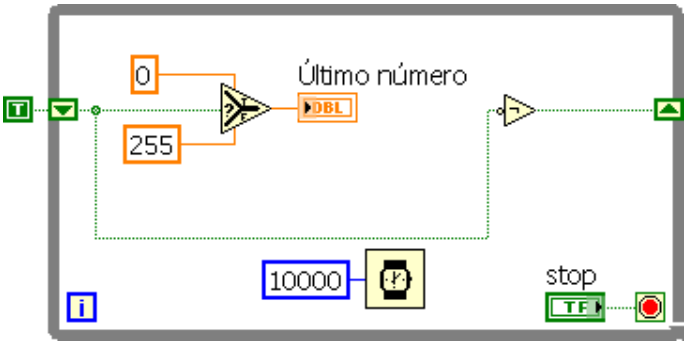



**Figura 121:** Función Select y Not en acción.

En este programa, va a visualizar el número 0 ó 255 en función del valor booleano que se introduzca en la función Select. Este programa comienza por introducir una constante verdadera en el registro de desplazamiento del ciclo While. Entonces esta constante verdadera se introduce en la función de selección. Ya que un valor booleano "verdadero" se introduce en la función Select, se selecciona el valor de la parte superior "t". Ha designado que este sea una constante numérica con valor 0. El número 0 es entonces mostrado en un indicador. A continuación, se genera el valor booleano opuesto al registro de desplazamiento a la derecha. Esto significa que la segunda vez que se ejecuta en el ciclo, será falso. Esto significa que la función de Selección elegirá el valor "f", que ha designado para ser una constante numérica con valor 255. Este número se muestra en el indicador. Cada vez que este programa se ejecuta, obtendrá salidas alternativas entre 0 y 255, pero a partir de 0, ya que se ha inicializado el registro de desplazamiento en "verdadero". Este ciclo, mientras que también contiene un retardo de diez segundos, observará que una vez pasado este, el valor en el indicador cambiará. También es importante señalar que este programa se debe ejecutar en la instancia de la aplicación principal.

**Taller:** Hacer que el robot NXT avance cuando el sensor ultrasónico es disparado. Después de eso, cada vez que se activa el sensor ultrasónico, el robot debe ir en la dirección opuesta de su último movimiento. El programa debe detenerse cuando el sensor de contacto es presionado.

# MÁQUINA DE ESTADO

Las máquinas de estado son una arquitectura de programación en la que rompen las acciones de un robot en una serie de modos, también conocido como estados. La forma de poner en práctica una máquina de estado en LabVIEW ™ es esencialmente una estructura de caso dentro de un ciclo While.

Recuerda cómo una estructura de caso de LabVIEW: le permite a uno de los muchos casos ejecutarse en un momento dado sobre la base de una condición. Las estructuras de caso que ha visto hasta el momento siempre han utilizado un valor booleano en la entrada. Va a aprender que se puede utilizar una cadena y la entrada al selector de la estructura de casos como una forma de definir más de dos estados diferentes en la estructura del caso.

El estado de un robot puede estar cambiando constantemente. Las máquinas de estado se ven afectadas por los sensores. En base a esto, el comportamiento de un robot puede ser diferente, basado en lo que ven, oyen o sienten. Por lo tanto, puede tener más de un camino a seguir en la máquina de estados.

La mejor manera de representar una máquina de estados es con un diagrama de estado. Un ejemplo se muestra en la Figura 122.

En un diagrama de estado, el flujo del programa se muestra con flechas entre varios estados, representados por círculos. Hay escritura en las transiciones antes de los estados o en los propios Estados para decir lo que está sucediendo, o cuando las situaciones se producen.

Las transiciones son el flujo entre los Estados y están representados por las flechas en el diagrama de estado. Es importante recordar que no puede haber más de una flecha que sale de un estado en particular, cada uno en representación de muchas condiciones posibles.

Analizará el diagrama de estado en la Figura 122. El círculo con el borde doble que dice "START" en el interior es donde el programa se inicia. Si sigue la flecha, verá que el programa entra en el Estado "Idle". La razón de que este estado es llamado inactividad se debe a que no está haciendo nada. Por lo tanto, cada vez que el robot se encuentra en estado de reposo, el robot no debe realizar ninguna acción.

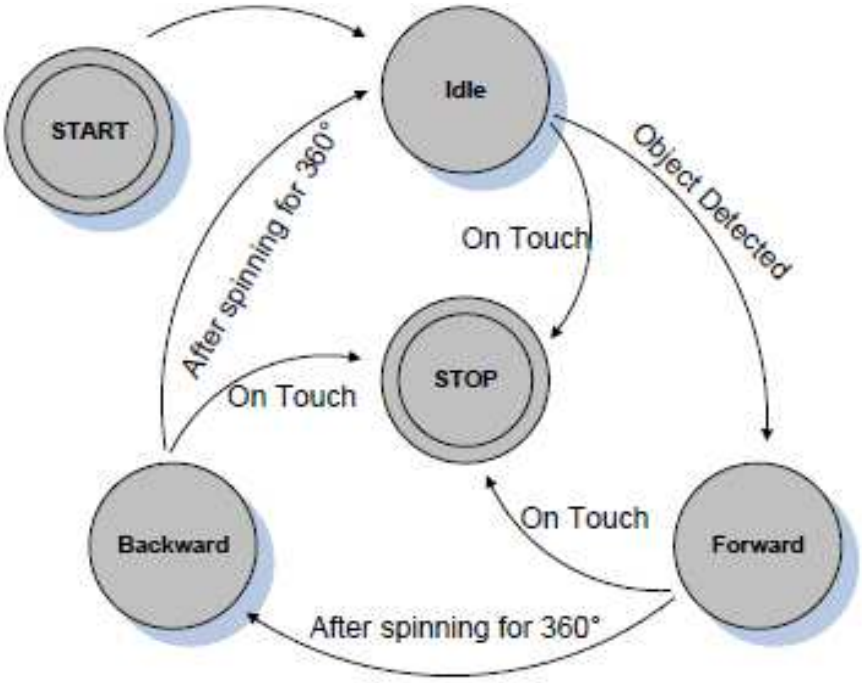



**Figura 122:** Diagrama de estado.

Si se fija en el estado de reposo, verá que hay dos flechas que salen de ella. Una va al Estado de "STOP", que es el final del programa, y el otro pasa a un estado llamado "Forward". Si se fija bien, verá que el Estado "Forward" sólo se produce cuando se detecta un objeto. Es por eso que la flecha que apunta a ese estado se denomina "Object Detected". Además, verá otra flecha marcada entre el Estado "Idle" y el Estado "STOP", que se llama "On Touch". Esto significa que cuando el sensor de contacto es presionado en estado de reposo, el programa va al Estado "STOP". Las etiquetas en las flechas son un camino corto para representar que va a pasar de un estado a otro.

Si tuviera que ir a "START" y luego al Estado "Idle", y asumir que un objeto fue detectado por el sensor ultrasónico, y luego ir al Estado "Forward". No tiene información completa sobre el Estado "Forward", pero a partir del nombre, puede suponer que se trata de que el robot mueva hacia delante. Una vez más, tiene dos

salidas posibles de este Estado "Forward". Cuando el sensor de contacto es presionado, se va al Estado "STOP" y después de que se ha avanzado durante un segundo, se dirige al Estado "Backward". El Estado "Backward" implica decirle al robot que se va a mover hacia atrás.

Desde el Estado "Backward", si el sensor de contacto es presionado, va a "STOP", y después de un segundo, vuelve al Estado "Idle".

Como un dato entero o double, hay otro tipo de datos en LabVIEW llamado cadena. A diferencia de un valor numérico que utiliza números, o un valor booleano que utiliza valor verdadero o falso, en una cadena se pueden utilizar letras y pares de números. Va a utilizar las constantes de cadena para nombrar y describir los estados dentro de una máquina de estados.

Creará una máquina de estados en LabVIEW. Como se mencionó anteriormente, una máquina de estados es simplemente una estructura de caso dentro de un ciclo While. Esto se muestra en la Figura 123.

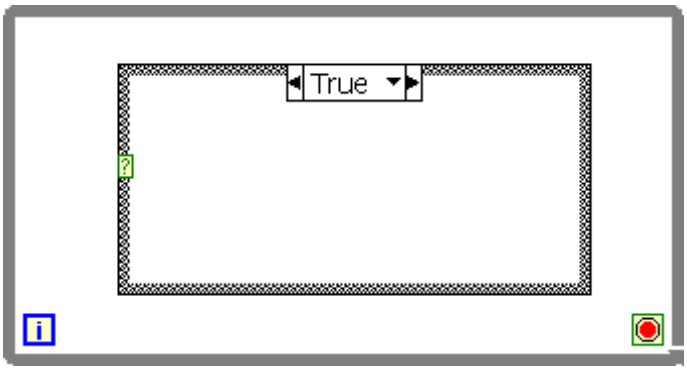



Figura 123: Máquina de estado.

Para que un robot tome una decisión, tiene que saber en qué estado esta, ya que realiza diferentes acciones en diferentes estados y tiene diferentes transiciones hacia y desde estos estados. Esto lleva al concepto que ha aprendido anteriormente en relación con los registros de desplazamiento. Con el fin de recordar la información de la repetición del ciclo a la siguiente iteración, debe tener un registro de desplazamiento. En este caso, el cambio de registro va a contener el estado actual del robot.

Puede recordar, para crear un registro de desplazamiento haga clic derecho en el borde del ciclo While y haga clic en agregar registro de desplazamiento. El resultado se muestra en la Figura 124.

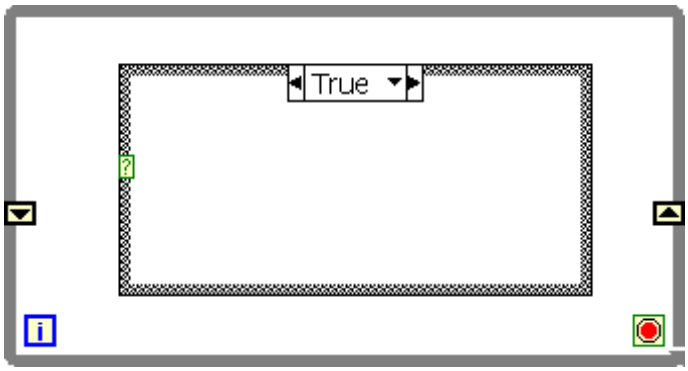



**Figura 124:** Máquina de estado con registro de desplazamiento.

Tiene que iniciar este cambio de registro y dirá que en este caso, quiere que la máquina de estado inicie en el estado "Idle". Para ello, va a crear una constante String. Esto se puede hacer dando clic derecho sobre el diagrama de bloques, seleccionando la sub-paleta String y haga clic en String Constant. Esto se muestra en la Figura 125.

Ahora puede escribir la palabra "Idle" y cablear la *String Constat* al registro de desplazamiento de la izquierda. Esto se muestra en la Figura 126.

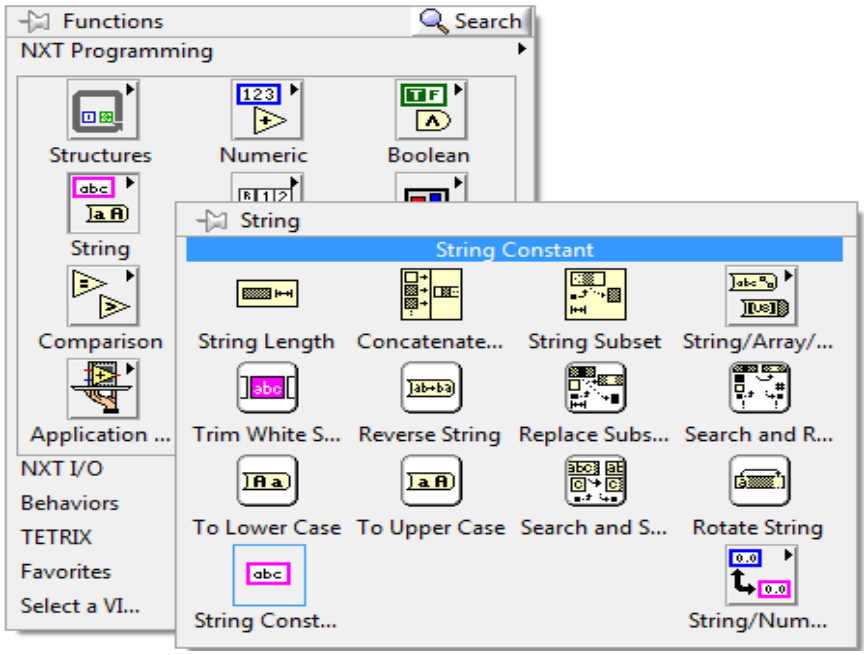



**Figura 125:** Localización de la String Constant.

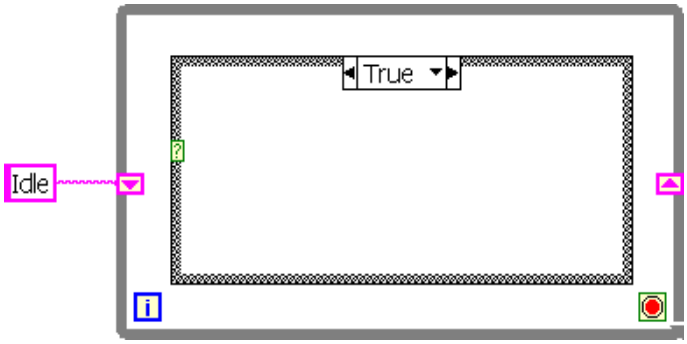



**Figura 126:** Inicializando el Shift Register.

Recuerde que todavía tiene una estructura de caso dentro del ciclo While. La estructura de caso le permite al robot hacer algo diferente, dependiendo de qué valor se conecta en el selector de caso. Si conecta el Shift Register de la izquierda al selector de caso, tendrá la capacidad para definir código diferente para los diferentes estados. Si el valor inicializado en el Shift Register es la cadena denominada “Idle”, entonces para escribir el código que se ejecutará en el estado “Idle”, necesita cambiar la String en la parte superior de la estructura de caso y denominarla “Idle”. Para renombrar un caso, simplemente haga doble clic sobre su nombre y escriba el nombre deseado. Esto se muestra en la Figura 127.

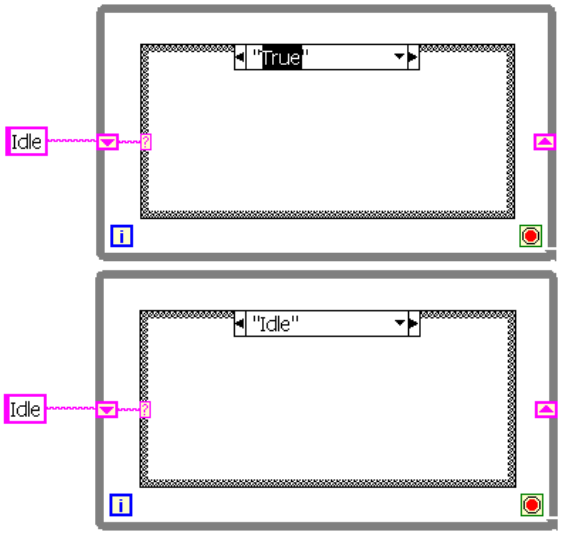



**Figura 127:** Renombrando un caso.

Ahora puede escribir el código en el estado “Idle” que se ejecutará tan pronto como se inicia el programa en el NXT.

Dirá que tiene una máquina de estados muy simple, donde se pasa del estado “Idle” al estado “2nd” o al estado “STOP”. La verdadera pregunta aquí es ¿Cómo podemos enviar el programa desde el estado “Idle” hasta el estado “2nd”? Para ello, tiene que crear una nueva String Constant en el estado “Idle” y llamar al estado “2nd”. A continuación cablear hasta el shift register de la derecha. Es por eso que tiene el shift register, para almacenar el estado. Esto ahora le permitirá pasar del estado “Idle” al estado “2nd”, como se muestra en la Figura 128.

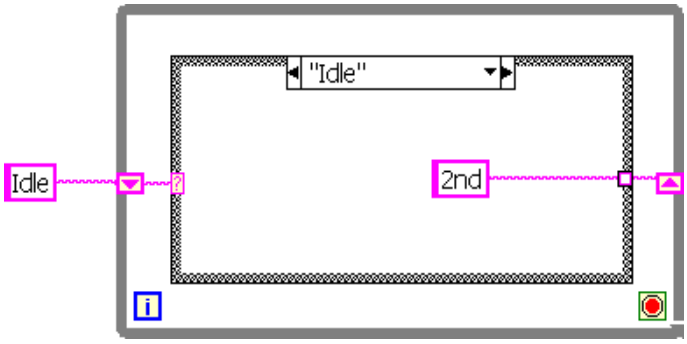



**Figura 128:** Ir de un estado a otro.

Ahora tiene que añadir un caso para el estado “2nd”. Para ello, haga clic derecho sobre la etiqueta del selector en la parte superior de la estructura de caso y seleccione Add Case After, como se muestra en la Figura 129.

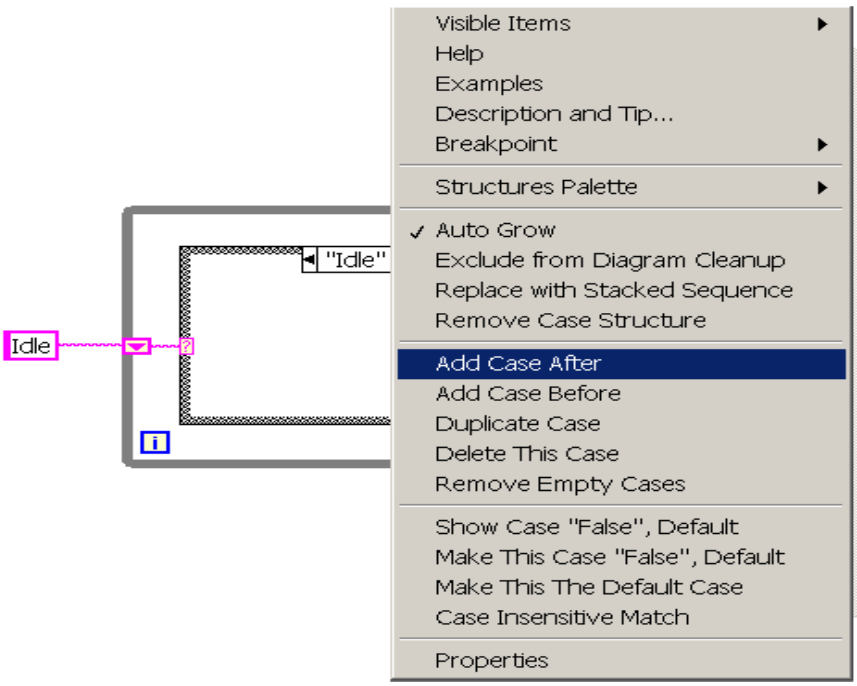



**Figura 129:** Adiriendo un caso despues.

Esto agregará una caja con un nombre en blanco, y puede darle el nombre correcto. En este caso se llamará “2nd”.

Ahora tiene que decirle al programa que irá desde el estado “2nd” hasta el estado “STOP”. Una vez más, tiene que crear una String Constant, con un nombre adecuado y conectarlo al shift register en el lado derecho. Los resultados se muestran en la Figura 130.

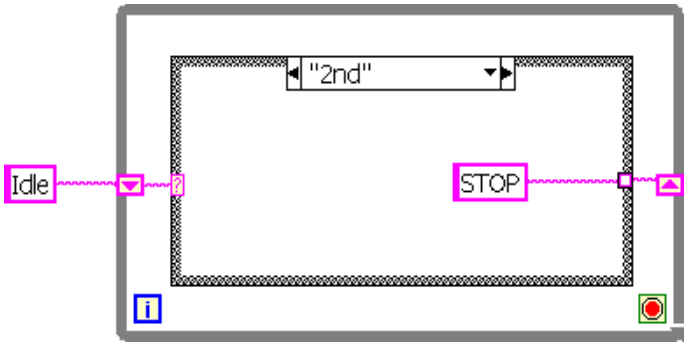



**Figura 130:** Ir al estado STOP.

La última cosa que necesita hacer es crear un estado "STOP". Va a observar su selección de casos y ver lo que tiene. Esto se muestra en la Figura 131.

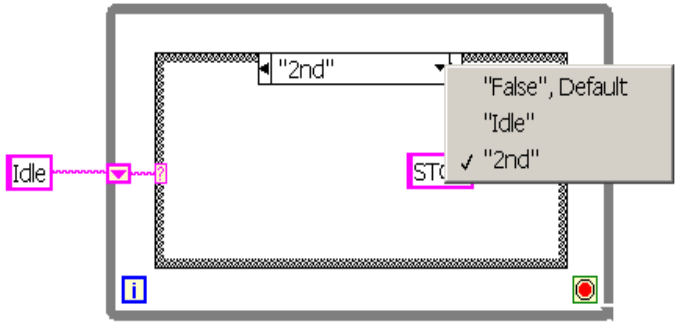



**Figura 131:** Menú desplegable del selector de casos.

Verá que tiene el estado "Idle" y el estado "2nd" que ha creado. Pero también tiene un tercer estado "False" que estaba allí por defecto cuando creamos la estructura de casos. Además es importante que tenga en cuenta la palabra Default a lado de la etiqueta del estado "False". El estado Default significa que el programa se destinará ahí si ninguno de los otros estados se aplica.

Utilizará el estado Default y cambiará el nombre para que sea el estado "STOP". Para cambia el nombre de un caso ya existente, puede hacer doble clic sobre la etiqueta y a continuación, escriba el nombre que le gustaría que tuviera. Esto se muestra en la Figura 132.

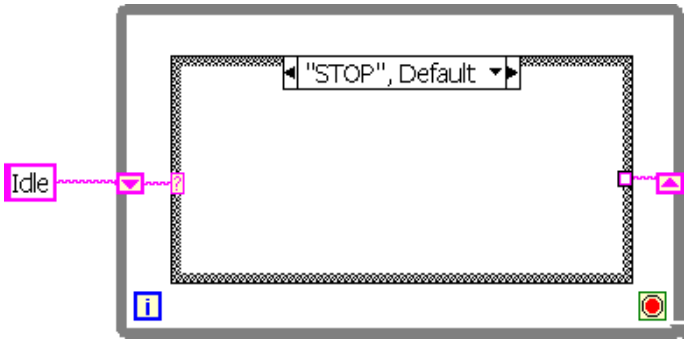



**Figura 132:** Estado Default.

En el estado "STOP", simplemente tiene que crear una manera de terminar el ciclo While. Para ello, creará una constante booleana True y la cableará hasta la terminal condicional. Esto se muestra en la Figura 133.

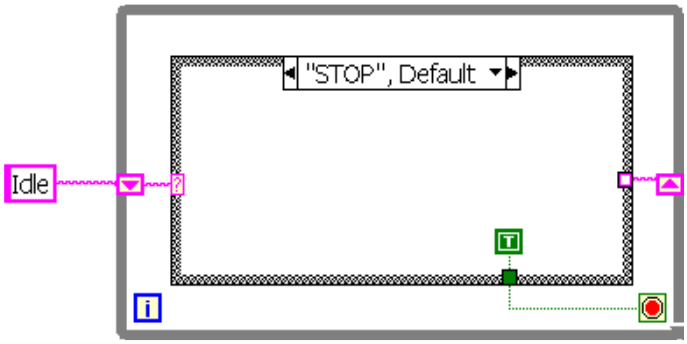



**Figura 133:** Estado "STOP".

Usted puede notar que tiene un icono con la flecha rota en la barra de herramientas. Si hace clic en la flecha rota, aparecerá una lista de errores, como se muestra en la Figura 134.

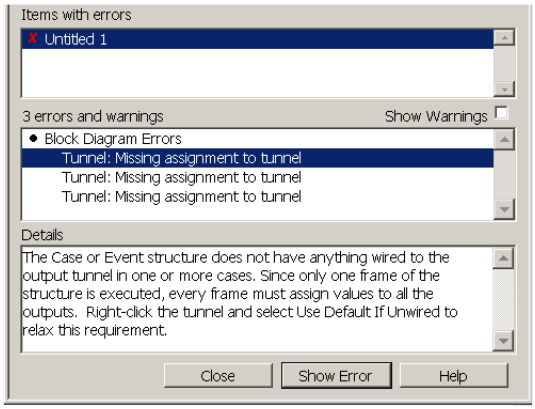



**Figura 134:** Lista de errores.

Verá que ha cometido el mismo error tres veces. No ha cableado los túneles de todos los casos, como se muestra en la Figura 135.

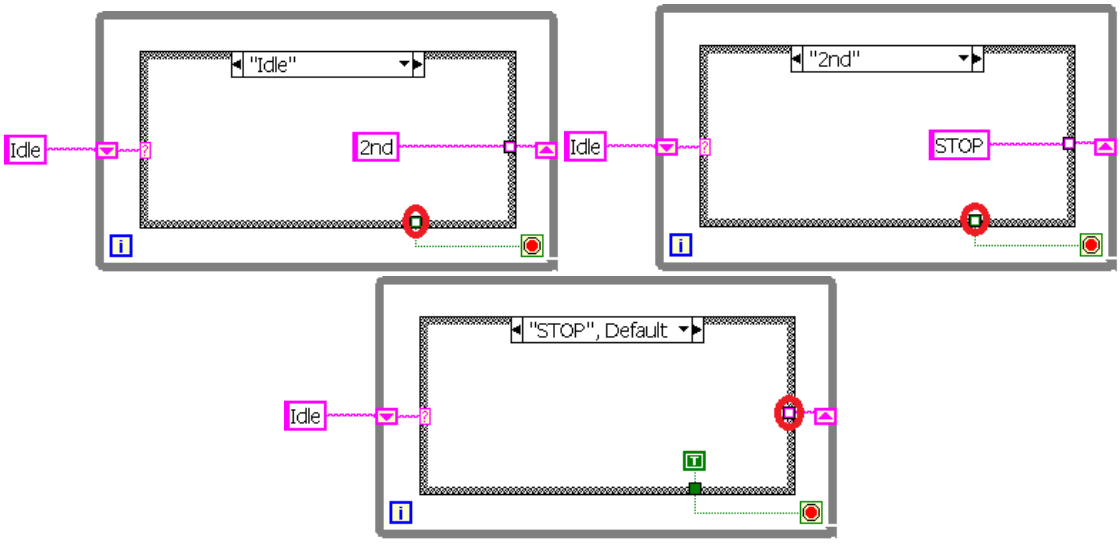



**Figura 135:** Túneles vacíos.

Vuelva al estado “Idle” y corrija el túnel vacío causado por la constante booleana. En el estado “STOP” tiene un valor True cableado a la terminal condicional del ciclo While para finalizar el programa, pero en el estado “Idle” y en el estado “2nd”, no queremos que termine el ciclo, así que será una constante False en el túnel para ambos casos. Hace esto mediante la colocación de constantes False en el túnel, tanto en el estado “Idle” como en el caso “2nd”. Esto se muestra en la Figura 136.

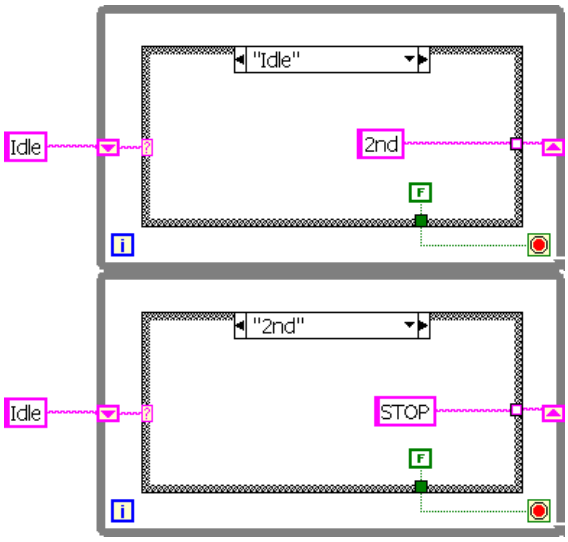



**Figura 136:** Cableado de la constante False en el estado “Idle” y “2nd”.

Todavía tiene un tunel vacío que es necesario corregir y que esta en el estado “STOP” porque no esta conectado al shift register del lado derecho. Despues de que el estado “STOP” no esta haciendo la transición a otro estado, sino solo esta finalizando el programa en ese estado. Puede conectar una cadena vacía para el shift register del

lado derecho o, simplemente, cablear el chift register desde el lado izquiero hasta el shift register del lado derecho, como se muestra en la Figura 137.

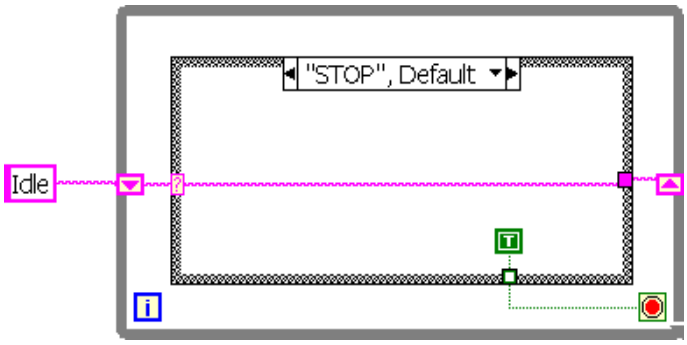



**Figura 137:** Cableando el shift register del lado derecho con el shift register del lado izquierdo.

Ahora se dará cuenta de que el túnel está lleno y ya no hay ningún error.

Hay un atajo que puede usar cuando tiene un túnel vacío, que es conveniente, pero puede ser algo peligroso. Va a quitar las dos constantes False que añadió en el estado "Idle" y en el estado "2nd" y retire el cable que creo en shift register en el estado "STOP". El acceso directo es hacer clic derecho en el túnel vacío, y seleccionar Use Default if Unwire. Solo lo hará por el cable verde proveniente de la constante booleana por ahora. Esto se muestra en la Figura 138.

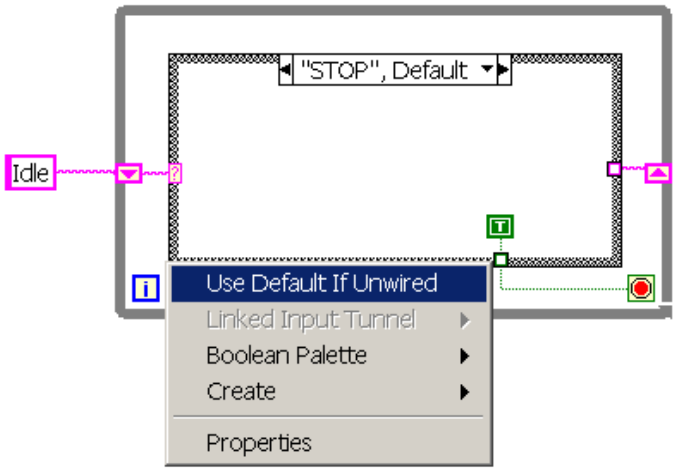



**Figura 138:** Acceso directo para tuneles sin cables.

Tenga en cuenta que despues de seleccionar Use Default if Unwired, el cuadro se medio llena, como se muestra en la Figura 139.

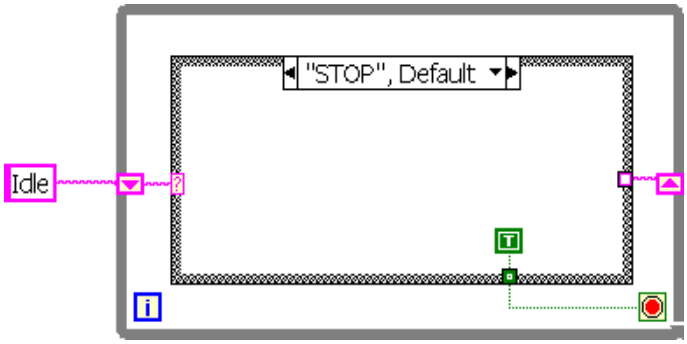



**Figura 139:** Tunel medio lleno.

Si se desplaza a través de los otros estados, el tunel sigue estando medio lleno, a pesar de que en los otros estados no hay nada conectado. Esto se muestra en la Figura 140.

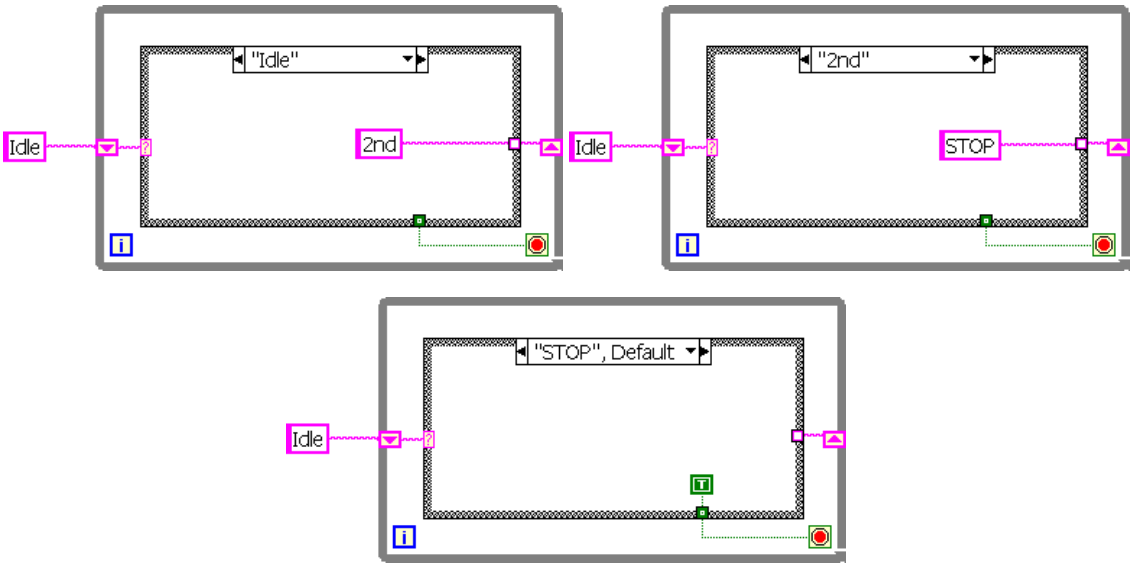



**Figura 140:** Los tres estados.

El valor por defecto que se utiliza en los dos casos que no tienen un cable conectado al tunel verde es False. La razón por la cual es False, es porque en los dos estados que no tienen algo conectado al tunel verde, no queremos que nuestro programa se detenga. Si los valores son False y esos datos entran en la terminal condicional eso significa que no se detendrá el ciclo. Solo el estado “STOP” da un True al valor de la terminal, por lo que solo el estado “STOP” terminará el ciclo y detendrá el programa.

Puede hacer lo mismo con el tunel rosa, que obtiene un valor del String y lleva la información al shift register. Debe tener cuidado porque si accidentalmente olvida cablear algo que tenía la intención que estuviera conectado, LabVIEW no le dice que es un error.

Por ejemplo si selecciona la opción Use Default if Unwired para el tunel rosa, y en el estado “2nd”, olvida cablear la String Constan “STOP” al tunel de color rosa, al igual que en la Figura 141.

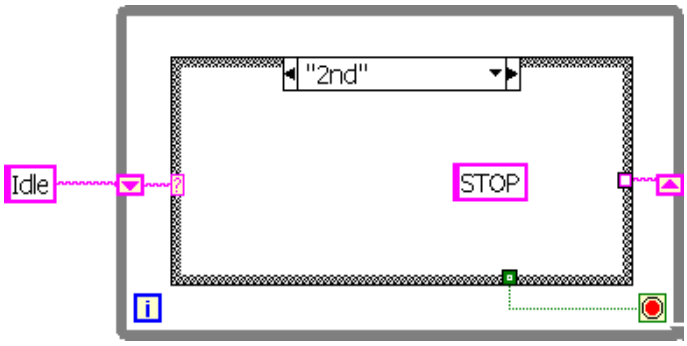



**Figura 141:** Desconexión de la String Constant “STOP”.

En este escenario, el programa se iniciará en el estado “Idle”, y luego irá al estado “2nd”, en el estado “2nd”, algo interesante se va a producir. Puesto que la String Constant no esta conectada, como se muestra en la Figura 141, una cadena vacía se pasa al shift register de la derecha. Sin embargo, no tenemos un caso de una cadena vacía, asi que la pregunta es ¿qué ocurrirá?

¡Irá al caso Default¡ Esto significa que se perderá totalmente el estado “2nd” por lo que es muy importante tener cuidado al seleccionar Use Default if Unwired con las cadenas. Se recomienda que no utilice esta opción en todos los cables de color rosa que contienen cadenas, de modo que no se le olvide cablear algo. Para ello basta conectar una String Constant vacía en el shift register en el caso “STOP”. Esto se muestra en la Figura 142.

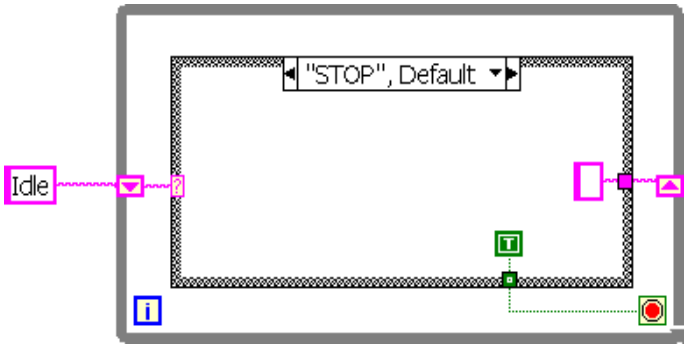



**Figura 142:** Cadena vacía para omitir el Use Default if Unwired.

También se puede notar qe el túnel se ha vuelto a llenar totalmente de color rosa, y que confirma que ha conectado algo a ese túnel en cada caso.

**Taller:** Implemente el siguiente diagrama de estado:

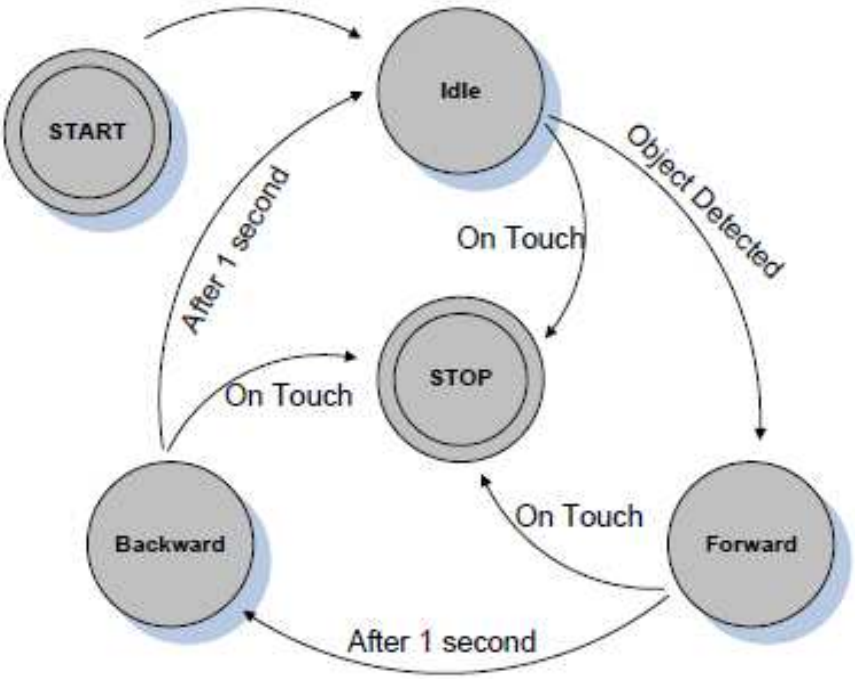



**Figura 143:** Diagrama de estados.

# UNA GRAN MÁQUINA DE ESTADO

Una estructura de caso anidada se utiliza cuando hay una estructura de caso dentro de otra estructura de caso. Estas estructuras anidadas llegan a ser muy utiles cuando tiene dos sensores, tales como el sensor de contacto y el sensor ultrasónico. El sensor de contacto genera un valor booleano, y lo mismo ocurre con el sensor ultrasónico. Tiene dos sensores con valores booleanos, lo que significa que tiene cuatro posibilidades diferentes: una posibilidad verdadera y una posibilidad falsa por cada sensor.

Utilizará la estructura de caso exterior para un sensor, y la estructura de caso interior para el otro sensor.

Una representación de casos anidados se muestra en la Figura 144.

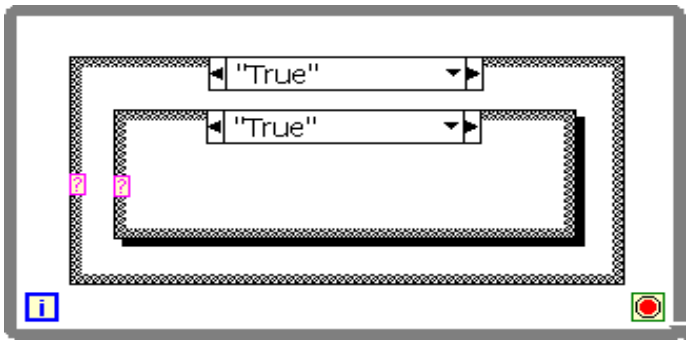



**Figura 144:** Estructura de caso anidada en un ciclo While.

Para comprender más sobre el uso de estructuras de caso anidadas con multiples sensores, va a crear un programa simple. Quiere que el robot se mueva siempre hacia adelante, pero quiere comprobar constantemente los sensores. Si el sensor ultrasónico o el sensor de contacto se dispara, el robot debe cambiar de dirección. Cada vez que se dispara un sensor, el robot debe hacer un pequeño giro.

Comenzará colocando funciones de control de motores dentro de un ciclo While como se muestra en la Figura 145.

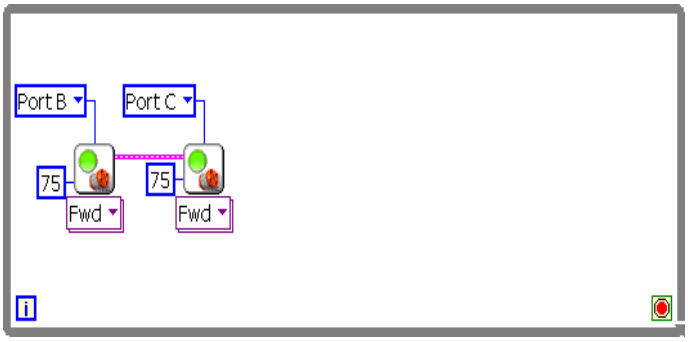



**Figura 145:** Control de motores dentro de un ciclo While.

A continuación creará las funciones de los sensores, similar a la forma en que los ha trabajado anteriormente, con un sensor de contacto y un sensor ultrasónico. Puede decir que el umbral para el sensor ultrasónico debe ser de 20 cm. También es necesario conectar la terminal NXT. El código debería parecerse a la Figura 146.

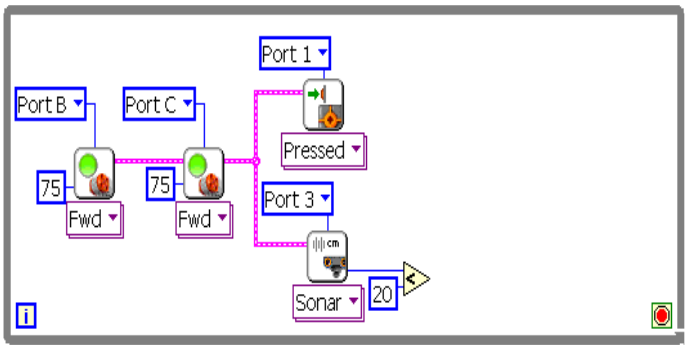



**Figura 146:** Código con sensores.

Ahora tiene que crear las estructuras de casos. Esta es la parte dificil. Dado que quiere leer la información de los sensores al mismo tiempo, como se ha mencionado antes, va a crear una estructura de caso anidada. Una estructura de caso será para el sensor de contacto, y la otra para el sensor ultrasónico.

Va a cablear el selector de caso de la estructura de caso exterior con el sensor de contacto. Si el sensor de contacto es presionado, mostrará un valor verdadero. Eso significa que el caso “True” se ejecutará, y puede poner código para que haga algo diferente aquí. Si el sensor de contacto no es presionado, se destinará al caso “False”. En este caso se quiere comprobar si el sensor ultrasónico se ha disparado. Aquí es donde la segunda estructura de caso irá. Va a cablear la salida booleana de la función de comparación al selector de caso de la estructura de caso interior. Si el sensor ultrasónico devuelve un valor verdadero, el caso “True” se ejectura, y aquí es donde va

a colocar el código para que haga algo diferente. Si el sensor ultrasónico devuelve un valor falso, el robot no hará más que seguir adelante.

Es importante tener en cuenta que tiene que crear una constante booleana "False" para la terminal condicional del ciclo While.

El código hasta ahora se muestra en la Figura 147.

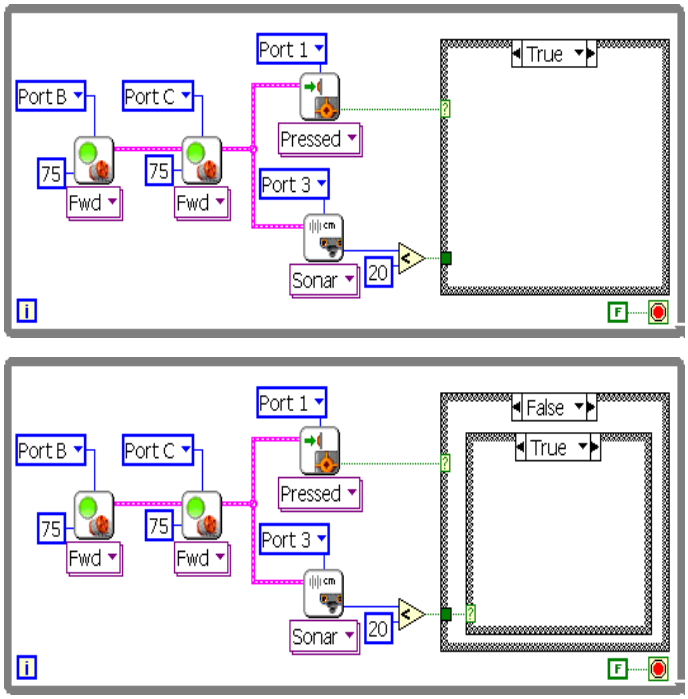



**Figura 147:** Estructuras de caso adecuadamente dispuestas.

Ahora todo lo que tiene que hacer es crear el código para el giro a la izquierda.

La mejor forma de escribir el código para dar la vuelta es con la función Wait For. Esta función se puede acceder mediante la sub-paleta NXT I/O. Puede hacer que uno de los motores avance hacia atrás por medio segundo y eso será suficiente para que el robot gire.

Si selecciona 0.5 segundos como la constante para la terminal Time (1 sec), que será perfecto para dar la vuelta con una velocidad del motor de -75. Esto se muestra en la Figura 148.

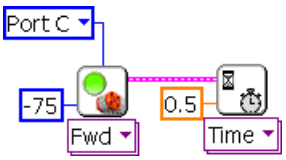



**Figura 148:** Girar el motor solo 0.5 segundos a una velocidad de -75.

Este código ahora tiene que se colocado en el lugar correcto. Este código debe ser colocado en el caso verdadero tanto de la estructura de caso exterior como en la estructura de caso interior.

Revisará lo que el código debe hacer. Siempre se va a mantener girando el motor hacia adelante, mientras que constantemente se verifican los sensores. Una vez que el sensor de contacto se dispara, el caso “True” de la estructura de caso exterior se ejecutará. Si el sensor de contacto no se dispara el caso “False” se ejecutará. El caso “False” contiene una estructura de caso en su interior. El caso “True” de la estructura de caso interna se ejecutará si el sensor ultrasónico se ha disparado. En el caso “False”, el programa no hará nada.

El código debería parecerse al de la Figura 149.

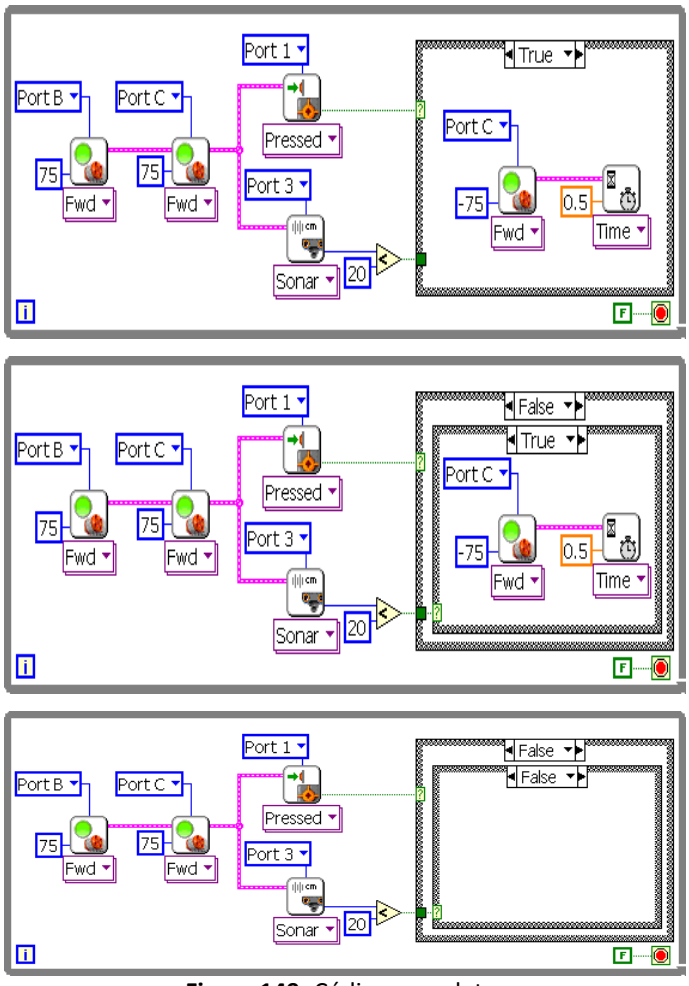



**Figura 149:** Código completo.

El diagrama de estado de la Figura 150, será similar al del ejercicio al final de esta unidad.

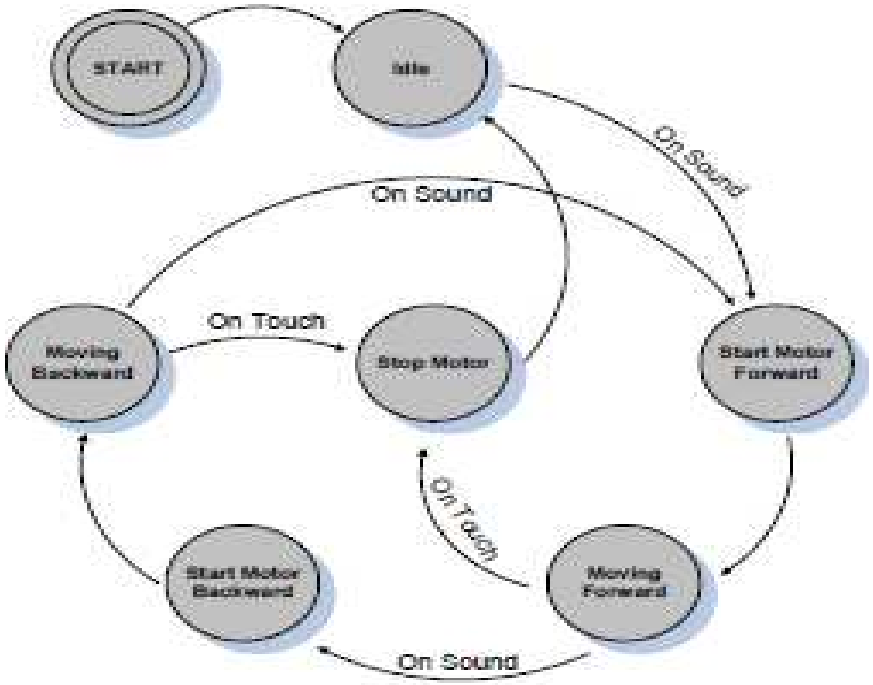



**Figura 150:** Diagrama de estados.

Debe prestar especial atención al caso “Stop Motor”. Al estado “Stop Motor” se puede llegar desde varios lugares: desde el estado “Moving Forward” y desde el estado “Moving Backward”. De esta manera, esta reutilizando ese estado. Pudo haber puesto el código del estado “Stop Motor” dentro del estado “Moving Forward” y del estado “Moving Backward”. Sin embargo, como la ha separado y ha hecho transiciones que pueden conducir a ese estado, realmente se puede beneficiar de reutilizar el código.

La ventaja de la reutilización de código es que se reduce el tiempo de programación y también se reduce el tamaño del código. Otra ventaja es que se más fácil de probar o depurar el código. Si tiene un problema con el estado “Stop Motor”, por ejemplo, se puede acceder y realizar cambios en una sola versión del mismo, y no tiene que hacer cambios dos veces en dos lugares diferentes.

También debe observar que en el diagrama de estados de la Figura 150, hay estados que solo realizan una acción. Si observa el diagrama de estado, inicia en el caso “START”, va al caso “Idle”, y cuando el sensor de sonido se activa, va al estado llamado “Star Motor Forward”. El estado “Star Motor Forward” va de inmediato al estado “Moving Forward”. El estado “Star Motor Forward” es un ejemplo de un estado que solo realiza una acción. En esencia, el estado “Star Motor Forward” enciende los motores y los deja encendidos, pero de inmediato salta al siguiente estado.

Va a crear un programa para ver los beneficios de los estados transitorios. Va a crear un programa en el que el robot se mueve hacia adelante hasta que el sensor de contacto es presionado. Va a crear el programa como una arquitectura de máquina de estados.

Antes de comenzar la dodificación, se describirá el programa y luego se creará el diagrama de estado.

Lo primero que quiere hacer al iniciar el programa es ponerlo en estado de reposo “Idle”. Por lo tanto, el diagrama de estado debe comenzar como la Figura 151.

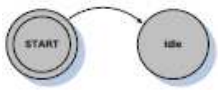



**Figura 151:** Inicio del programa.

Del estado inactivo “Idle”, necesita esperar hasta que se detecte un objeto, y luego quiere que los motores comiencen a girar. Va a crear una transición entre el estado “Idle” que se llama Object Detected y que apunte hacia el estado “Start Motor Forward”. Esto se muestra en le Figura 152.

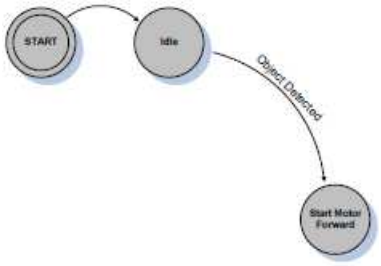



**Figura 152:** Transición de estado.

El estado "Start Motor Forward" encenderá los motores y de inmediato pasará al estado "Moving Forward". Este estado comprueba el sensor de contacto y si el sensor está presionado, va a enviar el programa al estado de reposo "Idle". El digarama de estado se muestra en la Figura 153.

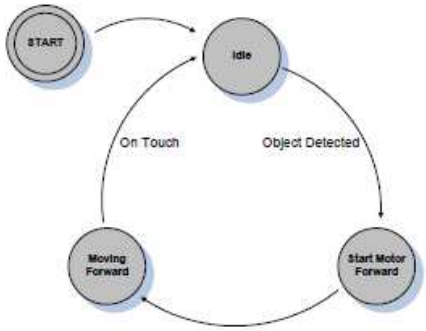



**Figura 153:** Diagrama de estado.

Ahora que tiene el diagrama de estados, puede crear facilmente el programa. Va a comenzar con un shift register. El shift register debe ser inicializado con una String Constant del estado "Idle". También es necesario colocar una estructura de casos dentro del ciclo While, y cablear el shift register de la izquierda con el selector de caso de la estructura de casos. El código debe verse similar al de la Figura 154.

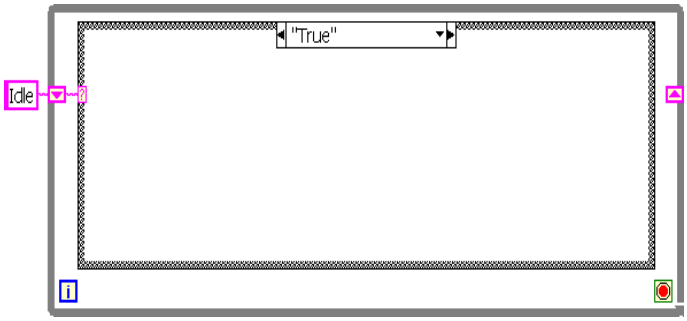



**Figura 154:** Inicio del programa.

Quiere que el caso “Idle” sea el predeterminado, por lo que puede pasar al caso por defecto de la estructura de casos, que actualmente se llama “False”, y cambiarle el nombre a “Idle”. En el estado “Idle”, quiere comprobar el sensor ultrasónico y quiere que el robot se detenga, por lo que colocará dos funciones de freno de motor, así como una función ¿Menor qué? Que compará la salida del sensor ultrasónico. Debe establecer el umbral del sensor ultrasónico que será de 20 cm. Es importante que también se cablee las terminales NXT de las tres funciones. El código debe ser similar al de la Figura 155.

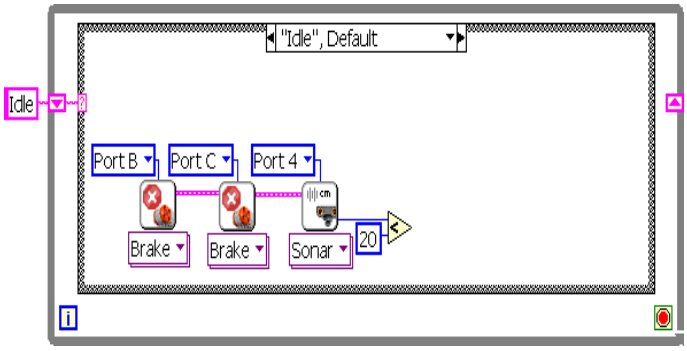
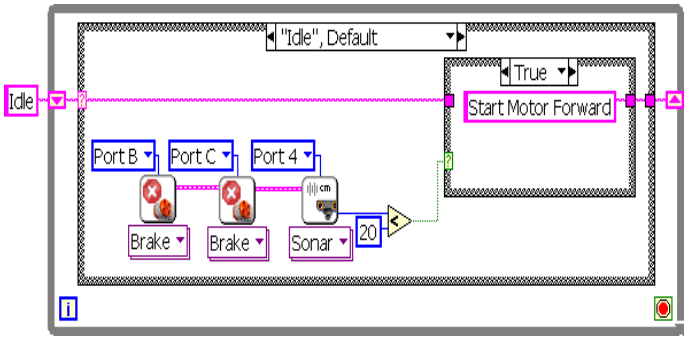



**Figura 155:** Estado “Idle”.

Ahora necesita una estructura de casos dentro del estado “Idle” para poder enviar el programa al estado “Start Motor Forward” si se detecta un objeto, o mantenerse en el estado “Idle” si no detecta nada. Cableará la salida de la función de comparación con el selector de caso de la estructura de casos anidada y creará una String Constant en el caso “True” con el fin de enviar al programa al siguiente estado. Para ello, va a cablear el shift register de la derecha. El código debe ser similar al de la Figura 156.



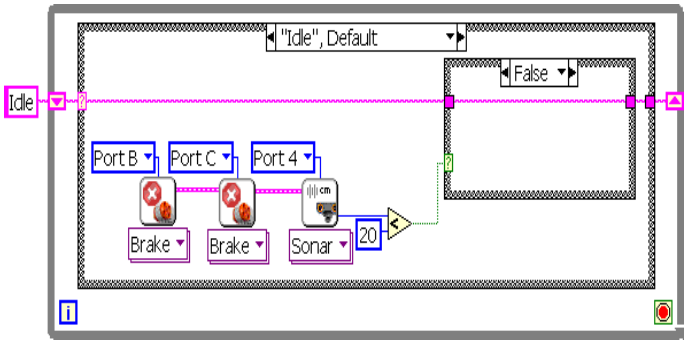



**Figura 156:** Estructura de caso anidado.

Vaya al caso “True” de la estructura de casos exterior y cambie el nombre del estado a “Start Motor Forward”. En este estado, todo lo que necesita hacer es arrancar los motores. También tiene que crear una String Constant, con el nombre “Moving Forward” y cablearla con el shift register de la derecha para que lo lleve al siguiente estado. El estado “Start Motor Forward” debe ser similar al de la Figura 157.

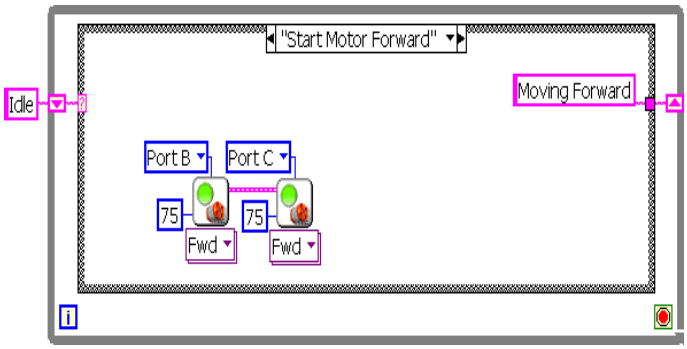



**Figura 157:** Estado “Start Motor Forward”.

Ahora puede crear el último caso, que es el estado “Moving Forward”. Puede dar clic derecho sobre la etiqueta de la estructura de casos y seleccionar Add Case After. Despues de cambiarle el nombre a “Moving Forward”, puede empezar a programar este caso. En este caso leerá el sensor de contacto. Si se pulsa, el programa, debe ir al estado “Idle” y si no, los motores deben permanecer girando. Agregar una función de sensor de contacto y cablearla al selector de caso de la estructura de casos. En el caso “True”, tiene que crear una String Constant con el nombre “Idle”. Cuando el sensor de contacto es presionado, el programa ejecutará el estado “Idle”. También tiene que cablear la String Constant al shift register del lado derecho. El código debe ser similar al de la Figura 158.

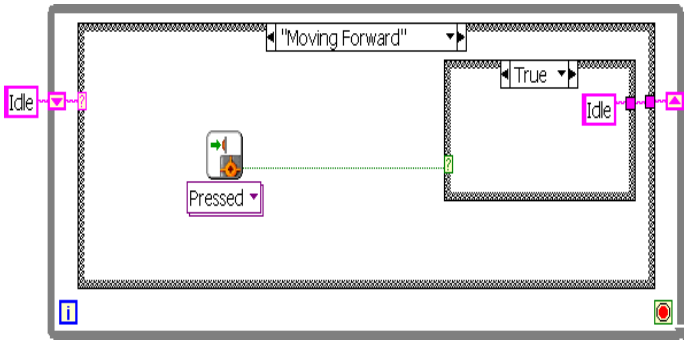



**Figura 158:** Estado "Moving Forward".

En el caso "False" de la estructura de casos anidada, quiere que los motores del robot permanezcan girando, lo que significa que quiere mantener el estado "Moving Forward". Para hacer esto, simplemente puede crear una String Constant, con el nombre "Moving Forward" y cablearla al shift register del lado derecho.

Todo lo que resta por hacer es cablear una constante "False" a la terminal condicional del ciclo While, con esto, puede ejecutar el programa.

**Taller:** Implemente el diagrama de estado mostrado en la Figura 159.

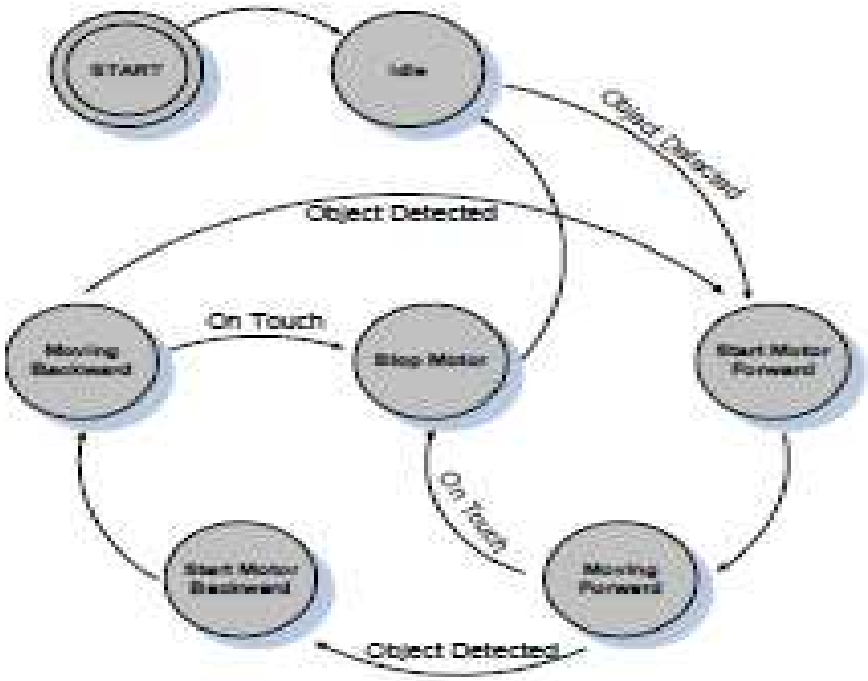



**Figura 159:** Diagrama de estado.

# PROGRAMACIÓN CON NXT 2.1 PROGRAMMING

El software NXT-G 2.1 versión educativo, es muy intuitivo permitiendo a alumnos y profesores navegar muy fácilmente por el entorno de programación visual. Está basado en iconos (bloques de programación) y además es capaz de programar el controlador NXT para que trabaje.

Desde la primera pantalla del software se puede acceder a dos guías animadas: "Getting Started" (primeros pasos) y "Software overview" (Visión del software):


**Figura 160:** Software NXT 2.1 Programming.

Los 41 bloques de programación (iconos) están situados en dos paletas distintas según el nivel de complejidad:

La paleta básica consiste sólo en los 7 bloques de programación básicos necesarios para controlar el robot de forma muy simple. Simplemente arrastra un bloque de programación de la paleta a la pantalla y tu robot se moverá.

La paleta completa permite acceder a todos los bloques de programación posibilitando una funcionalidad de programación muy avanzada a la vez que intuitiva.

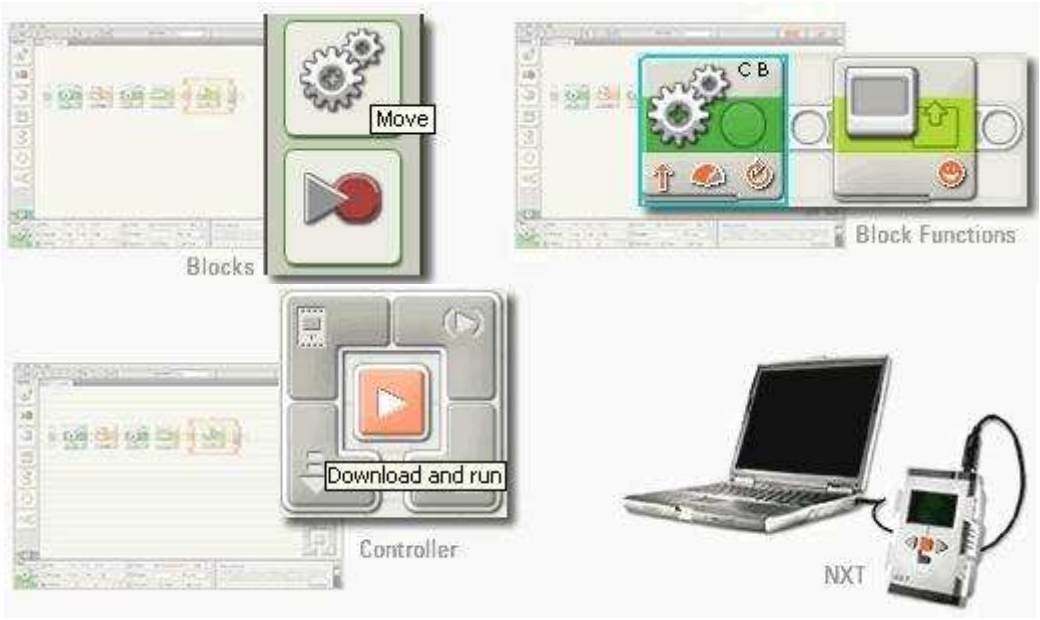



**Figura 161:** Bloques del software.

La nueva versión del software NXT-G 2.1 es compatible con Mac OS X y Windows XP- Vista- 7 y tiene nuevas funcionalidades de "Data Logging" (adquisición programada de datos) e incluye un visor de gráficos que hace sencillo analizar la información obtenida de los sensores como el nuevo sensor de temperatura. Las siguientes Figuras muestran el entorno del software NXT 2.1 Programming.

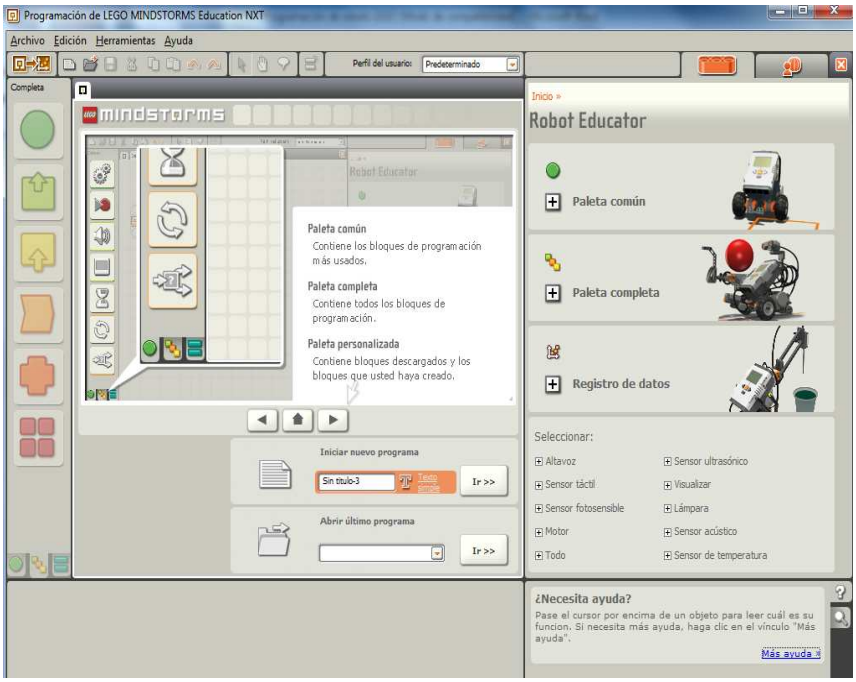
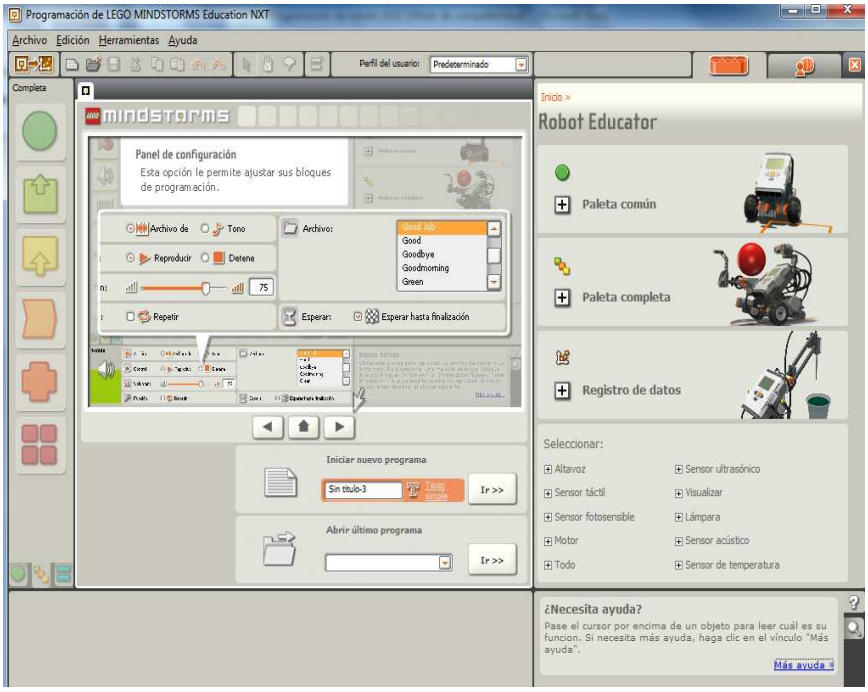



**Figura 162:** Ubicación de las paletas.




**Figrua 163:** Panel de configuración.

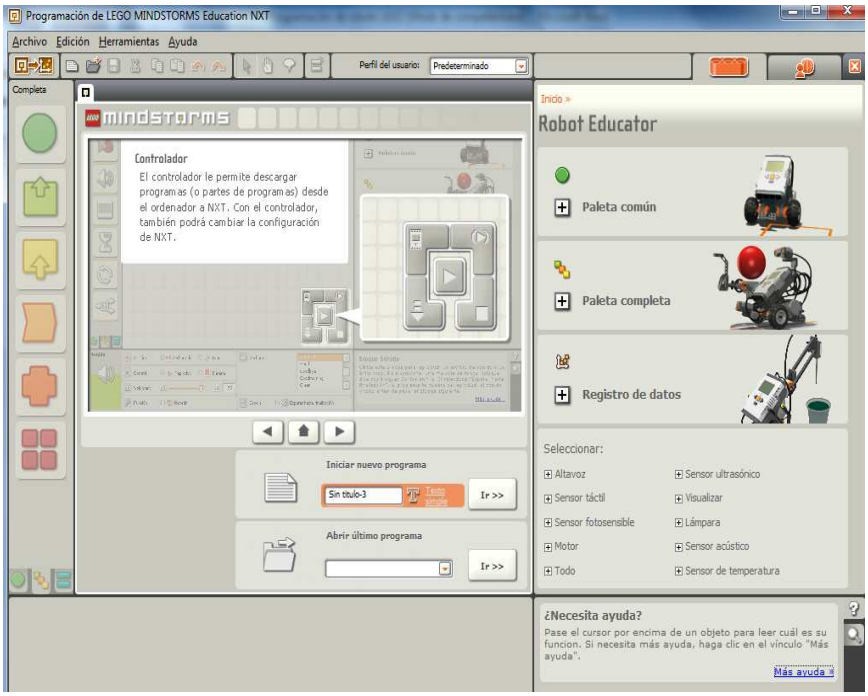



**Figura 164:** Controlador.

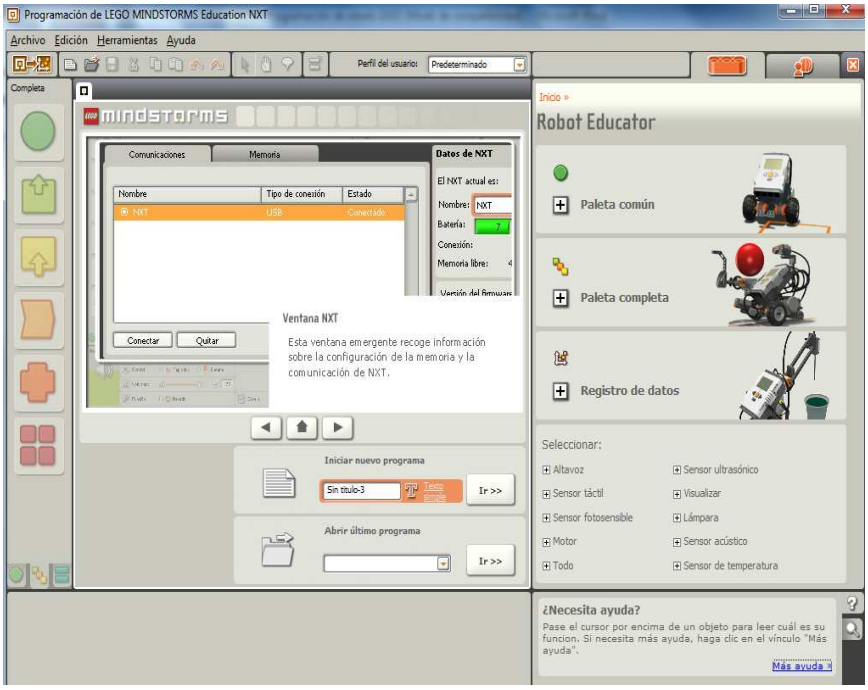



**Figura 165:** Ventana NXT.

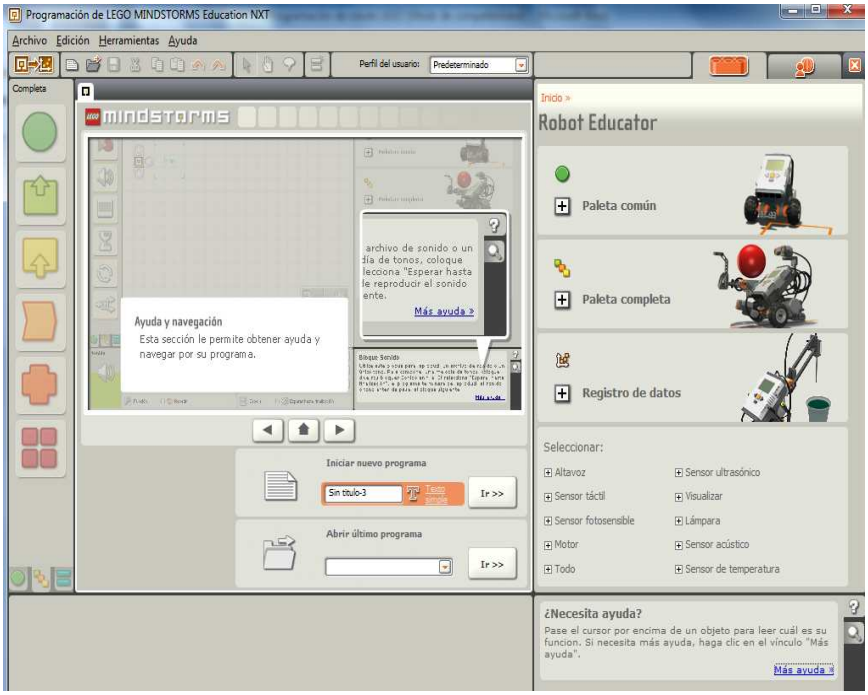



**Figura 166:** Ayuda emergente.

El "Robot Educator" es una completa guía de programación del entorno visual NXT-G 2.1 que contiene una serie de 46 tutoriales animados de distintos niveles de dificultad que resuelven "retos" o ejercicios para ayudar a los estudiantes (y profesores) a conocer como construir y programar un robot totalmente funcional.

Primero verás de forma animada un vídeo del robot actuando que constituye un "reto" o ejercicio de lo que tu robot tiene que ser programado para realizarlo.

La guía de construcción te permitirá construir el robot requerido para resolver el "reto".

Finalmente la guía de programación te propone un programa que resuelve el problema, que puede ser utilizado como inspiración o en caso de que tu programa no funcione como esperas.

Todo ello garantiza que el aprendizaje con el producto sea mucho más progresivo, rápido, divertido y sencillo. Si el usuario final del robot es un niño de menos de 12 años, esto será de una gran ayuda.


**Figura 167:** Pantalla de los retos del Robot Educator.

# MUEVE EL ROBOT

**Introducción.**

Antes de comenzar, es necesario mencionar que el robot con el que se trabajará en los capítulos de programación con NXT 2.1 Programming es el que se muestra en la Figura 168.

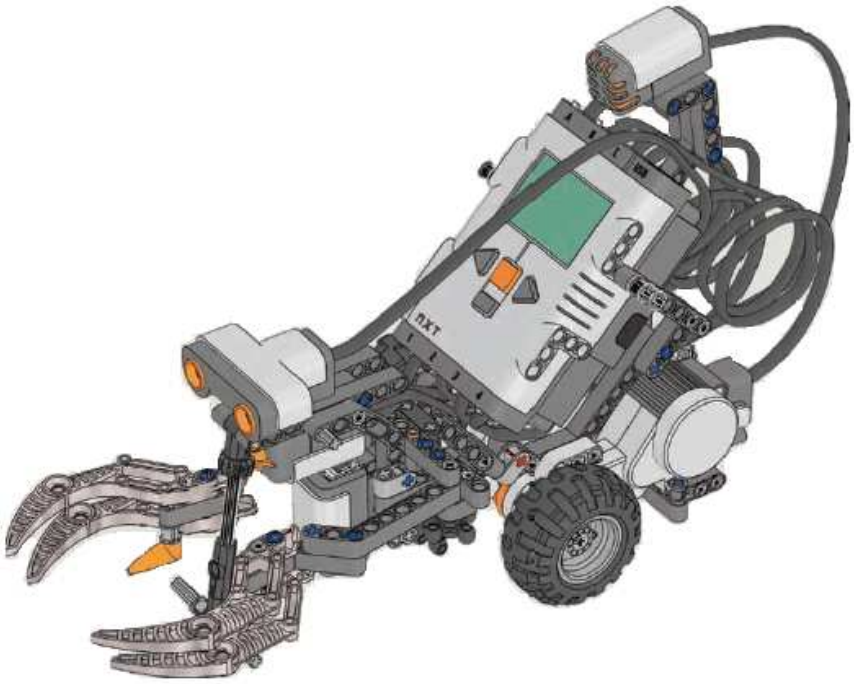


**Figura 168:** Robot que se utilizará.

Abra el software Mindstorms NXT en su ordenador.


**Figura 169:** Pantalla de inicio del software.

Ir a *Archivo>> Nuevo* para crear un nuevo proyecto.

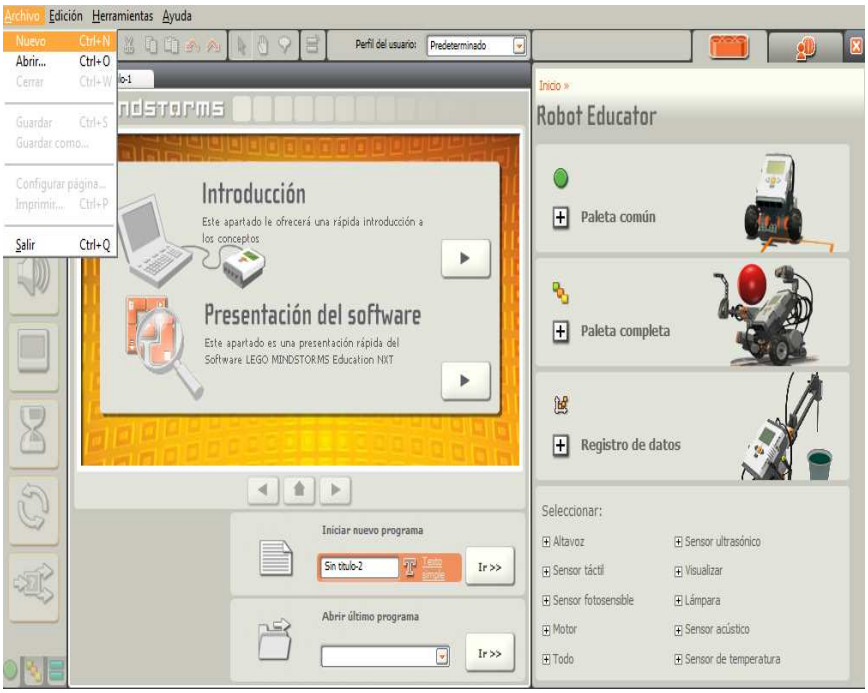



**Figura 170:** Arhivo >> Nuevo.

Pulse el botón naranja, para encender el Mindstorms NXT.

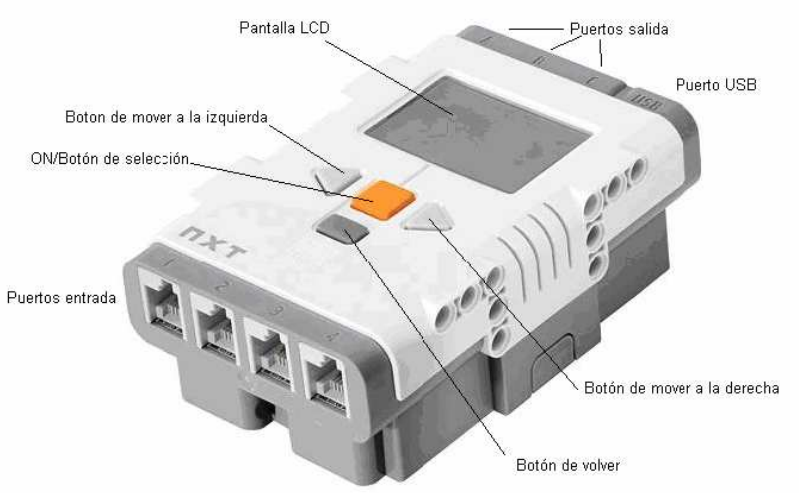



**Figura 171:** Controlador NXT.

Mediante un cable USB, conecte el controlador NXT a un puerto USB en su ordenador.

Detectar el robot haciendo clic en el botón de la ventana NXT en la parte inferior derecha de la pantalla como se muestra en la Figura 172.

**Figura 172:** Controlador.

Una ventana de diálogo de conexión se abre. Haga clic en el botón *Buscar* para detectar el robot que se conecta a la PC. El nombre de su robot debe aparecer en la lista en el lado izquierdo del cuadro de diálogo.

Haga clic en el robot, a continuación, haga clic en *Conectar*. Información detallada sobre el robot, incluyendo la vida de la batería, el espacio de almacenamiento libre en el bloque NXT, y la versión del firmware instalado debe aparecer en el lado derecho. Esto indica que el software ya está conectado. Esto se muestra en la Figura 173.

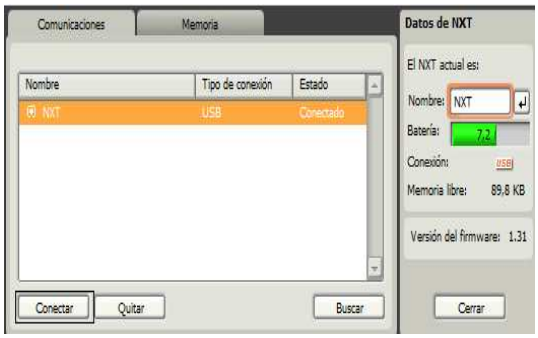



**Figura 173:** Comunicación.

Haga clic en la ficha de memoria en la parte superior de esta ventana de diálogo. Esta pantalla describe los programas que están cargados actualmente en el bloque NXT.

Cierre el cuadro de diálogo, y vaya a la paleta completa en la parte inferior de la barra de paletas en el lado izquierdo de la interfaz del software NXT 2.1 Programming como se muestra en la Figura 174.

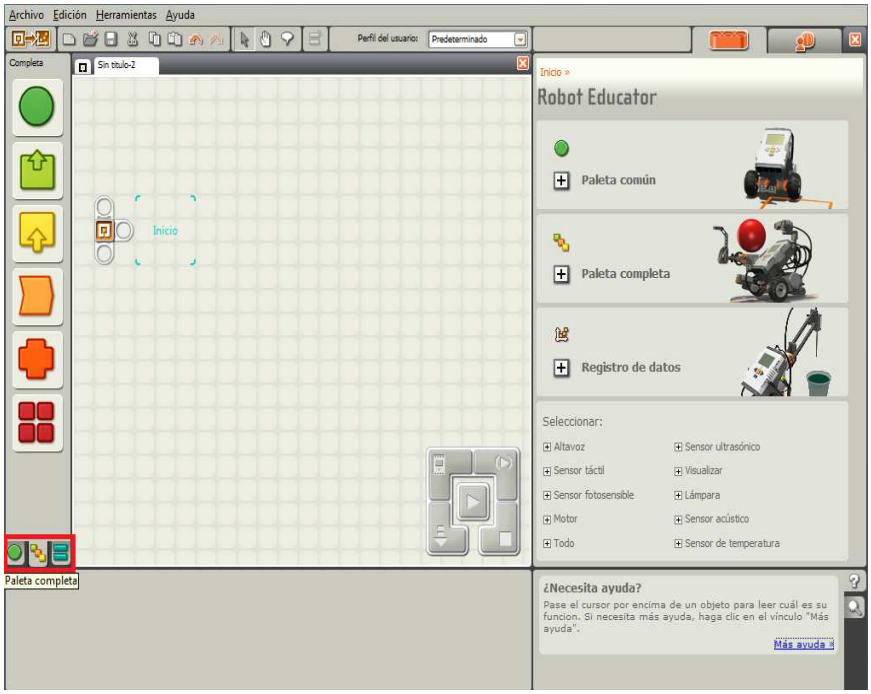



**Figura 174:** Paleta completa.

Coloca un bloque de motor que se encuentra en la sub-paleta Acción, tal y como se muestra en la Figura 175.

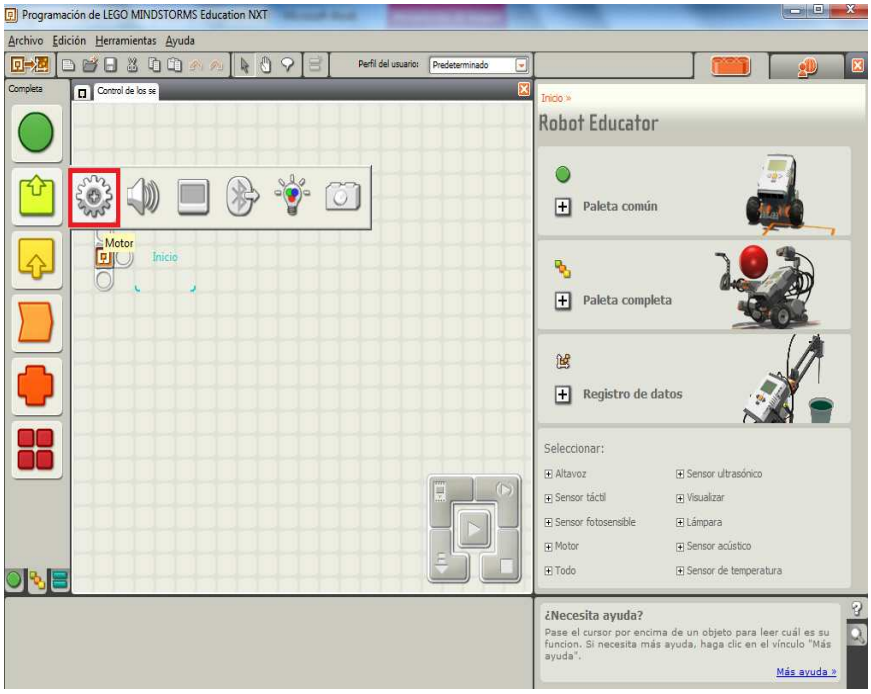



**Figura 175:** Sub-paleta Acción.

Haga clic en el bloque de motor. La sección de configuración de la Figura 176 debe aparecer en la parte inferior de la interfaz del software NXT.

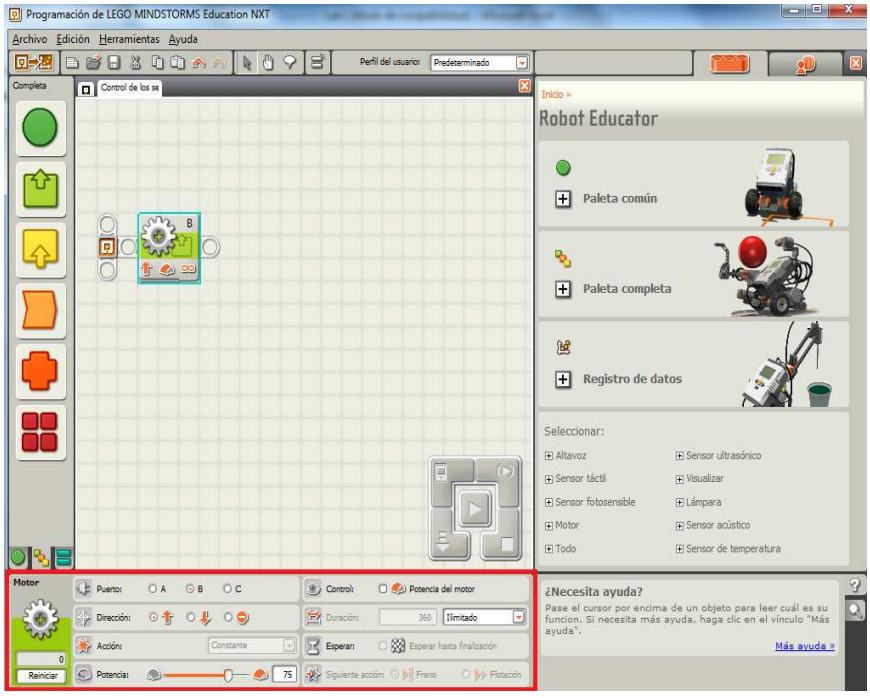



**Figura 176:** Panel de configuración.

**Primer programa.**

Crear un nuevo perfil:

Selecciona *Edición >> Administrar* perfiles de la barra de menú de la parte superior de la ventana, como se muestra en la Figura 177.

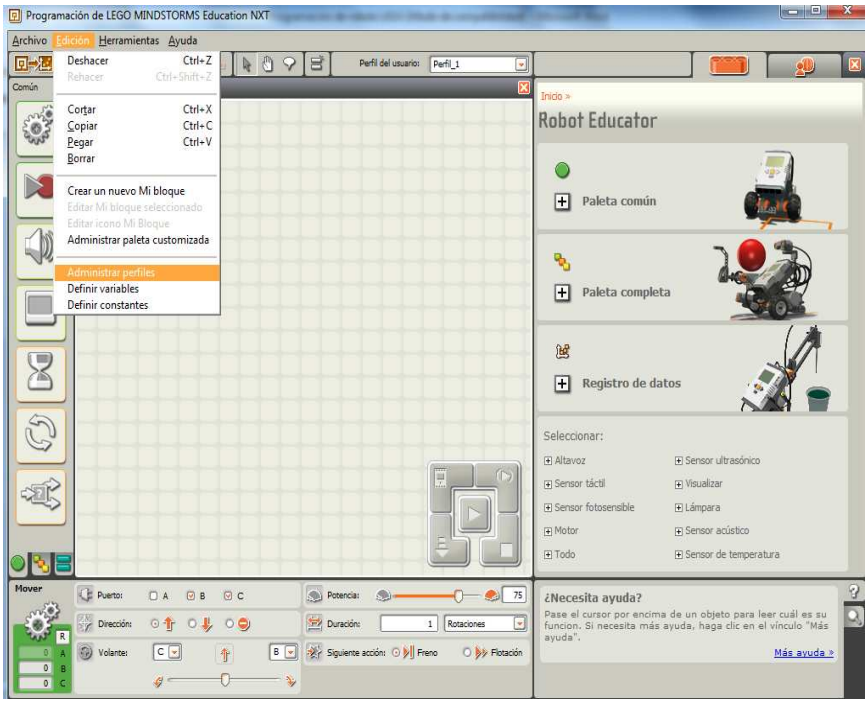



**Figura 177:** Administrar perfiles.

Haga clic en *Crear*, introduzca un nombre al perfil, haga clic en *Cerrar* para crear el nuevo perfil. Cada vez que regrese al NXT 2.1 Programming, usted puede seleccionar este perfil para volver a todas las configuraciones.


**Figura 178:** Crear perfil.

Selecciona tu perfil del desplegable *Perfil del usuario*. Como se muestra en la Figura 179.

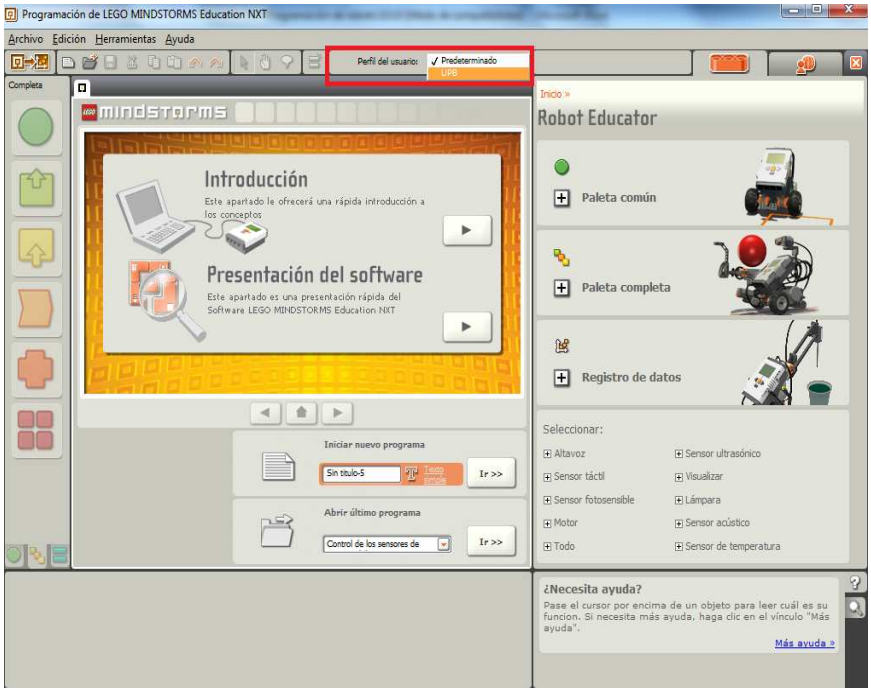



**Figura 179:** Selección del perfil.

Mueve el robot:

Selecciona *Archivo >> Nuevo*, o *Ctrl + N* para crear un programa nuevo.

Selecciona el bloque *Mover* de la paleta *Común* y colocalo en el haz de secuencia.

Este bloque es similar al bloque del motor. Sin embargo, tiene la capacidad de configurar varios motores para moverlos.

Para más información sobre cómo utilizar este bloque y otros, haga clic en el enlace *Más ayuda* que se muestra en la esquina inferior derecha del software NXT para acceder a la ayuda. Asegúrese de que el bloque del que desea obtener más ayuda está seleccionado.

Haga clic en el bloque *Mover* para que aparezca en la parte inferior del software la ventana de configuración. Configure este bloque como se muestra en la Figura 180.

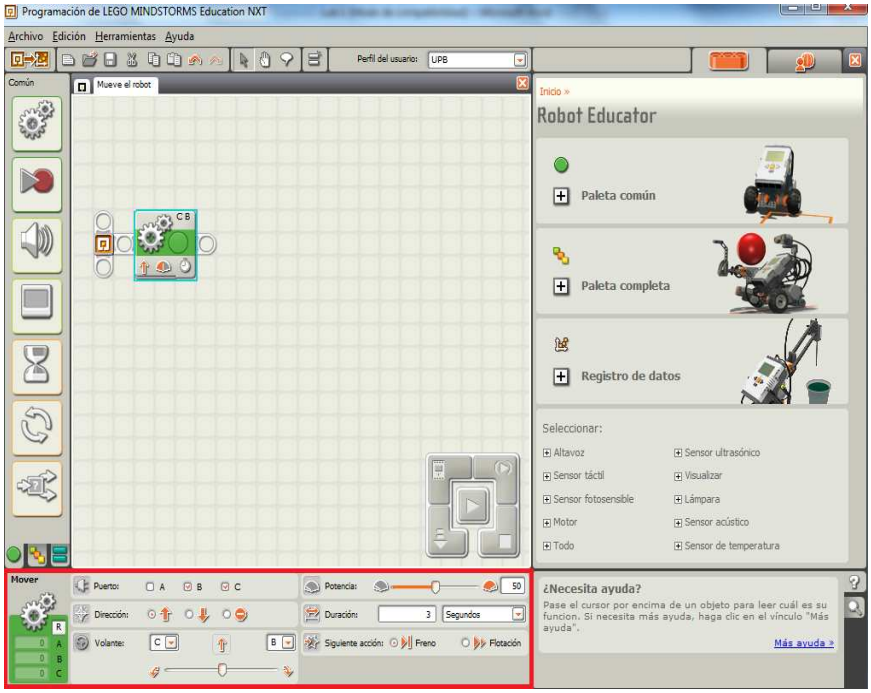



**Figura 180:** Configuración del bloque Mover.

Esta configuración indicará al robot seguir adelante durante 3 segundos a una potencia de 50%, y luego frenar cuando llegue a su destino. Tenga en cuenta que tiene un nivel de potencia del 50% en lugar del 100%. En general, es una buena práctica para especificar los valores conservadores de energía para los motores para ayudar a prolongar la vida en el robot.

Coloque otro bloque *Mover* en el haz de la secuencia, y configurarlo para que mueva el robot hacia atrás como se muestra en la Figura 181.

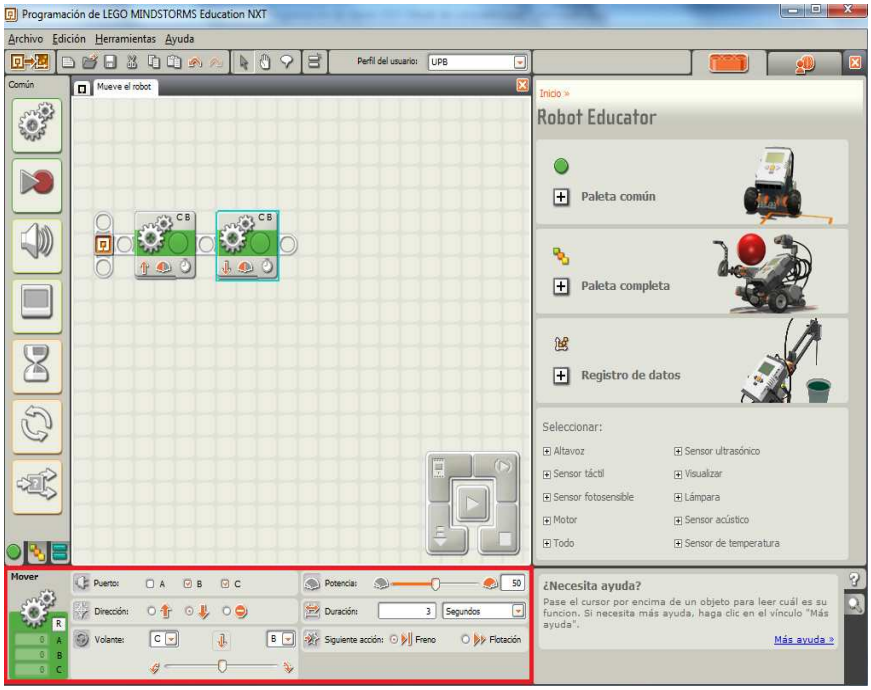



**Figura 181:** Configuración del segundo bloque Mover.

Una vez que el robot completa este bloque, regresará a su posición inicial.

Coloque a continuación otro bloque *Mover* que haga que el robot gire a la izquierda mediante la configuración que se muestra en la Figura 182.

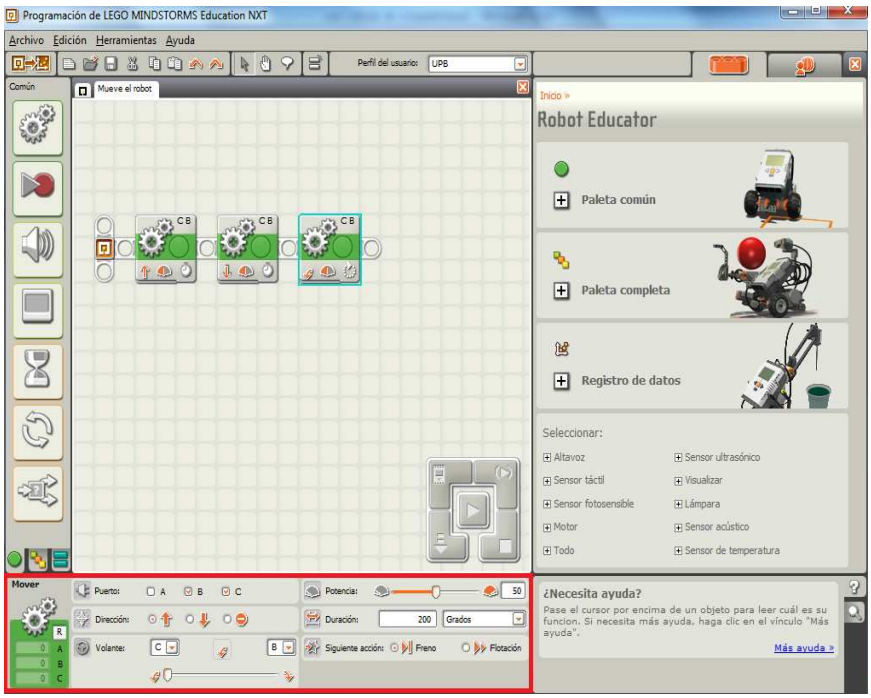



**Figura 182:** Configuración del tercer bloque Mover.

Tenga en cuenta que se ha seleccionado una duración de 200 grados en lugar de segundos.

Coloque y configure otro bloque *Mover* para mover el robot de nuevo a su posición inicial una vez más. Como se muestra en la Figura 183.

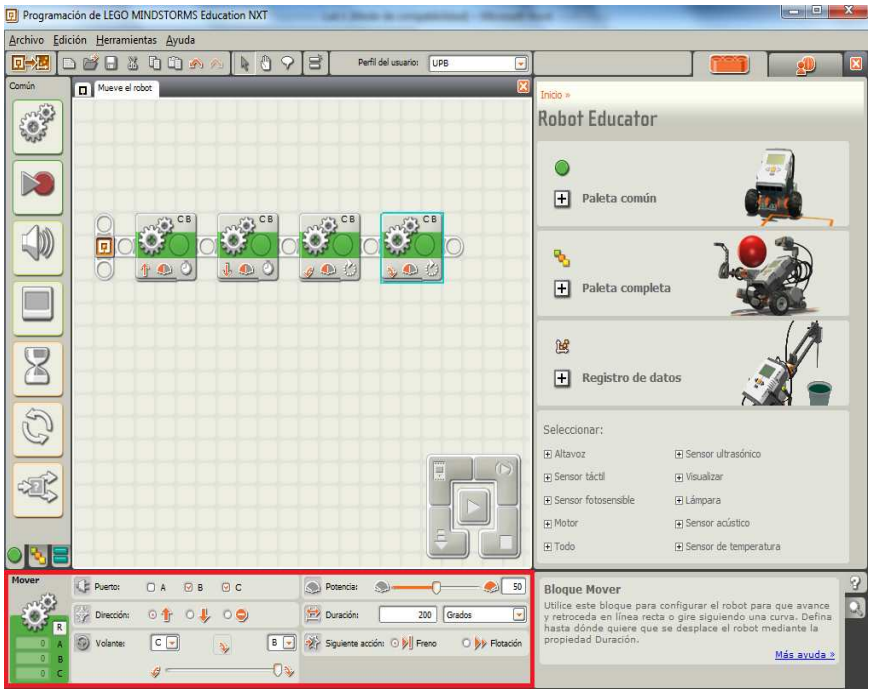



**Figura 183:** Configuración del cuarto bloque Mover.

**Consejo:** puede copiar y pegar bloques NXT mediante la selección de los objetos a copiar, y seleccionar *Edición >> Copiar*. De igual forma puede pegar los objetos seleccionando *Edición >> Pegar*. Los bloques pegados tendrán que ser colocados en el haz de la secuencia para que se active de nuevo.

Coloque y configure otro bloque *Mover*, el robot gira a la izquierda un ángulo ligeramente inferior al bloque anterior como se muestra en la Figura 184.

Tenga en cuenta que la barra de desplazamiento de dirección ha sido movido a la izquierda cinco espacios del centro, la duración se ajusta a 4 rotaciones, y la potencia se ha aumentado a 90%.

Coloque otro bloque *Mover*, y configurarlo para que el robot vuelva a su posición inicial. Como se muestra en la Figura 185.

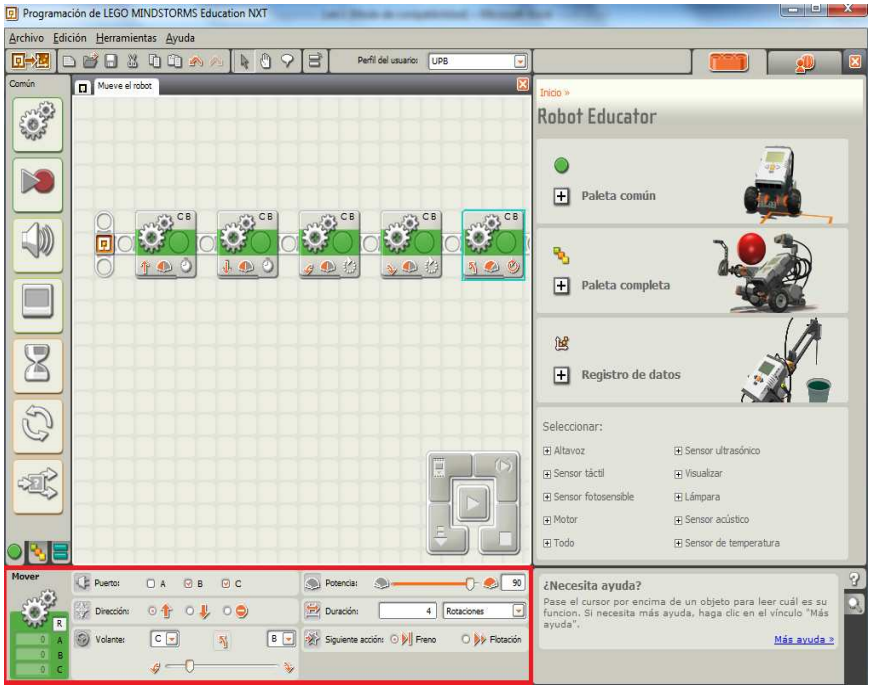



**Figura 184:** Configuración del quinto bloque Mover.

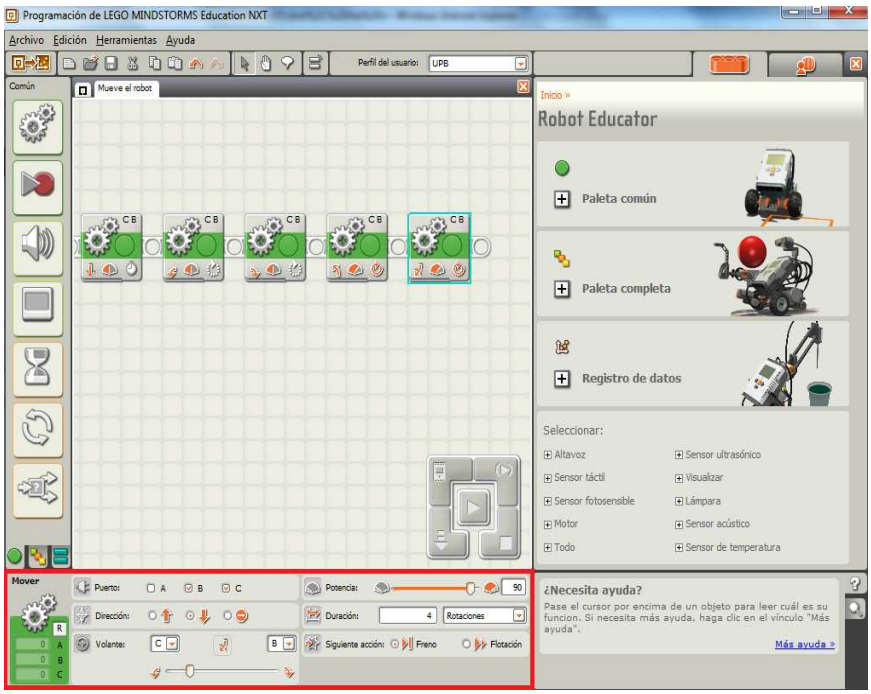



**Figura 185:** Configuración del sexto bloque Mover.

Guarde su programa: Vaya a *Archivo >> Guardar como*, tal y como se muestra en la Figura 186.

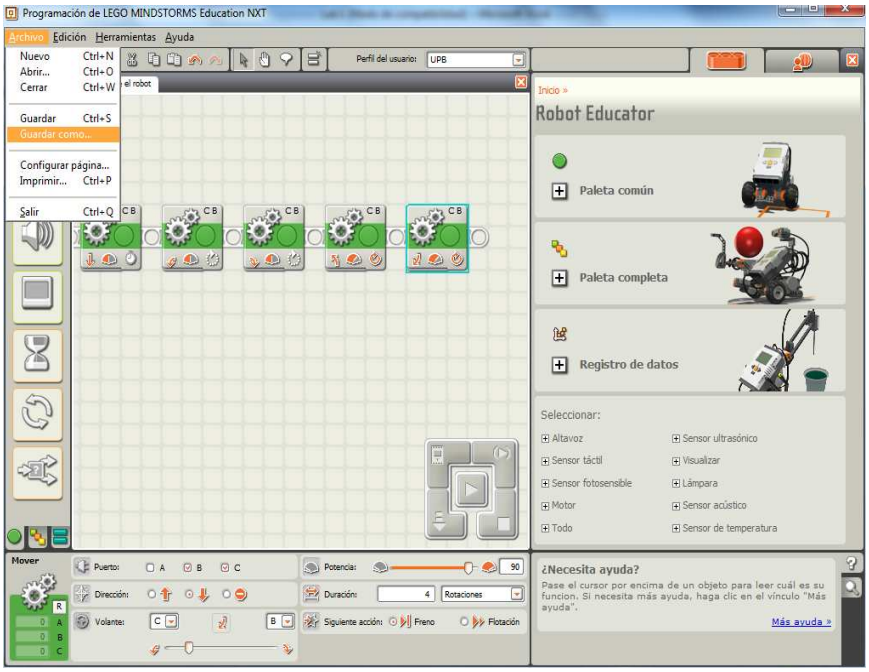



**Figura 186:** Guardar el programa.

De clic en *Examinar,* cree una nueva carpeta y nombrela *"Ejercicio 1"*. Como se muestra en la Figura 187.

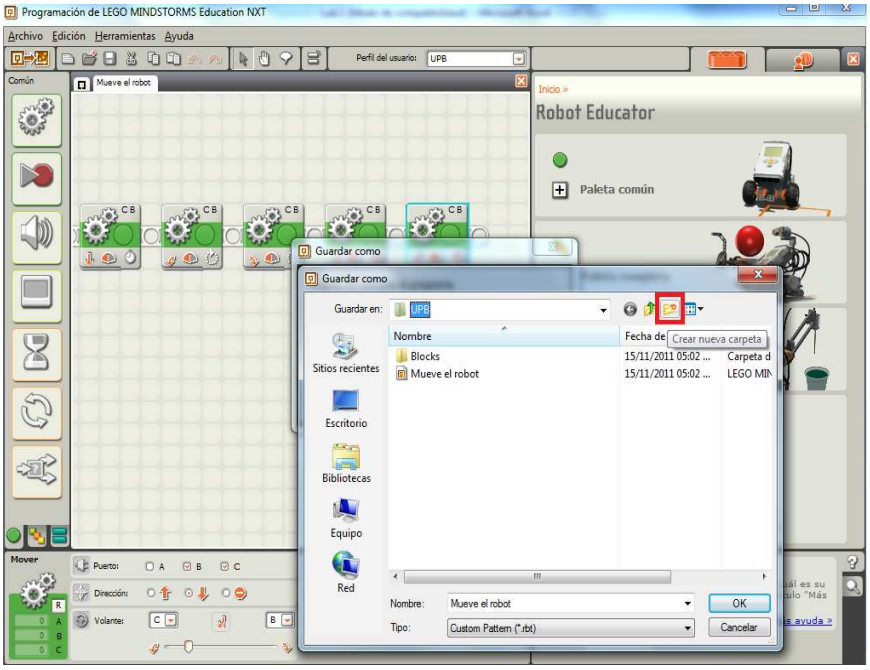



**Figura 187:** Crear nueva carpeta.

Escriba “Mueve el robot” como nombre del programa, y haga clic en *OK* y haga clic en *Guardar.*

Descarga y ejecuta el programa:

Con el robot conectado y encendido, haga clic en el botón Descargar que se encuentra en la esquina inferior derecha de la interfaz del software NXT como se muestra en la Figura 188.

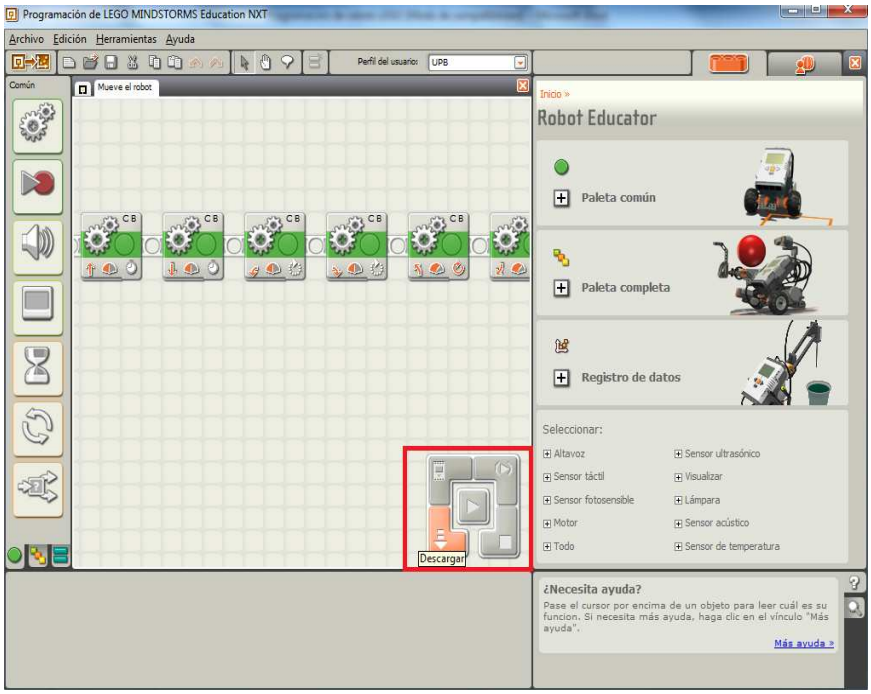



**Figura 188:** Descargar el programa al controlador del robot.

Esto descargará el programa, junto con otros archivos de soporte para el controlador NXT. Si la descarga se hizo correctamente, aparecerá una ventana de estado como la que se muestra en la Figura 189.

Si la descarga no se pudo completar usted observará una ventana de estado como la que se muestra en la Figura 190.


**Figura 189:** Descarga exitosa.

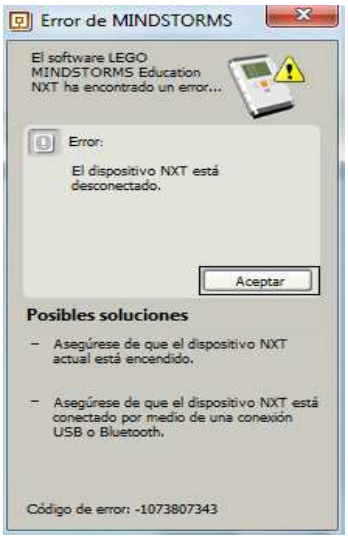



**Figura 190:** Descarga fallida.

Una vez que la descarga se realiza correctamente, desconecte el robot de su ordenador, y coloque en algún lugar en el piso donde no haya objetos en el camino.

Ejecute el programa en el robot navegando en Mis Archivos >> Archivos de software. Seleccione el programa y ejecutelo.

**Taller:** Mueva el robot en una trayectoria.

El objetivo de este taller es mover el robot en un patrón con forma de ocho, como se muestra en la Figura 191 utilizando una serie de bloques de movimiento.

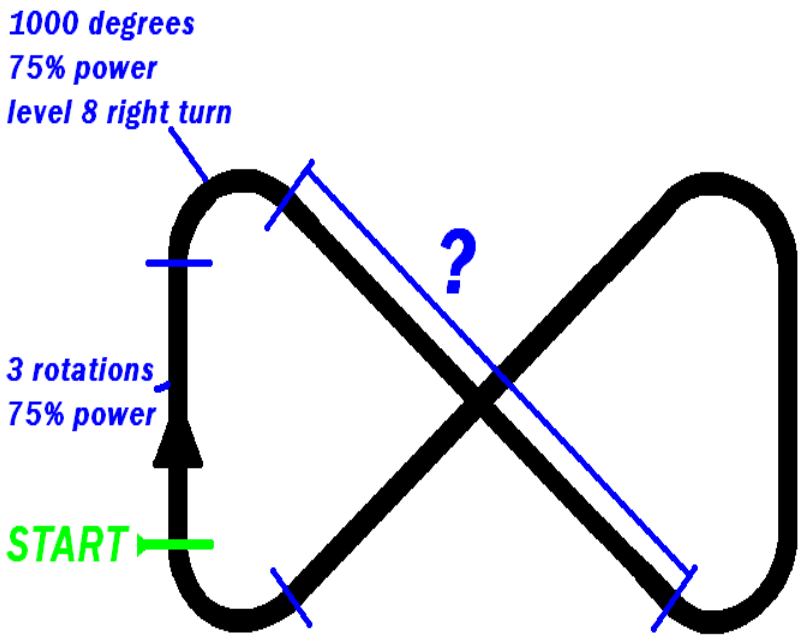



**Figura 191:** Patrón con forma de ocho.

Los siguientes criterios deben cumplirse para acreditar el taller:

1) El robot debe comenzar con dos bloques de movimiento que tienen los siguientes valores:

El bloque Mover # 1:


**Figura 192:** Bloque Mover #1.

El bloque Mover # 2:


**Figura 193:** Bloque Mover #2.

Tenga en cuenta que la acción seleccionada para estos dos bloques es la de *Flotación*. Esto es para asegurar un movimiento suave en la transición entre bloques. Usted puede tener el bloque de movimiento final configurado para frenar si lo desea.

2) El robot debe volver a la misma ubicación y orientación de donde comenzó.

3) ¡Una vez que se inicia el programa en el robot, no se le permite tocar el robot hasta que termine!

4) El robot debe viajar en el patrón con forma de ocho similar al que se muestra en la Figura 191.

# DETECTAR Y EVITAR OBSTACULOS.

**Monitoreando el sensor de contacto y el sensor ultrasonico.**

Abre el software LEGO Mindstorms NXT en su ordenador.

Usando un cable USB, conecte su robot a un puerto USB de su ordenador.

Presione el botón *Encender* de su robot.

Cree un nuevo proyecto seleccionando *Archivo >> Nuevo.*

Detecte el robot haciendo clic en el botón *Ventana NXT* que se ubica en la parte inferior derecha de la pantalla. Haga clic en su robot y clic en *Conectar.*

Vaya a la paleta completa en la parte inferior de la barra de paletas en el lado izquierdo de la interfaz del software NXT 2.1 Programming.

De la sub-paleta *Sensor*, coloque un bloque *Sensor de contacto* en el haz de secuencia.

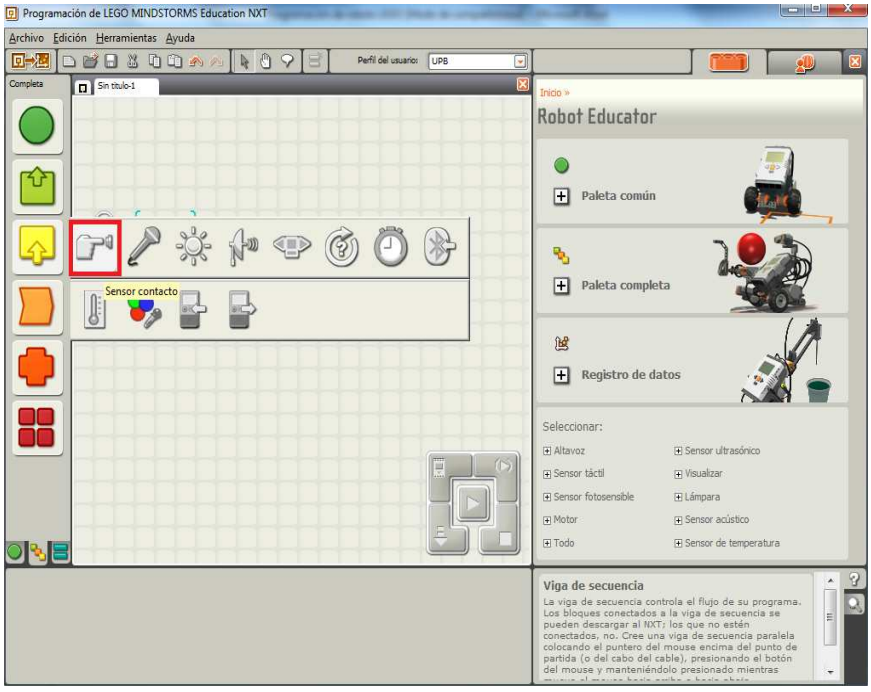



**Figura 194:** Ubicación del sensor de contacto en la paleta completa.

Haga clic sobre el Sensor de contacto, y observará el estado actual del sensor en la esquina inferior izquierda de la pantalla. Presione el sensor de contacto en su robot y observe que el estado del sensor cambia.

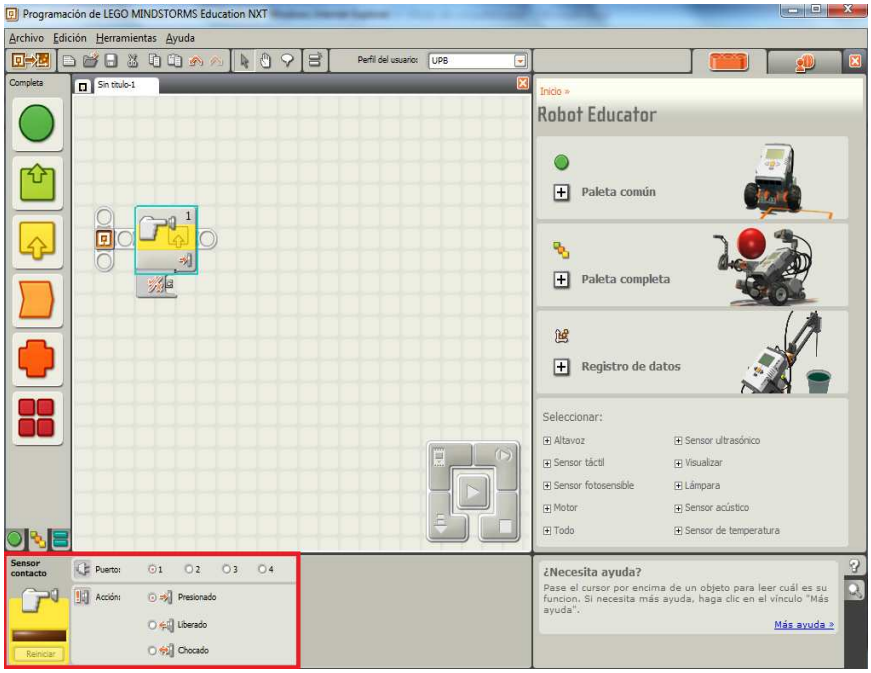



**Figura 195:** Estado del sensor de contacto.

De la sub-paleta *Sensor*, coloque un bloque sensor ultrasónico en el haz de secuencia.

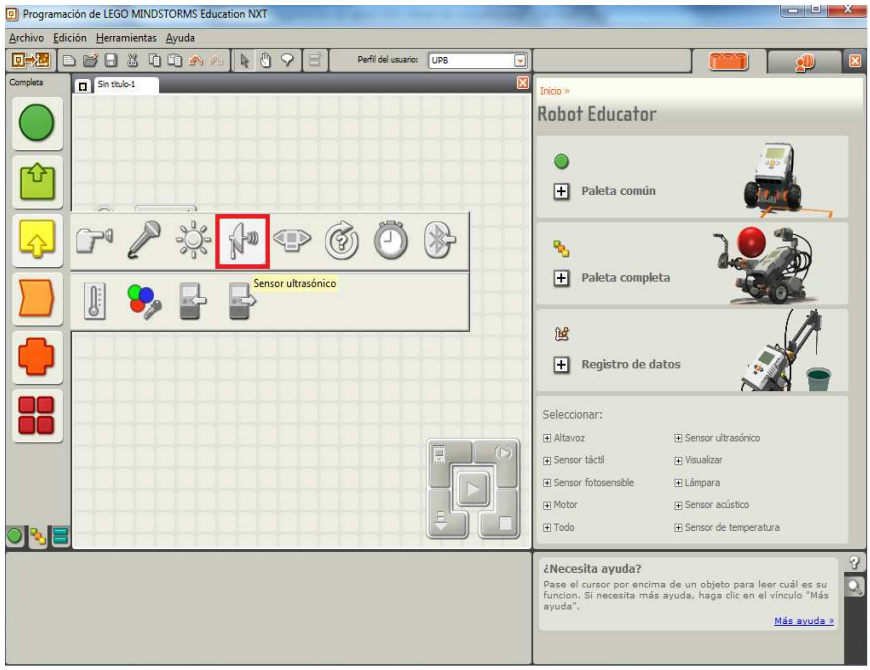



**Figura 196:** Ubicación del sensor ultrasónico en la paleta completa.

Haga clic sobre el Sensor ultrasónico. Observe que los valores en la ventana del sensor cambian al mover objetos en la parte frontal del sensor.

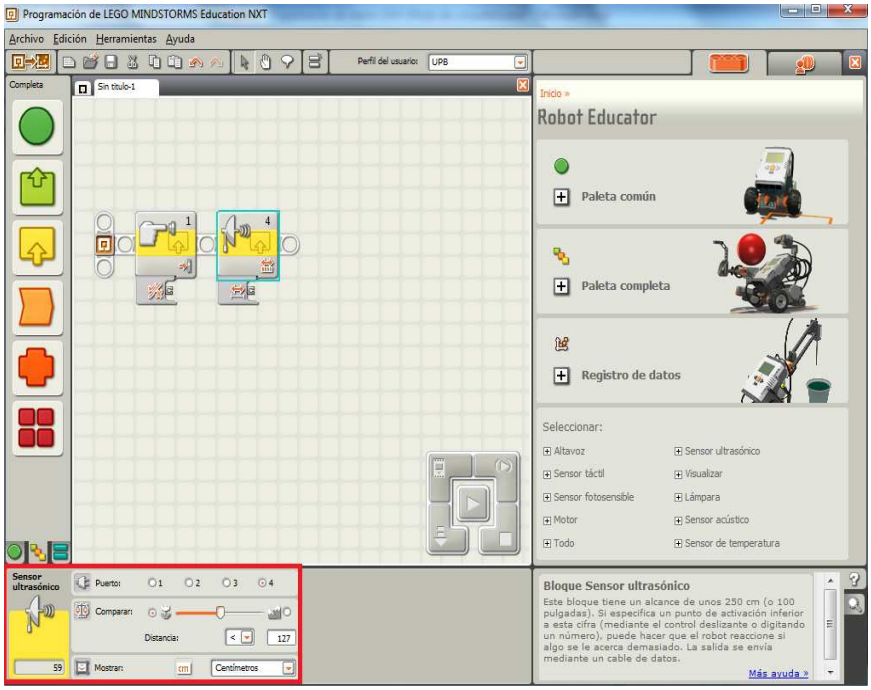



**Figura 197:** Estado del sensor ultrasónico.

Cierre el programa sin guardar cambios.

**Segundo programa.**

En el cuadro desplegable *Perfil de usuario*, seleccione su perfil.

Seleccione *Archivo >> Nuevo* para crear un nuevo programa.

Guarde el programa como “Detectar y evitar obstaculos”

Cambie la vista de la paleta completa. Haga clic en la paleta común y luego seleccione el bloque *Mover.*Coloque el bloque *Mover* en el haz de la secuencia, y configurelo como se muestra en la Figura 198.

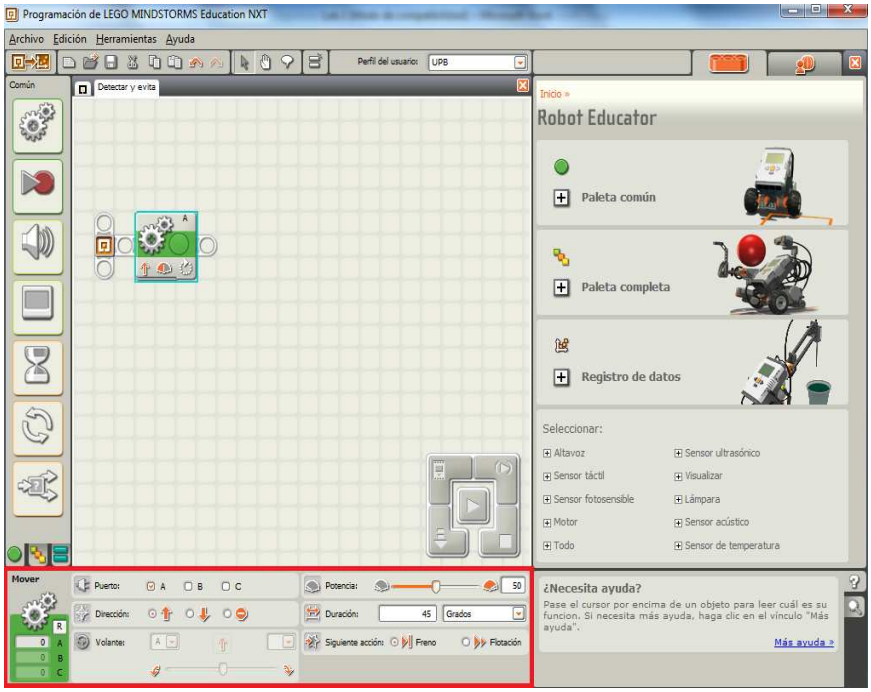



**Figura 198:** Configuración del bloque Mover.

Este bloque estará al mando para abrir parcialmente las garras del robot.

Seleccione el bloque *Bucle*. Coloque el bloque *Bucle* en el extremo del haz de la secuencia, y dejarlo en su configuración por defecto, como se muestra en la Figura 199.

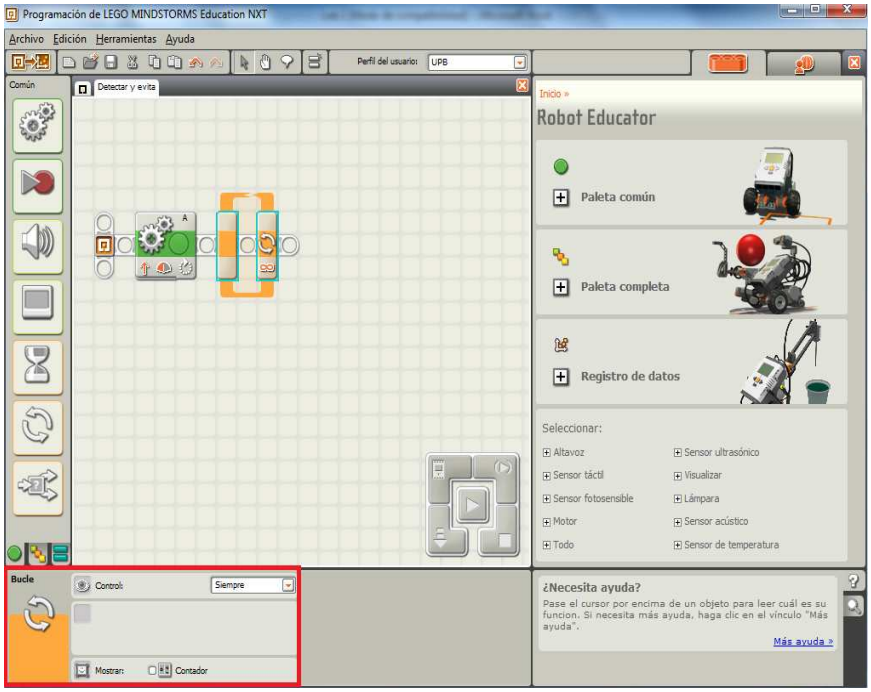



**Figura 199:** Configuración del bloque Bucle.

Seleccione el bloque *Bifurcación.* Coloque el bloque dentro de la estructura *Bucle* en el haz de secuencia. Haga clic en el bloque *Bifurcación,* y configurelo como se muestra en la Figura 200.

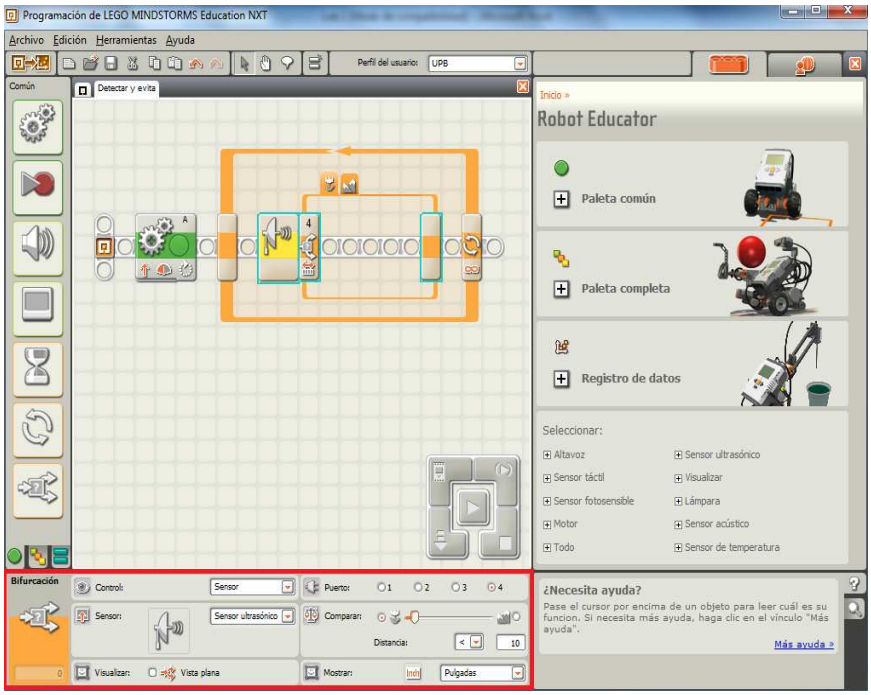



**Figura 200:** Configuración del bucle Bifuración.

Coloque otro bloque *Mover* dentro del caso “Verdadero” del bloque *Bifurcación.*Configure el nuevo bloque *Mover* como se muestra en la Figura 201.

Seleccione un bloque *Espera >> Distancia,* coloquelo a la derecha del bloque *Mover* que acaba de agregar y configurelo como se muestra en la Figura 202.

Cambiese al caso “Falso” del bloque *Bifurcación.* Coloque otro bloque Mover dentro de este caso y configurelo para que se mueva hacia adelante a una potencia del 50%, como se muestra en la Figura 203.

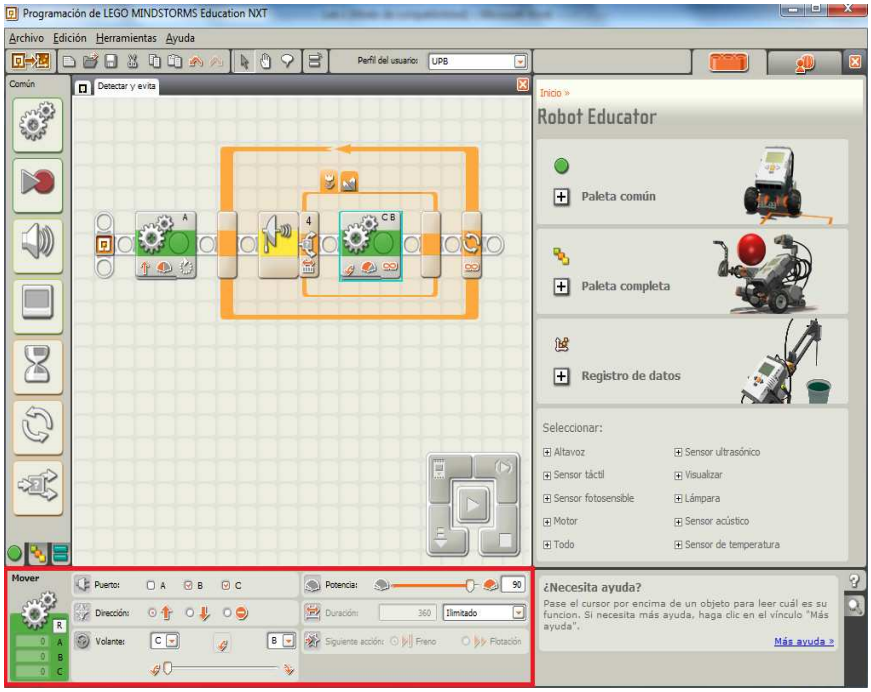



**Figura 201:** Configuración del bloque Mover.

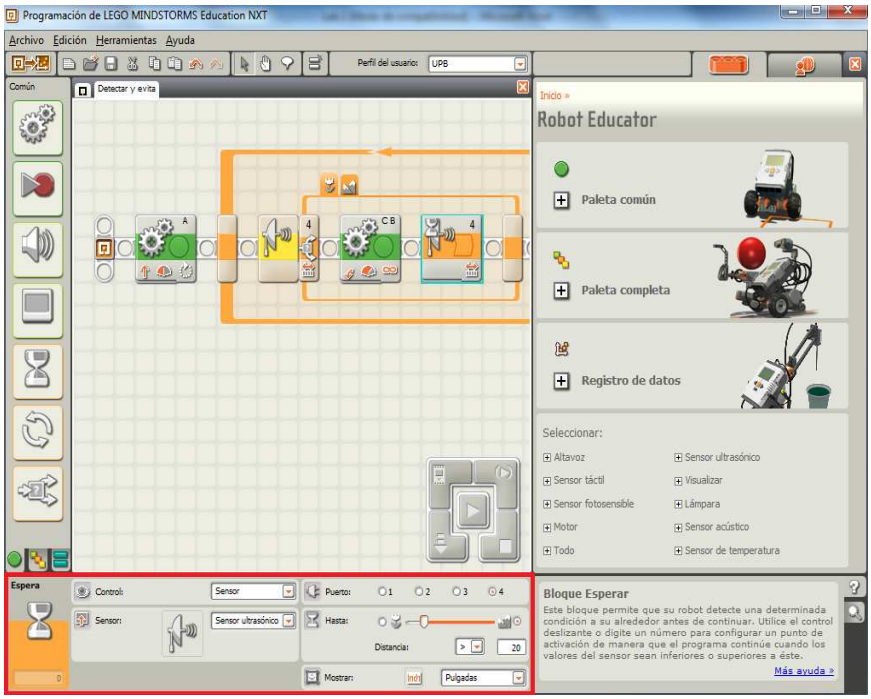



**Figura 202:** Configuración del bloque Espera >> Distancia.

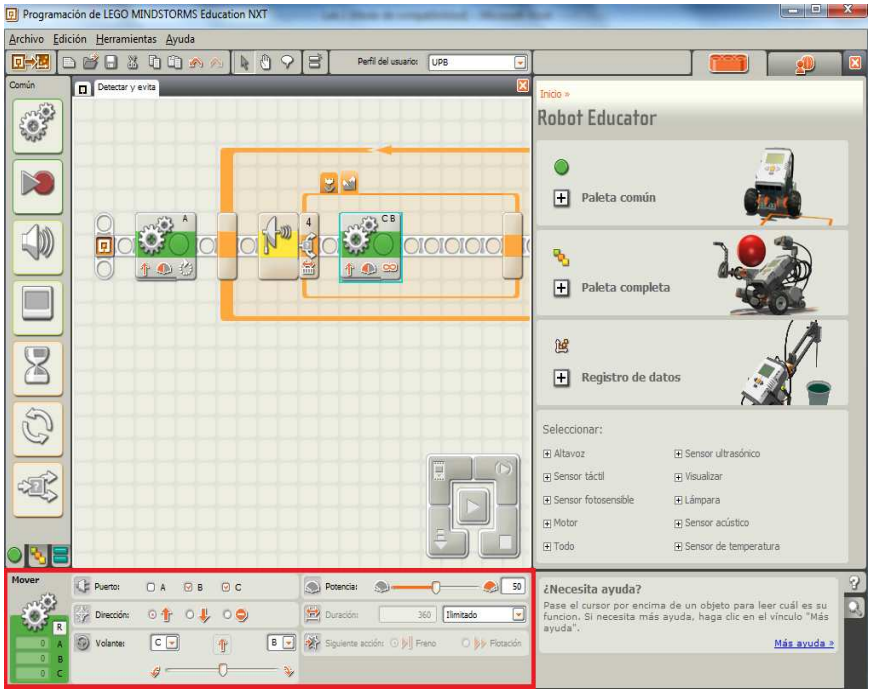



**Figura 203:** Configuración del bloque Mover.

Coloque otro bloque *Bifurcación* dentro del bloque *Bucle,* pero despues del primer bloque *Bifurcación* sobre el haz de secuencia. Configure el bloque *Bifurcación* como se muestra en la Figura 204.

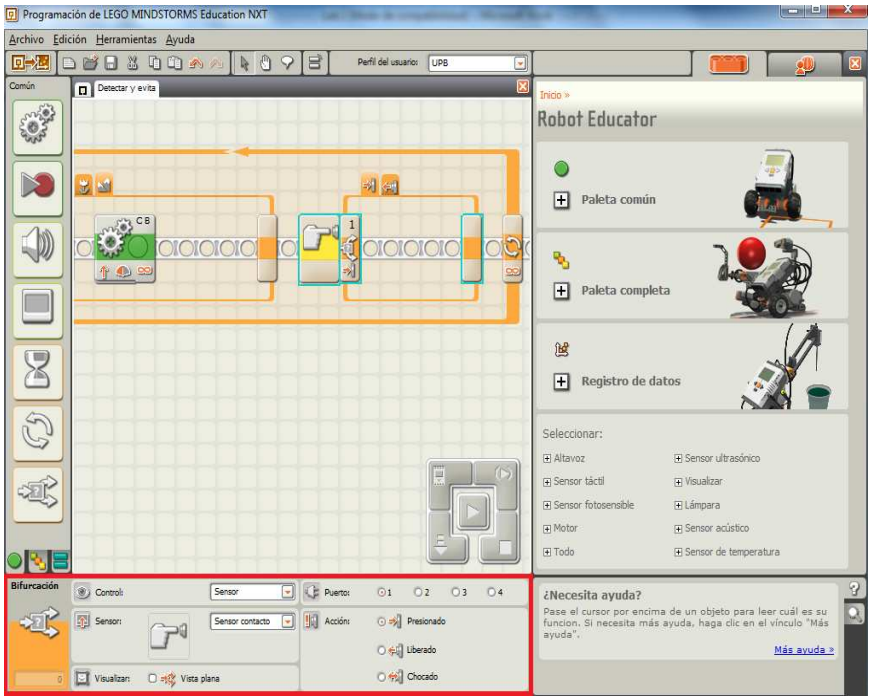



**Figura 204:** Configuración del segundo bloque Bifurcación.

Coloque un bloque *Sonido* dentro del caso "verdadero" del bloque segundo bloque de *Bifurcación*. Haga clic en el bloque *Sonido* y configurelo para reproducir un sonido, como se muestra en la Figura 205.

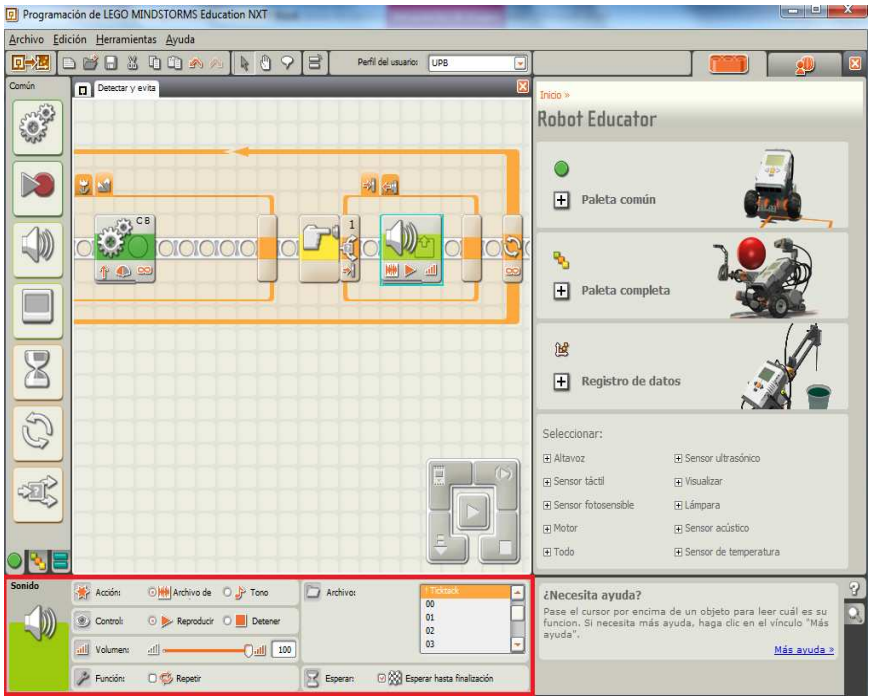



**Figura 205:** Configuración del bloque Sonido.

Coloque un bloque *Mover* después del bloque *Sonido* en el interior del bloque *Bifurcación* y configurelo como se muestra en la Figura 206.

Coloque otro bloque *Mover* despues del bloque *Mover* que acaba de agregar dentro del segundo bloque *Bifurcación,* y configurelo como se muestra en la Figura 207.

Nada será colocado dentro del caso *"Falso"* del segundo bloque *Bifurcación.*

Coloque otro bloque *Bifurcación* dentro del bloque *Bucle* despues del segundo bloque *Bifurcación.* Haga clic sobre el nuevo bloque *Bifurcación* y configurelo como se muestra en la Figura 208.

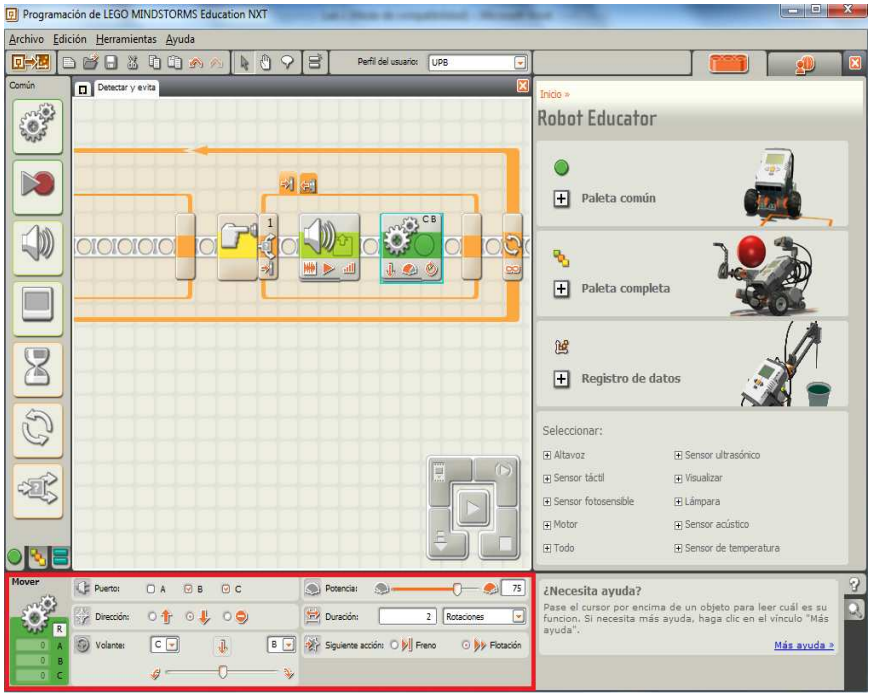



**Figura 206:** Configuración del bloque Mover.

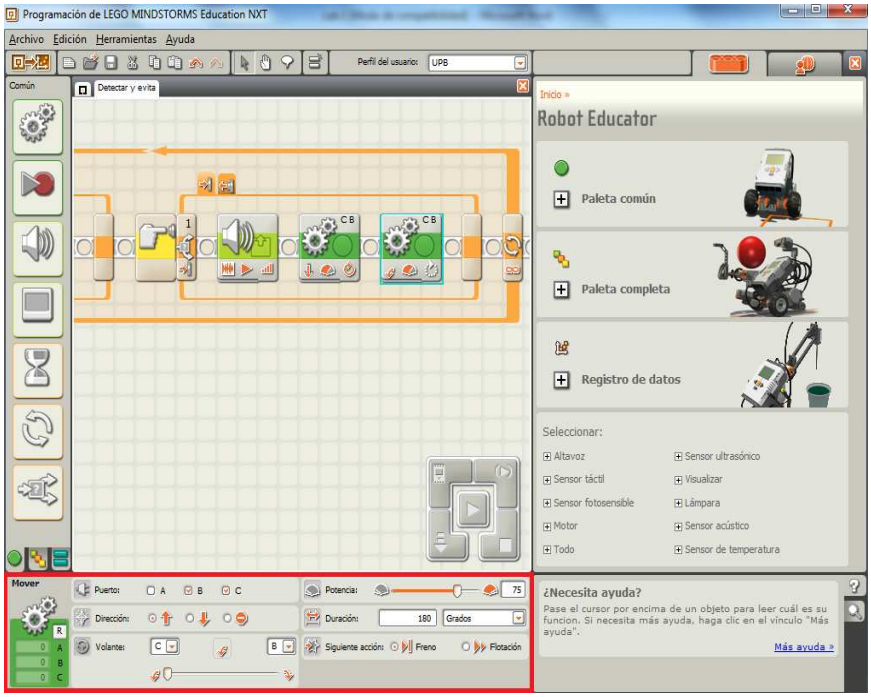



**Figura 207:** Configuración del segundo bloque Mover.

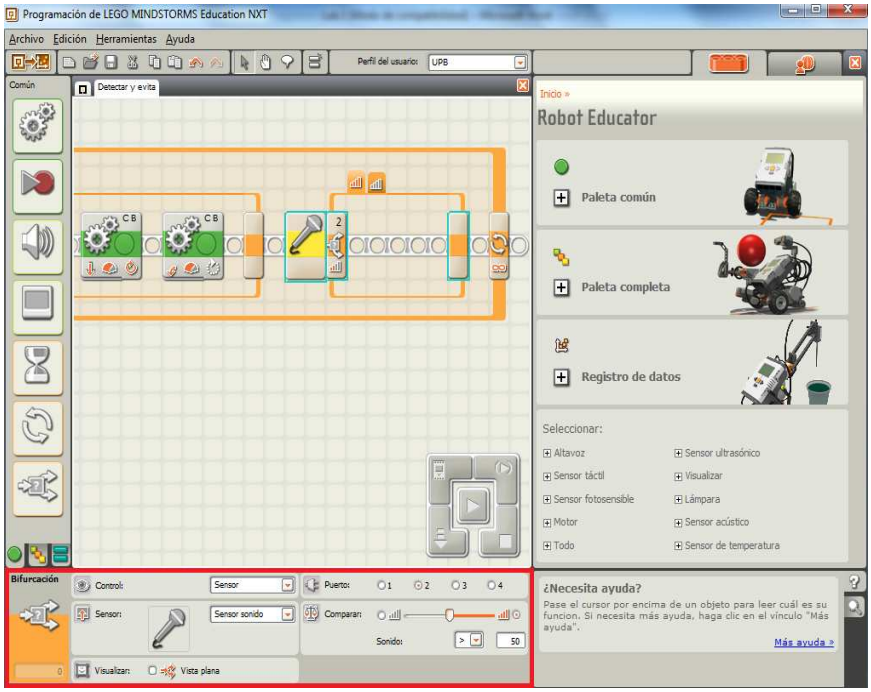



**Figura 208:** Configuración del tercer bloque Bifurcación.

Coloque un bloque *Mover* dentro del nuevo bloque *Bifurcación* en el caso *"Verdadero",* y configurelo como se muestra en la Figura 209.

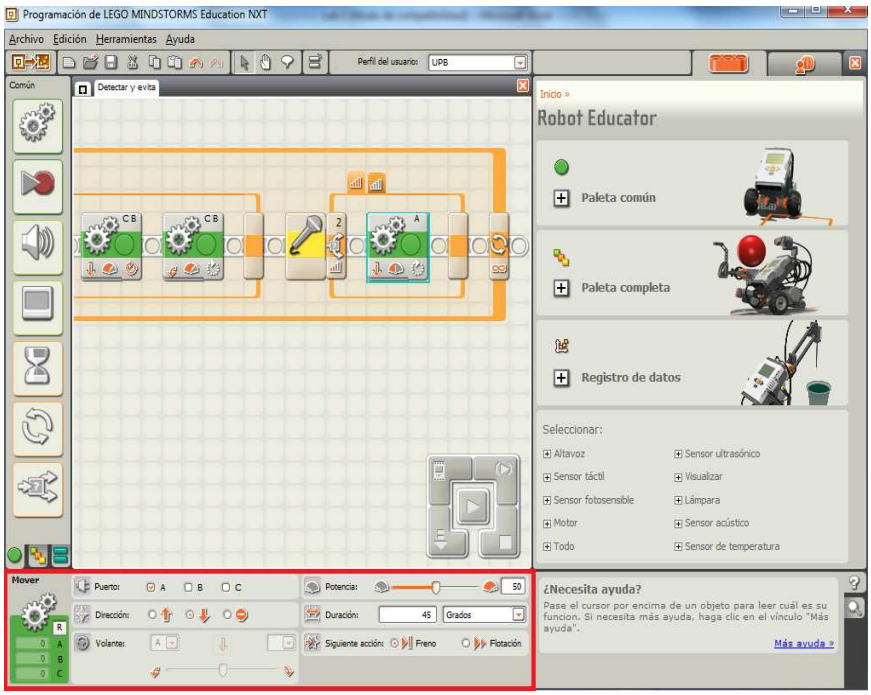



**Figura 209:** Configuración del bloque Mover.

Este bloque cerrará las garras antes de que el programa termine.

Coloque un bloque *Stop* que se encuentra en la sub-paleta de *Flujo* de la paleta *Completa* y coloquelo despues del bloque *Mover* que recientemente se ha colocado.

Nada será colocado dentro del caso *"Falso"* del tercer bloque *Bifurcación.*

El código del programa se muestra en la Figura 210.

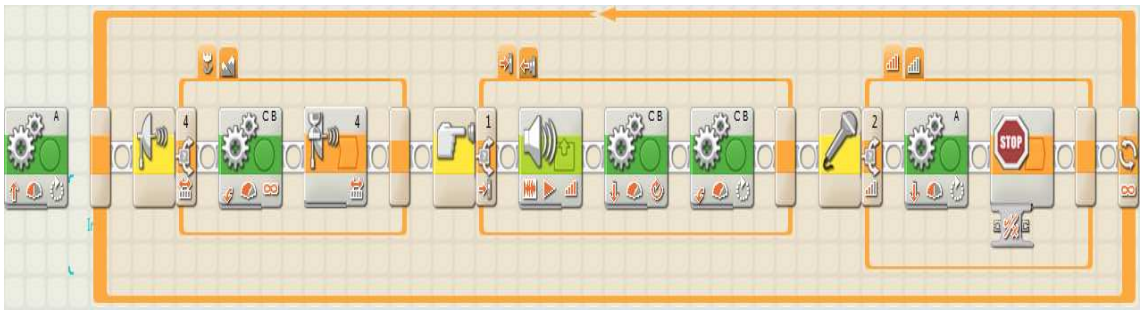


**Figura 210:** Código del programa.

Guarde su programa, y descarguelo a su robot.

Asegúrese de que las garras del Tribot están totalmente cerradas y en frente del Tribot. Hay un tercer motor en la parte inferior del Tribot que controla las garras. Gire el rayo negro que se adjunta a este motor para mover las garras en la parte delantera del Tribot.

Ejecute el programa. El robot debe abrir sus garras para que la barra del sensor de contacto este expuesta. Luego se debe avanzar en línea recta hasta que el sensor ultrasónico detecta un objeto dentro del rango de 10 pulgadas. Cuando se detecta el objeto, se debe girar hasta que el objeto ya no se detecta, y luego volver a avanzar. Si el robot pasa por un objeto que es demasiado pequeño como para ser detectado por el sensor ultrasónico, el objeto debe activar la barra del sensor táctil. Esto haría que el robot, gire a la izquierda, y reanude el movimiento. El robot debe emitir una "alarma" cuando la barra de sensor táctil está presionada. Si el robot detecta un ruido fuerte, las garras se deben cerrar y el programa debe dejar de ejecutarse.

**Taller:** Mover objetos pequeños en lugar de evitarlos.

El objetivo de este taller es ampliar la funcionalidad del Segundo programa. En lugar de evitar los pequeños objetos detectados por el sensor de contacto. Cuando un objeto pequeño se detecta el robot debe recoger el objeto con las garras, moverlo a un lado, y luego continuar en la dirección deseada. Utilice las pelotas de colores que vienen con el kit para simular los objetos pequeños.

# COMANDOS DE SONIDO

**Descripción del sensor de sonido.**

En el cuadro desplegable *Perfil de usuario*, seleccione su perfil.

Seleccione *Archivo >> Nuevo* para crear un nuevo programa.

Coloque un bloque de *Sensor de sonido* que se localiza en la paleta *Sensor,* como se muestra en la Figura 211.

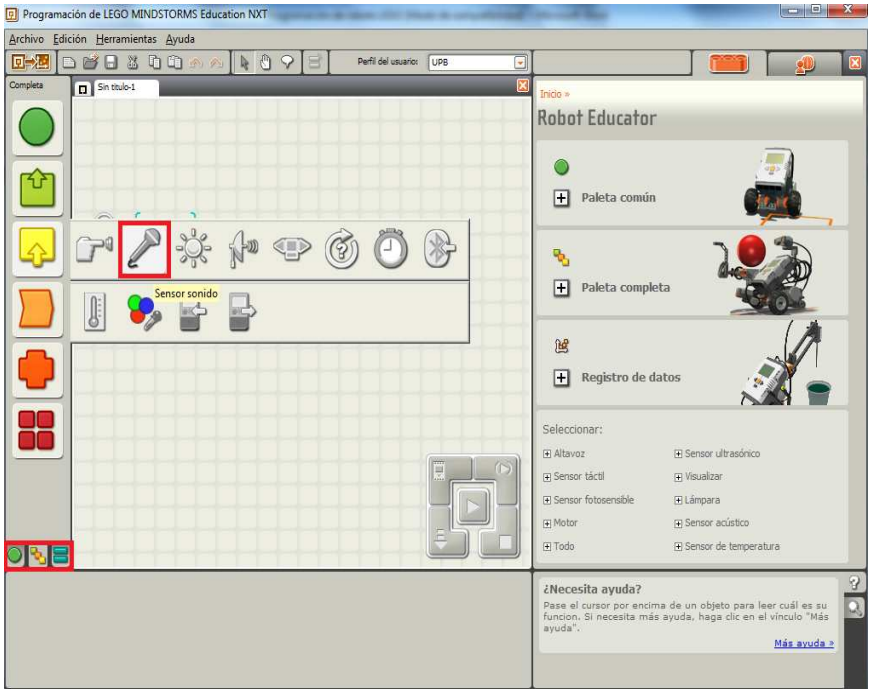



**Figura 211:** Localización del sensor de sonido.

Haga clic en el bloque del *Sensor de sonido*. La sección de configuración que se muestra en la Figura 212 debe aparecer en la parte inferior de la interfaz del software NXT 2.1 Programming.

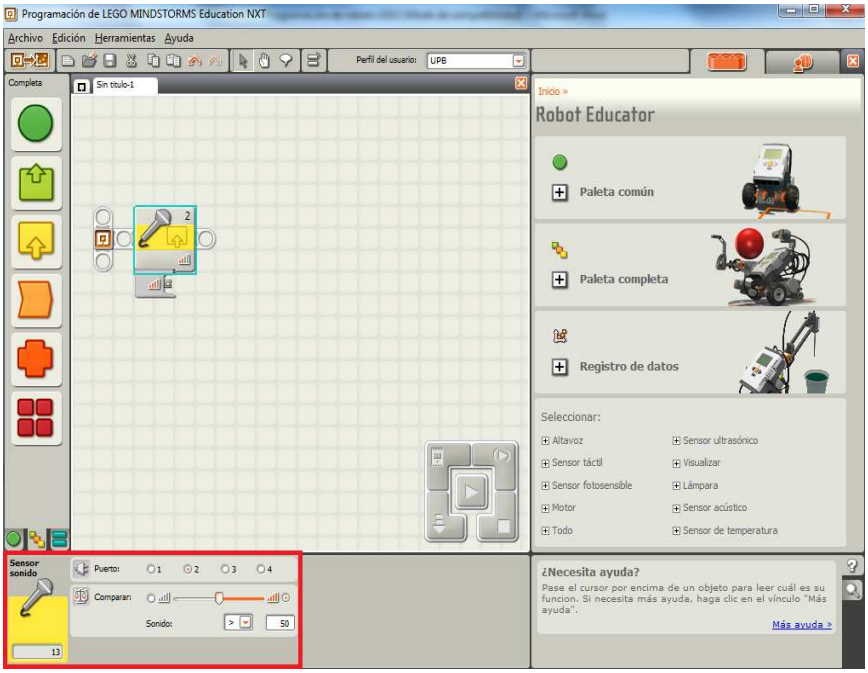



**Figura 212:** Ventana de configuración del Sensor de sonido.

Truene los dedos varias veces muy cerca del *Sensor de sonido*, y observe los resultados en el indicador de *Sensor de sonido*, como se muestra en la Figura 212.

Escriba un programa sencillo que mueva el Tribot hacia adelante con una potencia igual a la intensidad del sonido que escucha, como se muestra en la Figura 213.

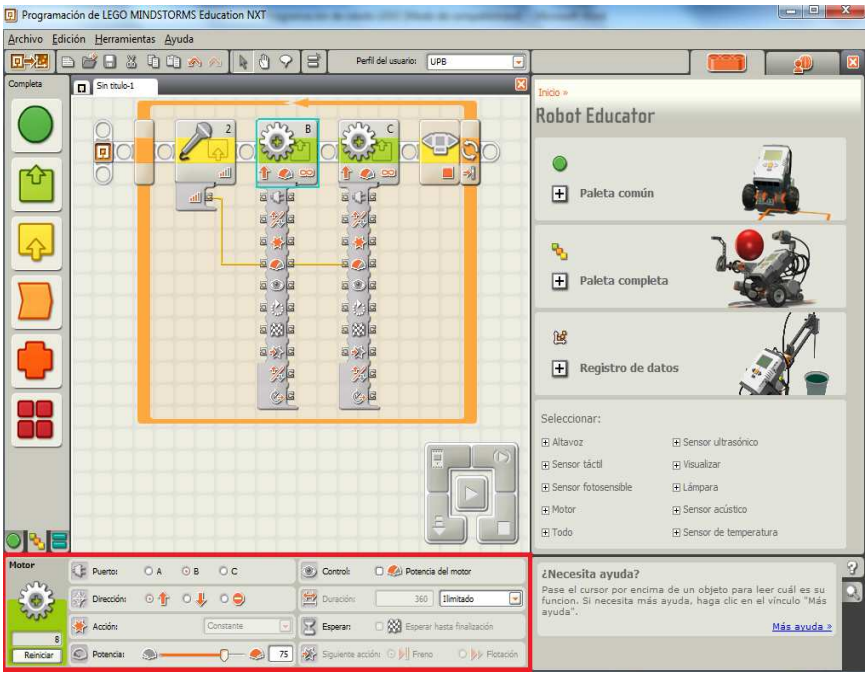



**Figura 213:** Programa que mueve los motores a la misma potencia que el sensor de sonido.

**Taller:** Levante el Tribot para que las ruedas no toquen el suelo. Cree un programa simple que gire las ruedas a plena potencia durante 10 segundos. Monitore el *Sensor*

*de sonido*, mientras las ruedas giran (un bloque de *Sensor de sonido* sigue siendo necesario para hacer esto).

**Conteo de eventos de sonido.**

El *Sensor de sonido* no sólo tiene la capacidad de medir los niveles de volumen, sino que también puede ser usado como un interruptor. En muchos casos, una estructura *Bifuración* puede ser configurado para alternar entre un caso verdadero y un caso falso, si el nivel de volumen de un sonido detectado cae encima o debajo de un determinado nivel o umbral. Sin embargo, a veces es necesario disponer de múltiples casos que el mismo sensor debe ser capaz de alternar. Este es un ejemplo donde puede ser utilizada una variable para almacenar el número de veces que el sensor ha detectado el sonido de audio dentro de un cierto período de tiempo. LEGO Mindstorms NXT tiene un bloque *Variable* que se encuentra en la *paleta de datos* que se puede utilizar para este propósito.

Para demostrar este concepto, considere el diagrama de la Figura 214.

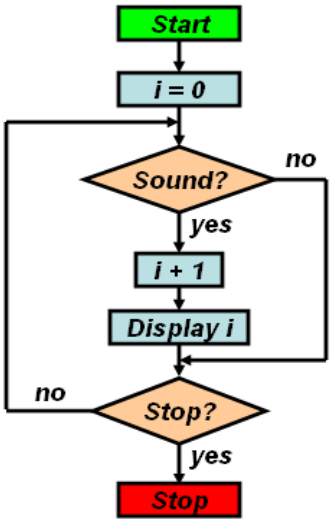



**Figura 214:** Diagrama de flujo.

Los nodos del rectángulo azul en el diagrama de flujo indican las funciones o procesos. Los diamantes de color naranja son las preguntas o los interruptores. Las flechas representan el resultado de un cálculo o una respuesta a una pregunta. Por lo general, los valores de los resultados también están escritos junto a las flechas. Tenga en cuenta que hay también los nodos: iniciar y detener. Es importante tener estos nodos para mostrar los pasos que se ejecutan en primer lugar, y qué pasos pueden dar lugar a la terminación del programa.

Vamos a traducir el diagrama de flujo en el código, siga estos pasos:

Cree un nuevo programa NXT, y guardelo como *"Conteo de evento de sonidos.rbt"*

Una variable llamada "i" debe ser definida. Vaya a *Edición>> Definir las variables* para abrir el cuadro de diálogo *Editar Variables*, como se muestra en la Figura 215. Cree una variable de Tipo de datos: *Numéricos* llamada i y cierre el diálogo, como se muestra en la Figura 216.

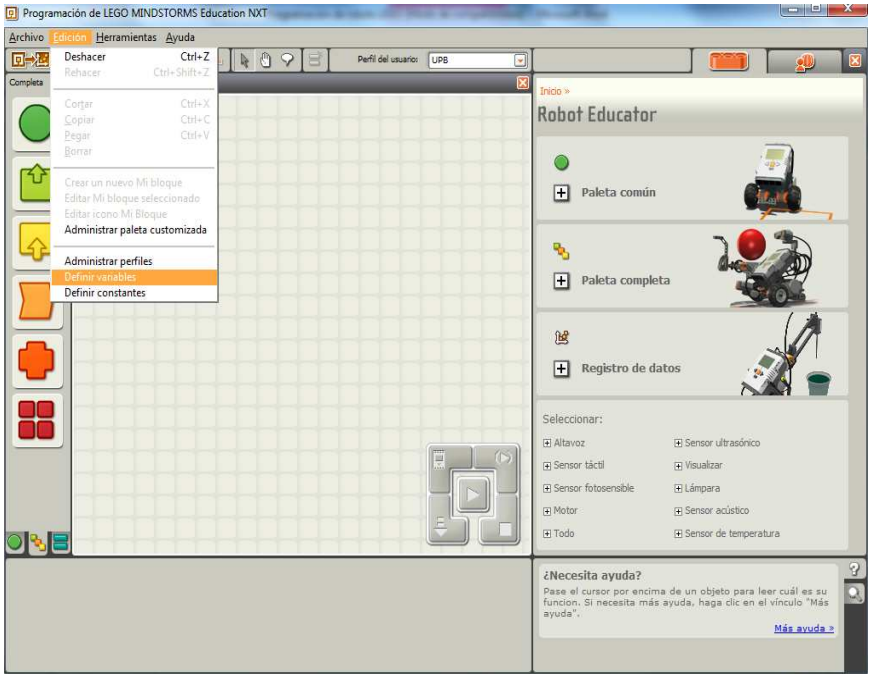



**Figura 215:** Localización del sub-menú Definir variables.

Coloque un bloque *Variable* que se encuentra en la *paleta de datos* sobre el haz de secuencia. Seleccione la variable "i" de la lista, y configure el bloque *Variable* para escribir un valor de 0 para esta variable. Esto asegura que no hay datos de basura presentes en la variable antes de que se utilice en el programa. Es muy recomendable hacer esto cada vez que utilice la definición de las variables en los programas para garantizar un comportamiento esperado.

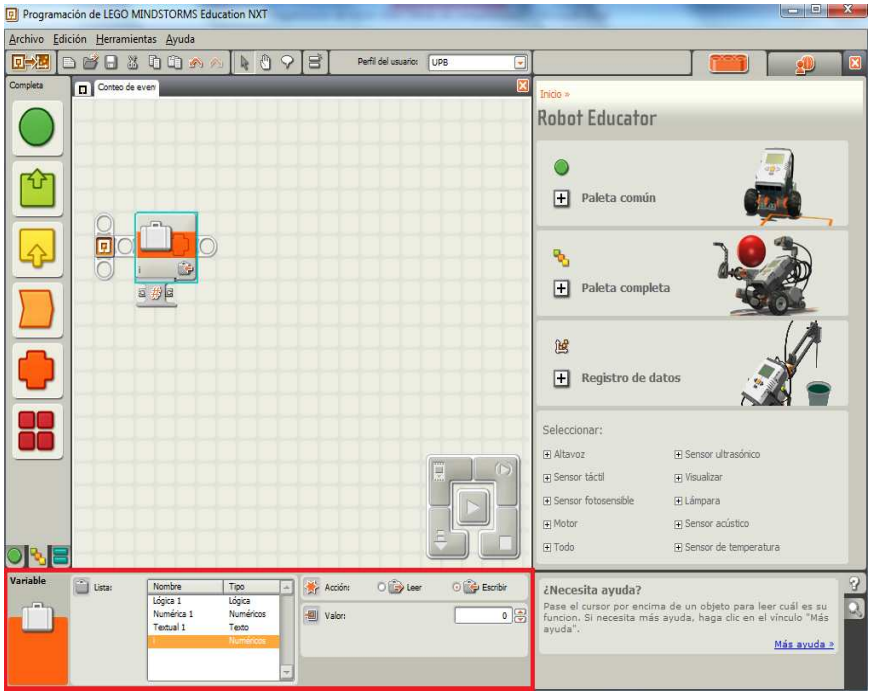



**Figura 216:** Configuración del bloque Variable.

Coloque una estructura *Bucle* después del bloque *Variable*. Configure el *Bucle* para que se detenga cuando se pulse el botón ENTER del NXT, como se muestra en la Figura 217.

Coloque un bloque *Bifurcación* que se encuentra en la *paleta de flujo* dentro del *Bucle*. Configure el bloque *Bifurcación* como se muestra en la Figura 218.

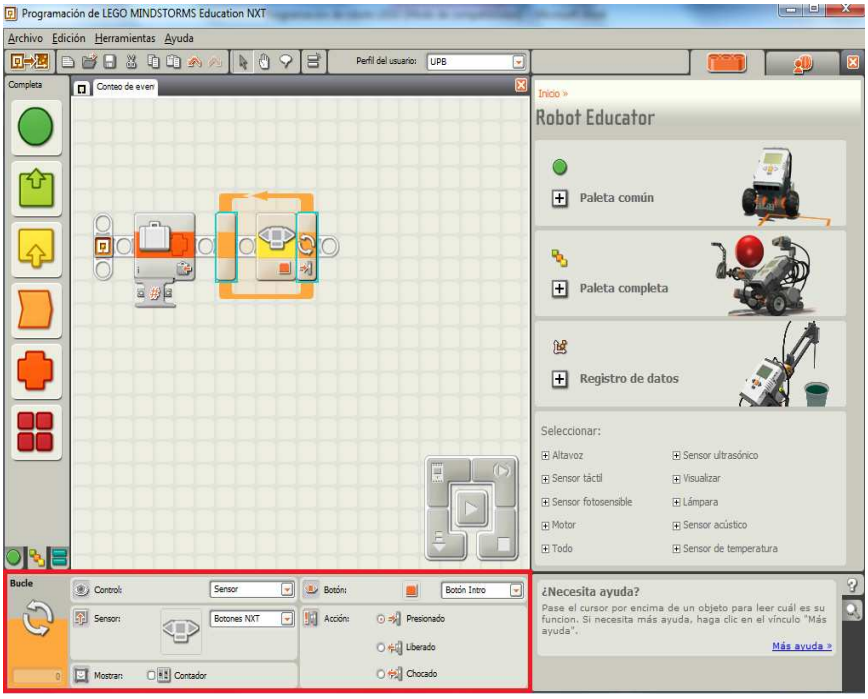



**Figura 217:** Configuración del bloque Bucle.

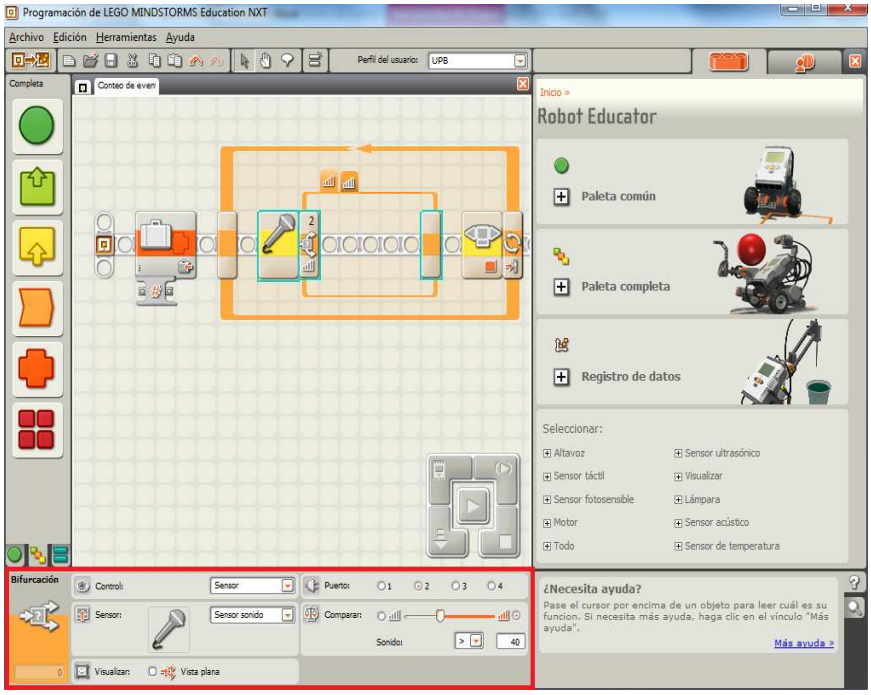



**Figura 218:** Configuración del bloque Bifurcación.

Insertar otro bloque *Variable* dentro del caso *"verdadero"* del bloque *Bifurcación*. Configure este bloque *Variable* para leer el valor de "i".

Observe en el diagrama de flujo que "i" se incrementa cuando se detecta un sonido. Use un bloque *Matemáticas* que se encuentra en la *paleta de datos* para incrementar el valor almacenado en "i". Ate el valor de salida del bloque *Variable* a la entrada A del bloque *Matemáticas*. Esto hará que el bloque *Variable* pase el valor de "i" cuando se lee en el bloque *Matemáticas* para que pueda ser incrementado. En la ventana de configuración del bloque *Matemáticas*, configure B es igual a 1 para que "i" sólo se incremente en 1. El resultado final de esta configuración debe tener el aspecto de la Figura 219.

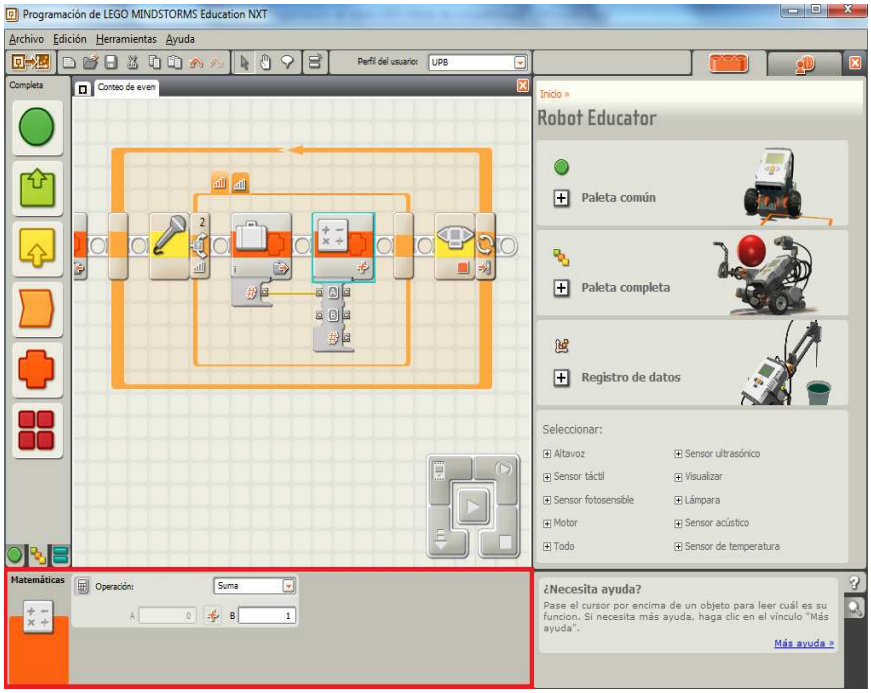



**Figura 219:** Configuración del bloque Matemáticas.

Para almacenar el resultado de este cálculo, coloque otro bloque *Variable* después del bloque *Matemáticas*. Configure el bloque *Variable* para escribir en la variable "i". Conecte la salida del valor del bloque *Matemáticas* a la entrada de valor del bloque *Variable,* como se muestra en la Figura 220.

El siguiente paso en el diagrama de flujo para el nuevo número es que se muestre en la pantalla frontal del controlador NXT. Para mostrar este número en la pantalla, primero se debe convertir de un número a texto. Esto se debe a que la pantalla en el controlador NXT sólo acepta texto en la pantalla.

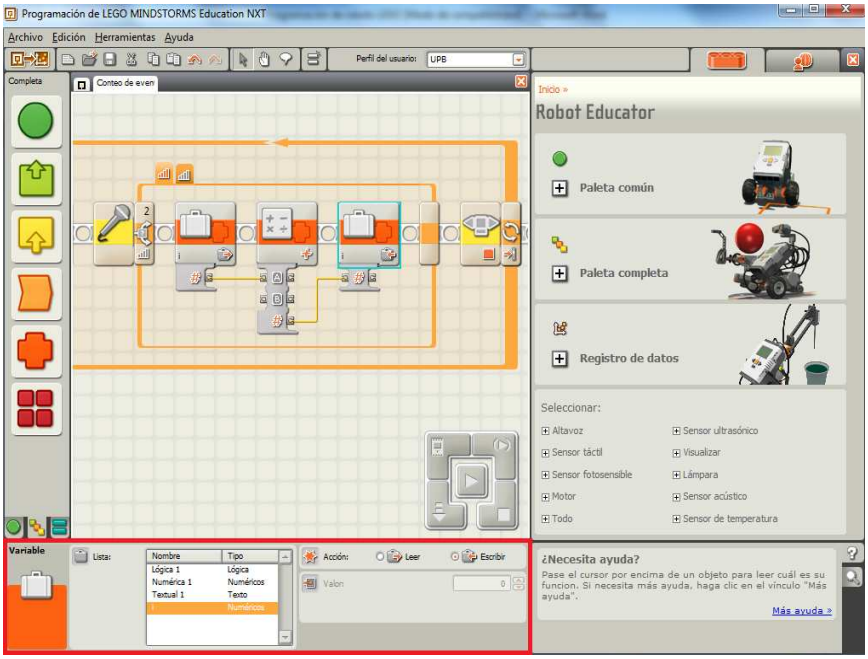



**Figura 220:** Configuración del bloque Variable.

Convierta "i" en texto, coloque un bloque *Número a texto* que aparece en la *paleta avanzada* después del bloque *Variable* recientemente colocado. Conecte la salida del valor del bloque *Variable* adyacente a la entrada del valor del bloque *Número a texto*, como se muestra en la Figura 221.

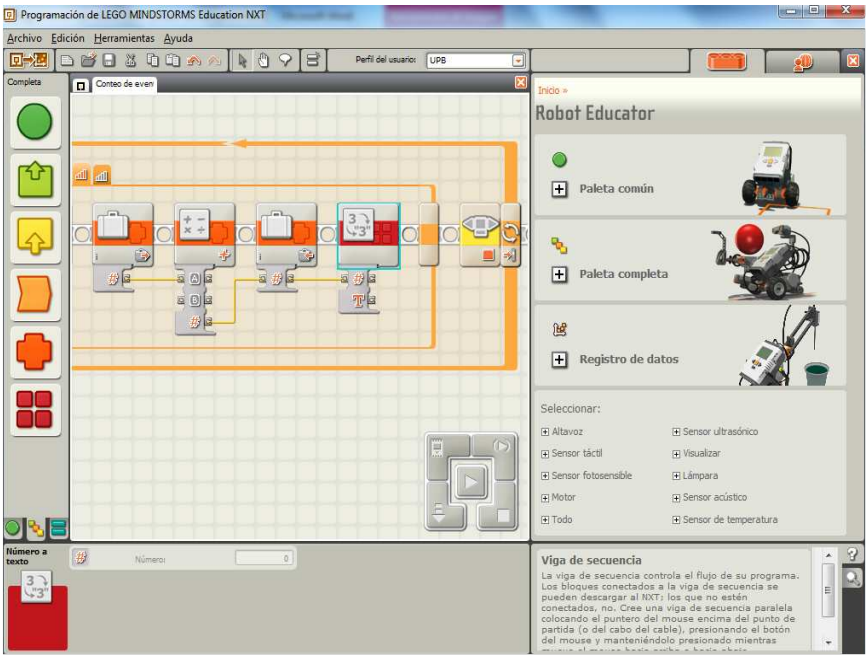



**Figura 221:** Bloque Número a texto.

Coloque un bloque *Visualizar* que se encuentra en la *paleta de acción* después del bloque *Número a texto*, y configurelo como se muestra en la Figura 222.

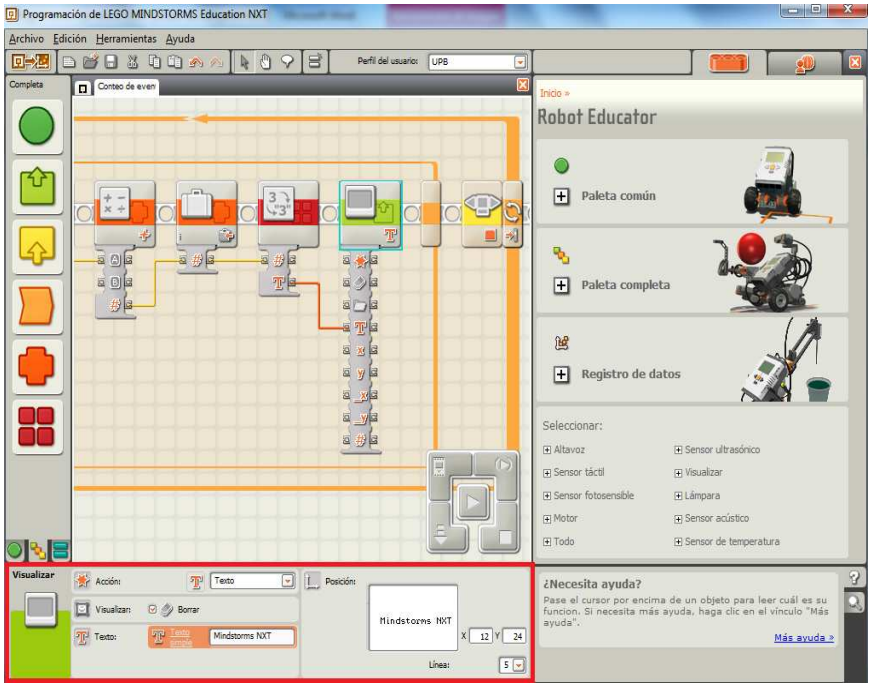



**Figura 222:** Configuración del bloque Visualizar.

Para ver las entradas y salidas que están disponibles para el bloque de pantalla, mueva el cursor hasta el borde inferior izquierdo del bloque Visualizar hasta que el cursor se convierte en punto de arriba a abajo. Haga clic en el borde inferior izquierdo del bloque de la pantalla. Un gabinete de entradas y salidas deben abrirse por debajo de ella.

Conecte la salida de texto del bloque *Número a texto* a la entrada de texto del bloque de la pantalla. Esto hará que el valor de "i" se muestre en la pantalla del controlador NXT en lugar de *"Mindstorms NXT"*.

Coloque un bloque de *Sonido* que se encuentra en la *paleta de acción* al principio del caso *"verdadero"* del bloque *Bifurcación*. Configure el bloque de *Sonido* como se muestra en la Figura 223.

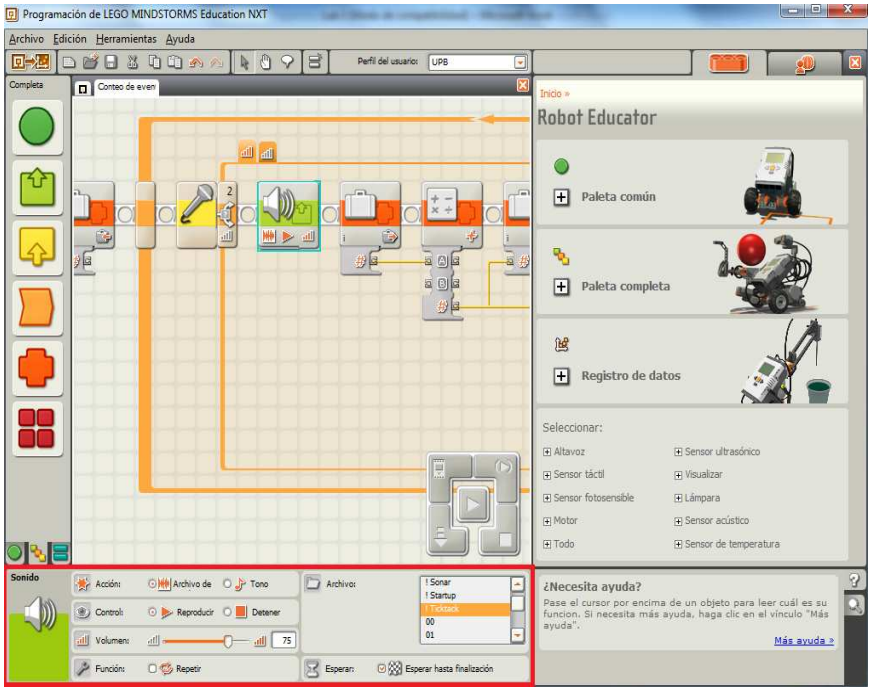



**Figura 223:** Configuración del bloque Sonido.

Este bloque está destinado a reproducir un sonido cuando el caso *"verdadero"* del bloque *Bifurcación* se activa. Se trata de una técnica de depuración de gran ayuda, ya que se notifica al oyente un sonido cuando se ha detectado, sin tener que mirar la lectura del sensor en la pantalla. Esta es otra técnica de depuración de gran ayuda para escribir programas. Agregar marcas como esta ayudan a ilustrar lo que se está ejecutando actualmente en el programa.

El código del programa se muestra en la Figura 224.

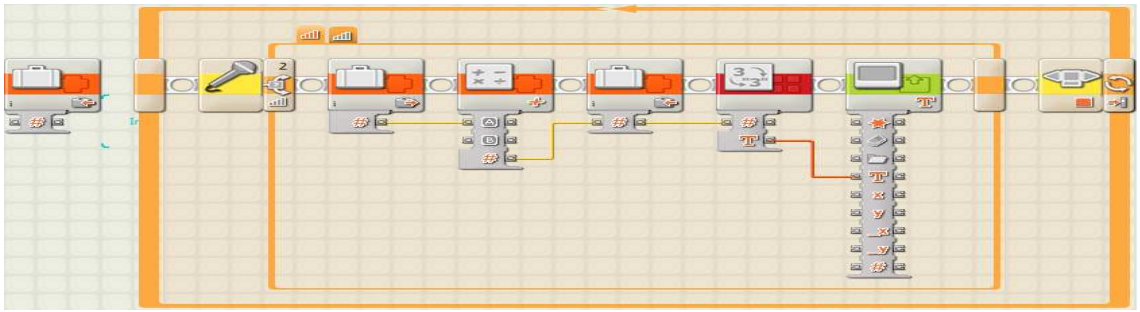


**Figura 224:** Código del programa.

**Nota:** El gabinete del conector para el bloque de la pantalla puede ser cerrado una vez que todas las conexiones se han realizado para ahorrar espacio. Las conexiones se mantienen intactas, mientras que el gabinete está cerrado.

Guarde el código, y descarguelo en el NXT. Ejecute el programa, y vea la pantalla mientras truena los dedos en las proximidades del sensor de sonido.

Hay un par de cosas que puede hacer para corregir los problemas en el código del programa. Una de ellas es, asegurarse de que el umbral de nivel de sonido que se establece para el estado del interruptor del sensor de sonido es lo suficientemente alto como para filtrar el ruido ambiental del entorno y de los motores, pero lo suficientemente bajo como para detectar un ruido a una distancia razonable (4 - 12 pulgadas de distancia.) Encuentre el umbral, que es único para todos los robots, y se encuentra generalmente a través de pruebas. Asegúrese de que el umbral que se muestra en la Figura 225 está muy bien adaptado a su Tribot.

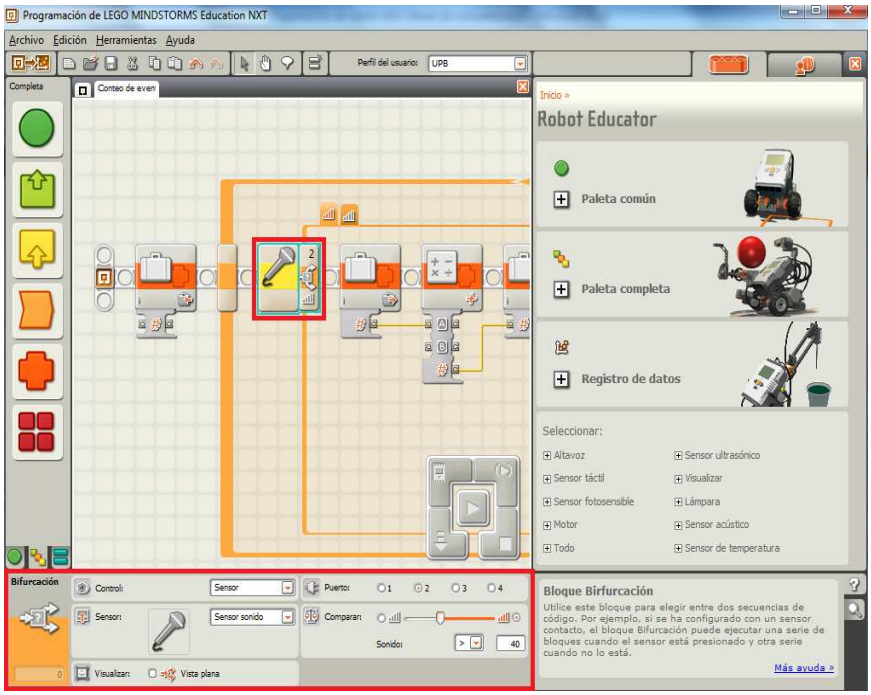



**Figura 225:** Configuración del umbral para el sensor de sonido.

Otra forma de mejorar el rendimiento de conteo es dejar de leer el sensor de sonido una vez que el sonido se ha detectado hasta que el volumen del sonido cae por debajo del umbral. Esto evita que el código registre un sonido largo y fuerte (como un silbido), como varios sonidos.

Para solucionar este problema, coloque un bloque *Espera* que se encuentra en la *paleta de flujo* después del bloque de la pantalla. Configure este bloque para que espere hasta que el nivel de ruido detectado por el sensor de sonido este debajo del umbral que se describe en el estado del interruptor del sensor de sonido. Esto hará que la condición de paso para ejecutar el caso *"verdadero"* ocurre cuando el sonido que es detectado esta por encima de un cierto nivel, y la estancia en el caso *"verdadero"* hasta que el sonido detectado es inferior a este umbral.

Una correcta implementación de este código (Figura 226) tendrá el registro de un silbido largo y fuerte como un solo sonido.

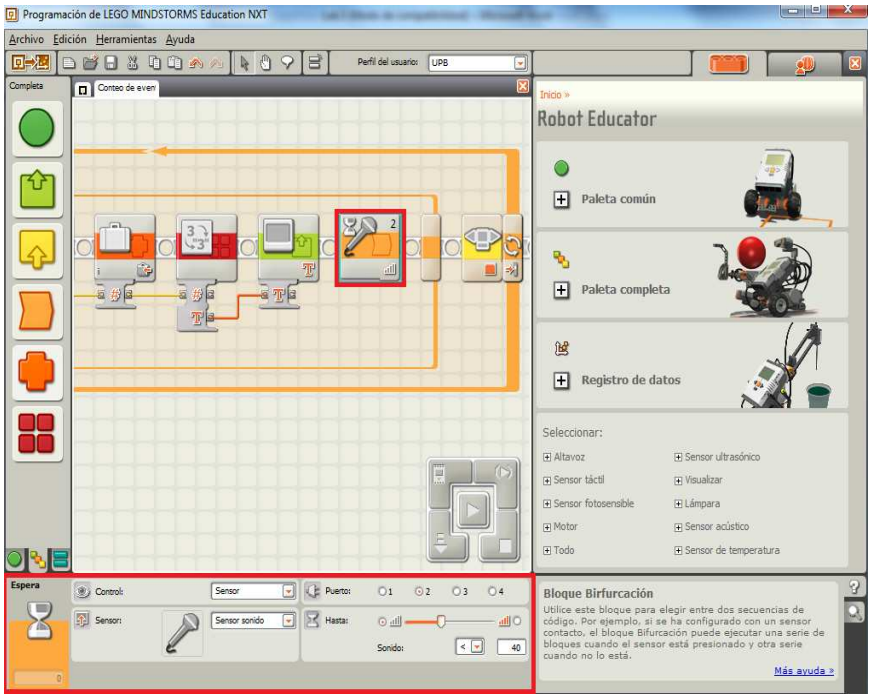



**Figura 226:** Configuración del bloque Espera.

**Taller:** Control del Tribot utilizando comandos de sonido.

El objetivo de este taller es el control del Tribot usando sólo su sensor de sonido. Además, el Tribot automáticamente debe evitar los objetos que son detectados por el sensor ultrasónico. Usando el conocimiento del sensor de sonido, sensor ultrasónico, las variables, la evitación de objetos a partir del taller anterior, y diagramas de flujo, completa este taller mediante el cumplimiento de los siguientes requisitos:

El Tribot debe demostrar todos los elementos que se muestran en el diagrama de flujo de la Figura 227 en la secuencia correcta.

El robot debe avanzar continuamente hasta que se procesa un comando ya sea a la izquierda, derecha, o se detenga.

El Tribot debe mantener una cuenta exacta de los sonidos detectados en un intervalo de dos segundos antes de que la cuenta se transforme en un comando. Si el Tribot detecta un objeto con el sensor ultrasónico, lo debe evitar.

El Tribot debe interpretar un sonido como una vuelta a la izquierda, dos sonidos como una vuelta a la derecha, y tres o más como una orden de parada dentro de un intervalo de muestreo de 2 segundos. De lo contrario, el Tribot debe moverse en la dirección de avance.

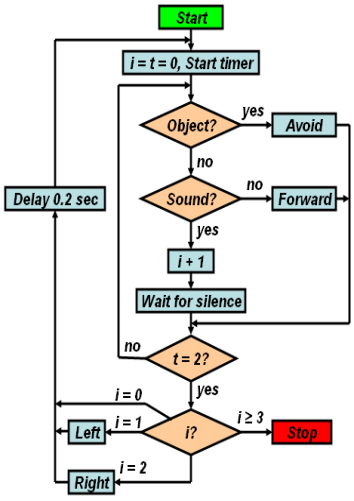



**Figura 227:** Diagrama de flujo.

**Consejos útiles:**

Insertar un sonido *"bandera"* en el comienzo del caso *“verdadero”* del interruptor del sensor de sonido para asegurarse de que el sonido ha sido detectado. El uso correcto de este bloque hará que el Tribot haga un ruido cuando se escucha y cuenta con un sonido. Esto le ayudará a depurar el reconocimiento de sonidos.

Utilice un bloque *Variable* para almacenar *"i"*.

Dos circuitos están involucrados en este programa. Uno está anidado dentro del otro.

Un bloque *Bifurcación* multi-estado es necesario para procesar: a la izquierda, a la derecha, o detener. Usted puede agregar más condiciones a una estructura *Bifurcación* y las escribe en la barra, como se ilustra a continuación:


**Figura 228:** Configuración del bloque multi-estado.

El bloque *Detener* se encuentra en la *paleta de flujo* y detiene el programa cuando sea necesario.

Un pequeño retraso de tiempo de 0,2 segundos se necesita en el final de todo el código dentro del bucle principal como se muestra en el diagrama de flujo. Use un bloque *Espera* con el parámetro de control establecido en la hora y la demora ajustada a 0,2.

Un bloque *Mover* establecido con una duración ilimitada servirá para garantizar un buen funcionamiento en la dirección de avance.

No es necesario hacer algo, si no hay ningún objeto detectado por el sensor ultrasónico.

Configuración de los motores en la Siguiente acción: *Flotación*, en lugar de Freno puede ofrecer resultados más suaves.

# ¿HERMANOS?

**Configuración de una conexión Bluetooth entre dos Tribots.**

Seleccione *Más información >>* de la parte inferior de la ventana del software NXT 2.1 Programming.

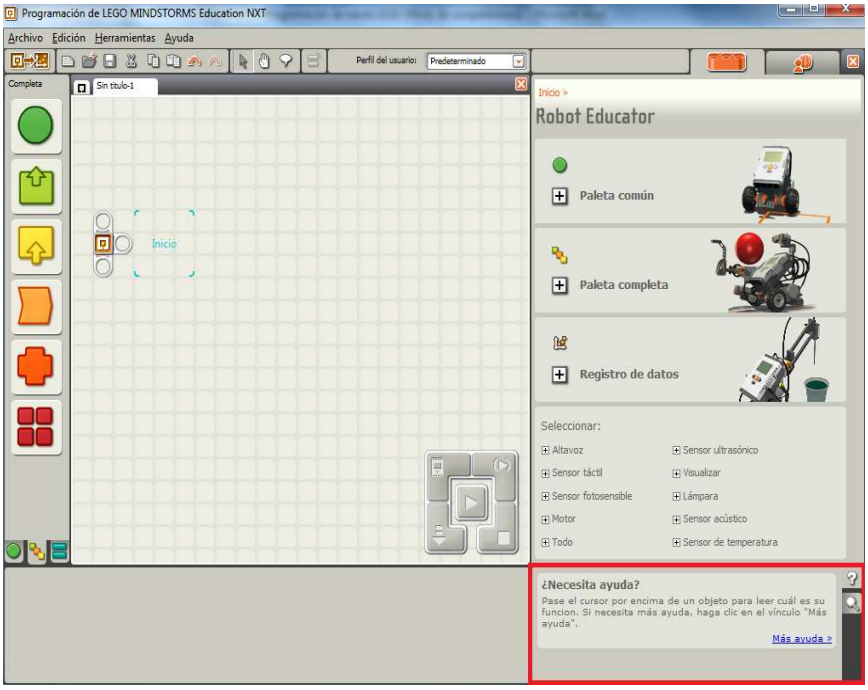



**Figura 229:** Localización del menú ayuda.

Las instrucciones sobre cómo establecer una conexión Bluetooth entre dos Tribots se explica en la *Configuración del NXT para la comunicación inalámbrica* de la ayuda del *Bloque Enviar mensajes*. La ayuda del *Bloque Enviar mensajes* se encuentra en la sección *Bloques de acción* de la vista de contenido de la ayuda, como se muestra en la Figura 230.

Decida qué robot será el maestro (Gran Hermano) mediante la creación de la conexión. La conexión del robot esclavo (Hermano pequeño) se configurará automáticamente.

Hacer la conexión número "1" entre los robots.

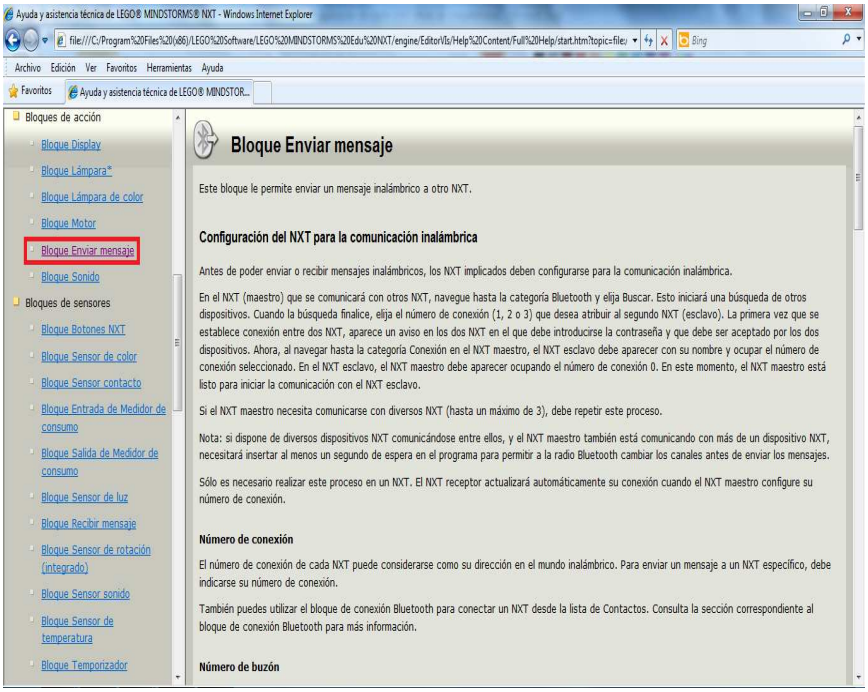



**Figura 230:** Ayuda para la comunicación inalámbrica.

**Gran Hermano.**

En el cuadro desplegable *Perfil de usuario*, seleccione su perfil.

Seleccione *Archivo >> Nuevo* para crear un nuevo programa.

Guarde el programa como *Gran Hermano.*

Asegúrese de que la vista de la paleta completa esta seleccionada.

Coloque un bloque *Bucle* que se encuentra en la *paleta de flujo*. Configure el bloque *Bucle*, como se muestra en la Figura 231.

Coloque un bloque *Enviar mensaje* que se encuentra en la *paleta de acción* dentro del *Bucle*, y configúrelo como se muestra en la Figura 232. Esta es la configuración del "*Gran Hermano"* para enviar el mensaje de texto "Adelante" al buzón del primer *"Hermano pequeño"*.

Coloque un bloque *Mover* después del bloque *Enviar mensaje*, y configúrelo como se muestra en la Figura 233.

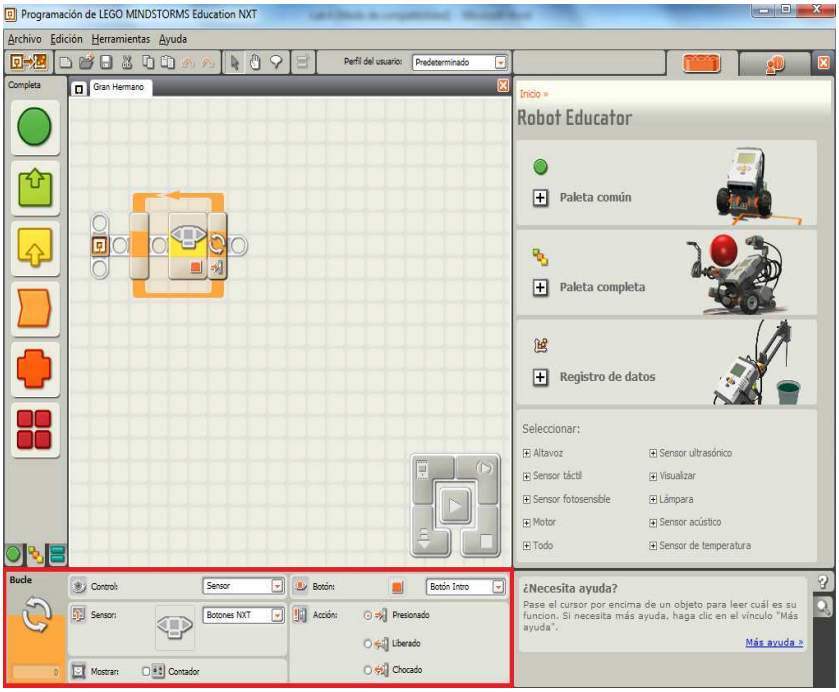



**Figura 231:** Configuración del bloque Bucle.

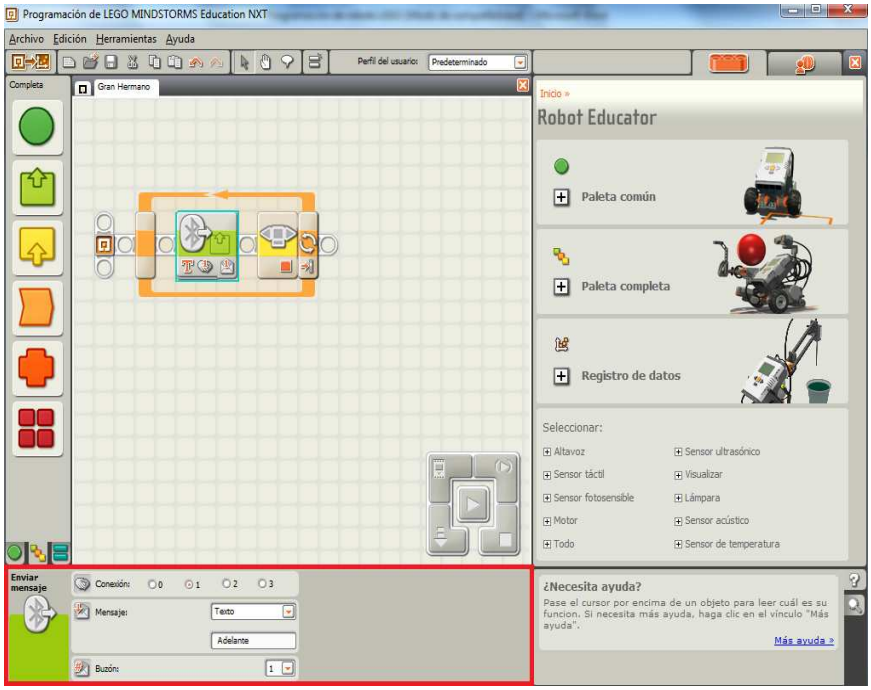



**Figura 232:** Configuración del bloque Enviar mensaje.

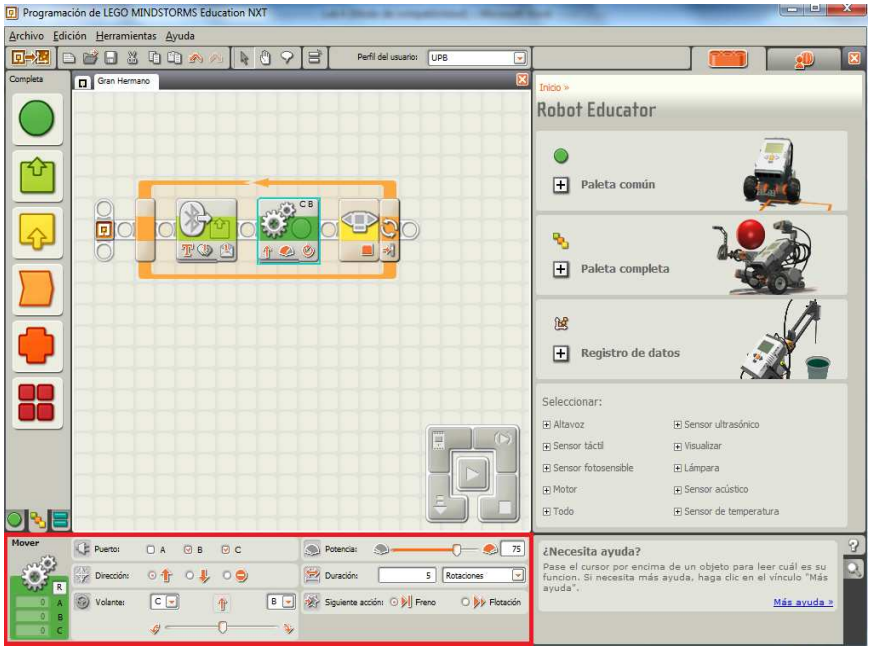



**Figura 233:** Configuración del bloque Mover.

Coloque otro bloque *Enviar mensaje* después del bloque *Mover*, y repita la configuración de la Figura 232, con el mensaje de texto *"Izquierda"*.

Coloque un bloque *Mover* después del bloque *Enviar mensaje*, y configúrelo como se muestra en la Figura 234.

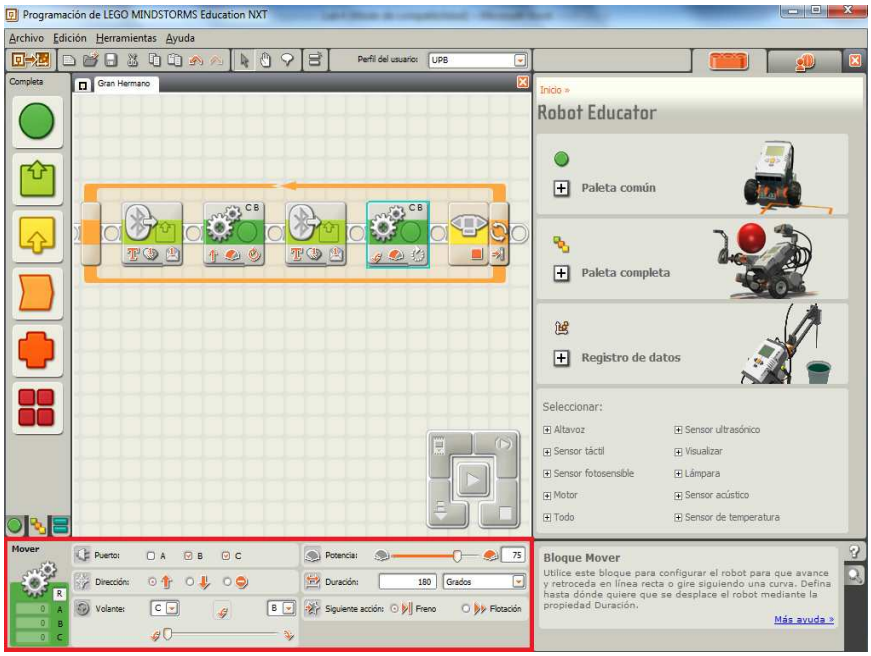



**Figura 234:** Configuración del segundo bloque Mover.

Coloque otro bloque *Enviar mensaje* después del bloque *Mover*, y repita la configuración de la Figura 232.

Coloque un bloque *Mover* después del bloque *Enviar mensaje*, y configúrelo como se muestra en la Figura 233.

Coloque otro bloque *Enviar mensaje* después del bloque *Mover*, y repita la configuración de la Figura 232, con el mensaje de texto *"Derecha"*.

Coloque un bloque *Mover* después del bloque *Enviar mensaje*, y configúrelo como se muestra en la Figura 235.

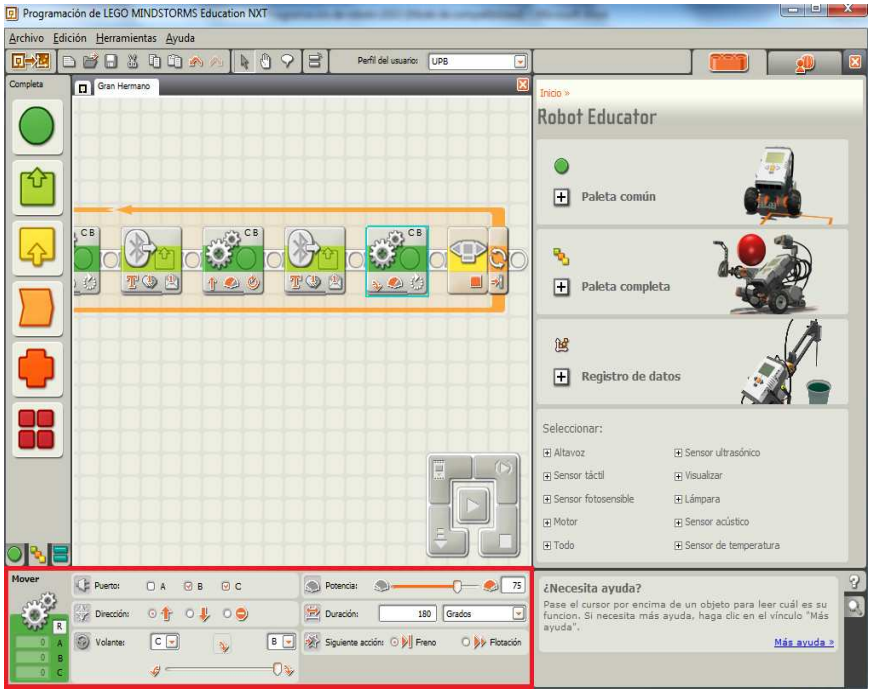



**Figura 235:** Configuración del bloque Mover.

Coloque otro bloque *Enviar mensaje* después del bloque *Mover*, y repita la configuración de la Figura 232.

Coloque un bloque *Mover* después del bloque *Enviar mensaje*, y configúrelo como se muestra en la Figura 233.

Coloque otro bloque *Enviar mensaje* después del bloque *Mover*, y repita la configuración de la Figura 232, con el mensaje de texto *"Spin"*.

Coloque un bloque *Mover* después del bloque *Enviar mensaje*, y configúrelo como se muestra en la Figura 236.

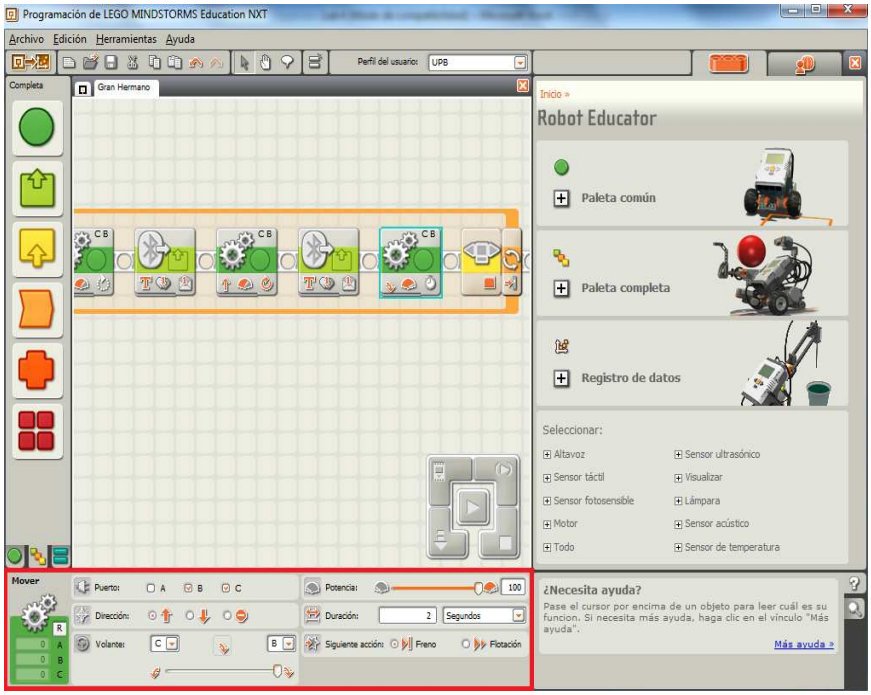



**Figura 236:** Configuración del último bloque Mover.

El código del Gran Hermano debe verse como el de la Figura 237.

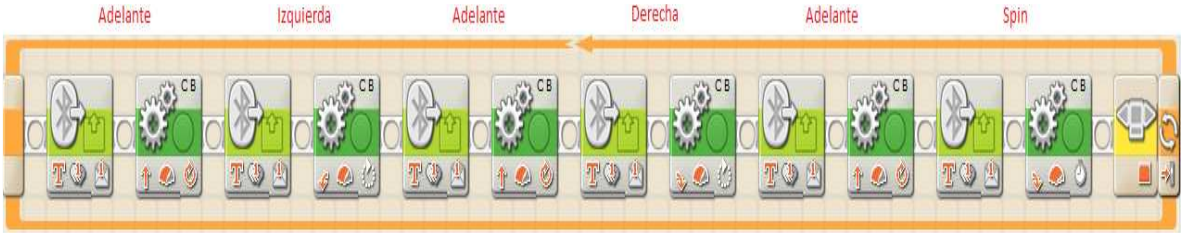



**Figura 237:** Código del Gran Hermano.

**Sugerencia:** Usted puede agregar comentarios al código, haga doble clic en cualquier parte del espacio en blanco y escriba su comentario. Esta práctica mejora la legibilidad del código, y es altamente recomendado para la programación.

Guarde el programa y descargarlo en uno de los Tribots que se usará como Gran Hermano.

**Hermano pequeño.**

Seleccione *Archivo >> Nuevo* para crear un nuevo programa.

Guárdelo como *"Hermano pequeño".*

Seleccione la vista de la paleta completa de la izquierda.

Coloque un bloque *Bucle* en la *paleta de flujo*, y configúrelo como se muestra en la Figura 238.

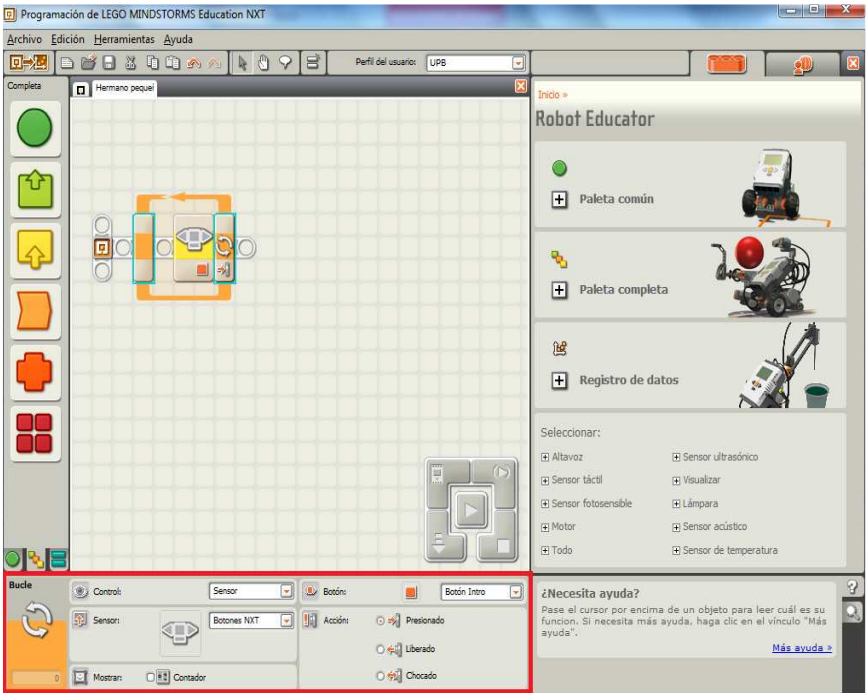



**Figura 238:** Configuración del bloque Bucle.

Coloque un bloque *Recibir mensaje* se encuentra en la *paleta Sensor*, y configúrelo como se muestra en la Figura 239. Este bloque se utiliza para escuchar los comandos enviados desde el Gran Hermano a través de tecnología inalámbrica Bluetooth.

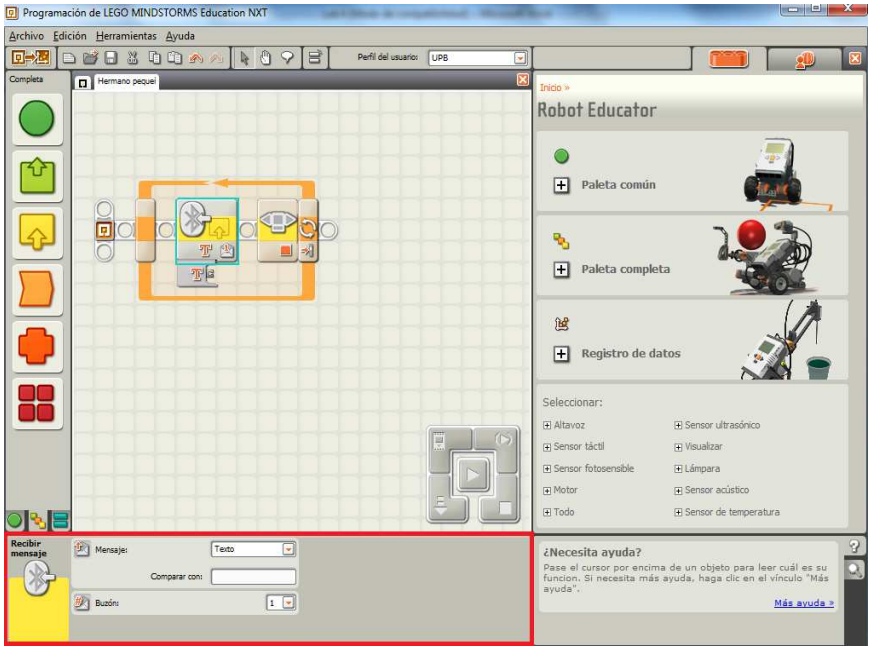



**Figura 239:** Configuración del bloque Recibir mensaje.

Coloque un bloque *Bifurcación* que se encuentra en la *paleta de flujo* después del bloque *Recibir mensaje*, y configúrelo para cambiar entre los cinco valores diferentes de texto como se muestra en la Figura 240.

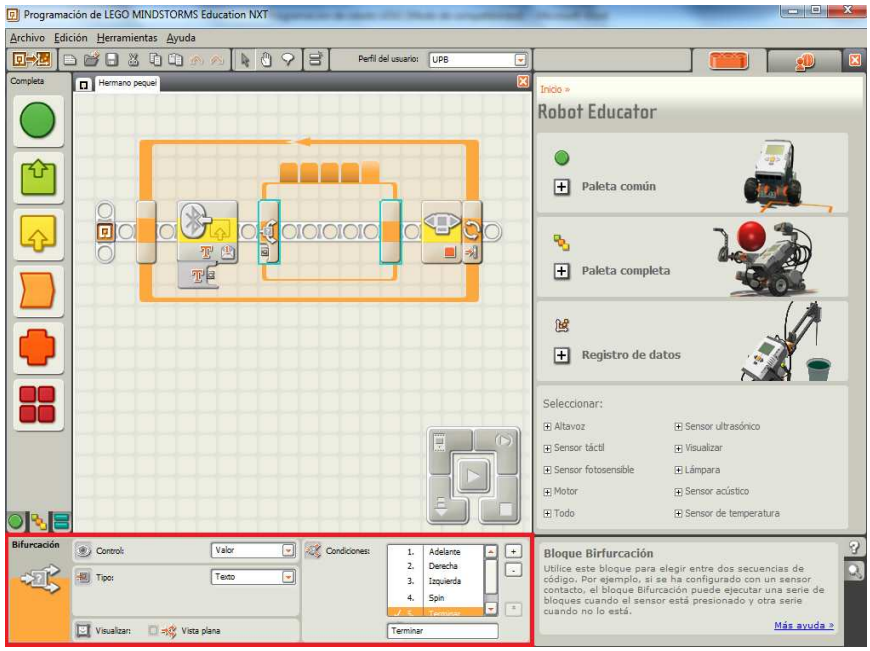



**Figura 240:** Configuración del bloque Bifurcación.

**Importante:** Asegúrese de desmarcar Vista plana antes de añadir casos nuevos.

Asegúrese de agregar el caso por defecto haciendo clic en el botón asterisco en la esquina inferior derecha de la ventana de configuración, mientras que el caso

*"Terminar"* se pone de relieve. Esto asegurará que si algún otro mensaje de texto se recibe y que no sea los de la lista, el caso Terminar será ejecutado por defecto. Esto evitará que el Tribot se mueva cuando un extraño mensaje de texto se reciba.

Coloque un bloque *Mover* dentro del caso *"Adelante"* del bloque *Bifurcación*, y configúrelo para que avance hacia adelante como se muestra en la Figura 241.

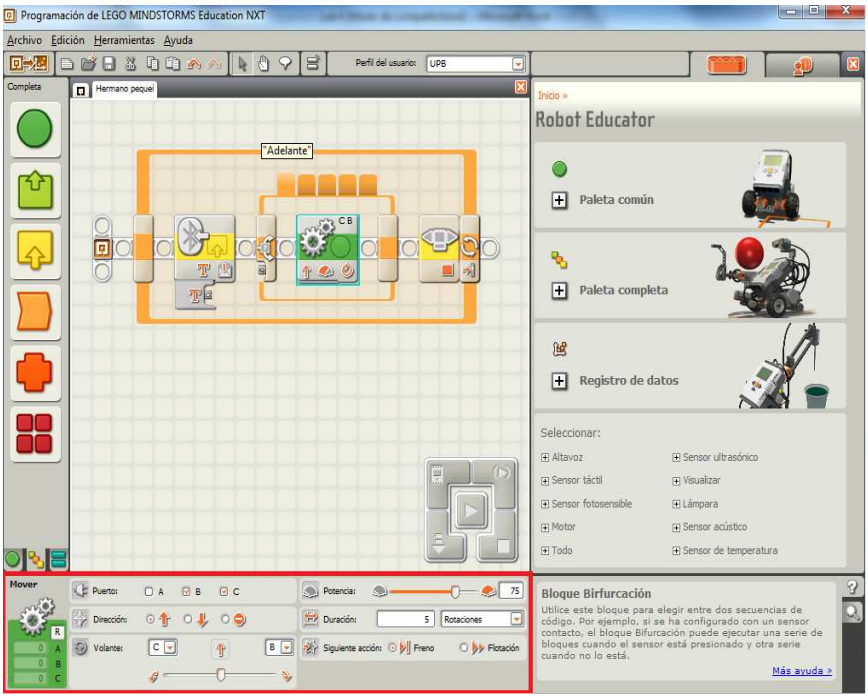



**Figura 241:** Configuración del bloque Mover del caso "Adelante".

Coloque otro bloque *Mover* dentro del caso *"Izquierda"* del bloque *Bifurcación,* y configúrelo como se muestra en la Figura 242.

Coloque otro bloque *Mover* dentro del caso *"Derecha"* del bloque *Bifurcación,* y configúrelo como se muestra en la Figura 243.

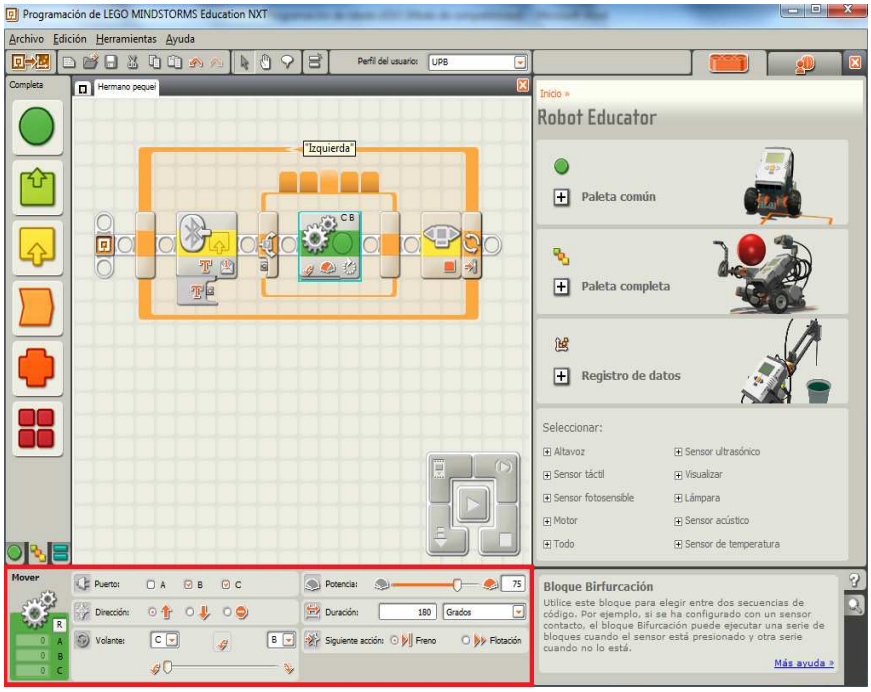



**Figura 242:** Configuración del bloque Mover del caso "Izquierda".

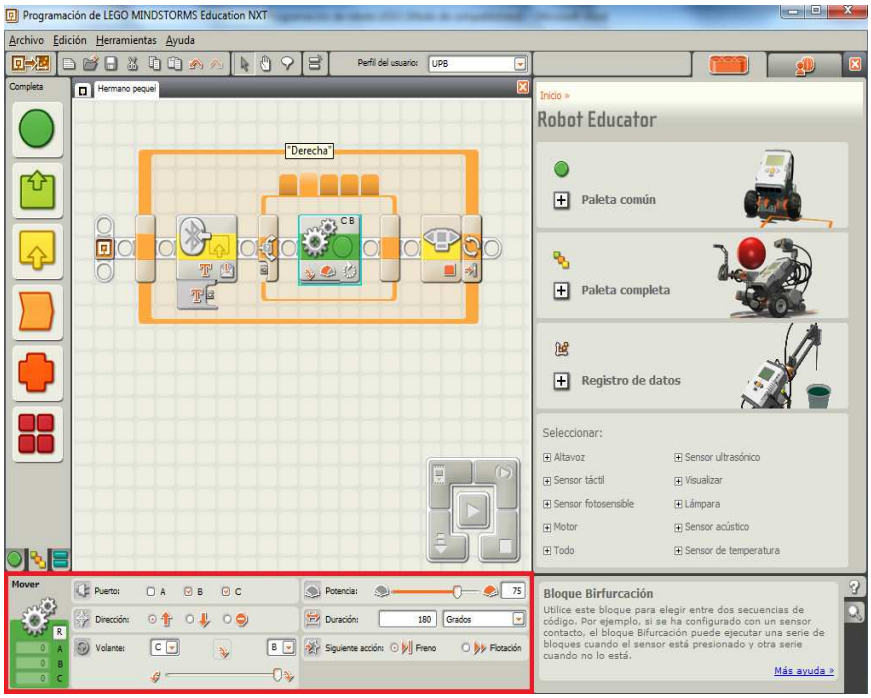



**Figura 243:** Configuración del bloque Mover del caso "Derecha".

Coloque otro bloque *Mover* dentro del caso *"Spin"* del bloque *Bifurcación,* y configúrelo como se muestra en la Figura 244.

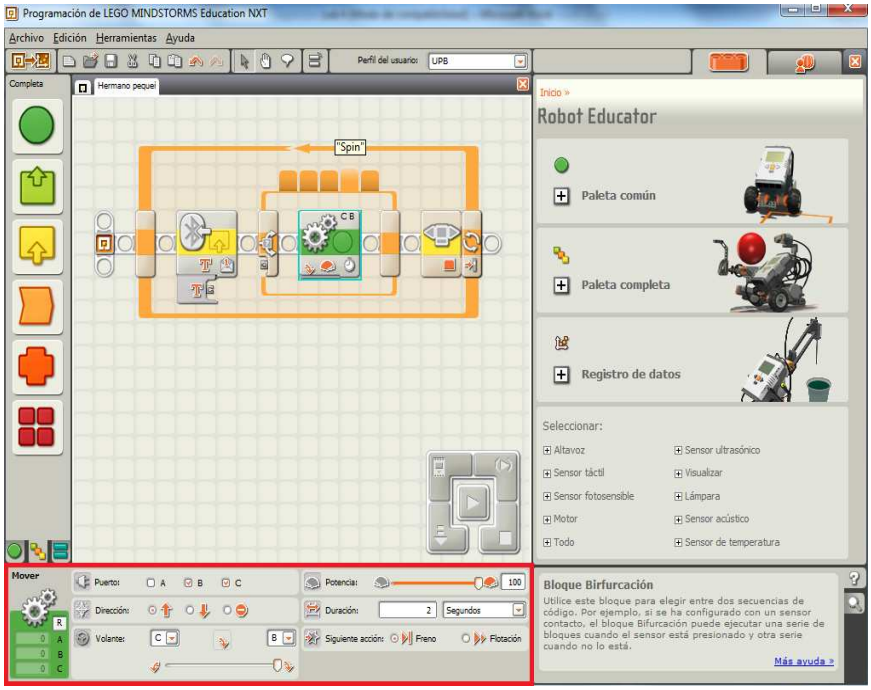



**Figura 244:** Configuración del bloque Mover del caso "Spin".

Conecte la salida del bloque *Recibir mensaje* a la entrada del bloque *Bifurcación*, como se muestra en la Figura 245.

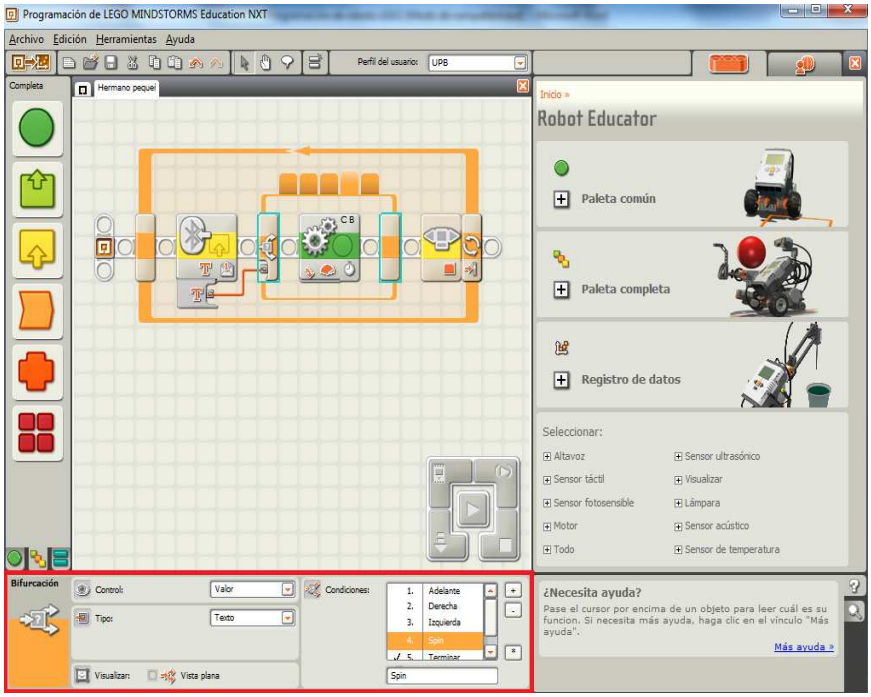



**Figura 245:** Conexión del bloque Recibir mensaje con el bloque Bifurcación.

Coloque bloques de visualización al comienzo de cada caso para ver qué comando está siendo procesado por el *Hermano pequeño*. Configure cada bloque de visualización para que aparezcan las imágenes: "Fordward", " Right 02", "Left 02", "Rotation", y

"Stop 02", respectivamente, en los casos que proceda del conmutador. Utilice un ajuste de posición de 45 para X y 25 para Y, al centro de las imágenes; con excepción del caso “Stop 02” que requiere de un ajuste de 25 para X y 8 para Y.

**Sugerencia:** Usted puede agregar comentarios al código, haga doble clic en cualquier parte del espacio en blanco y escriba su comentario. Así es como el comentario "A D I S T" se ha añadido como se muestra en la Figura 246 para describir las pestañas en el panel de control sin tener que navegar a través de ellas. Esta práctica mejora la legibilidad del código, y es altamente recomendado para la programación.

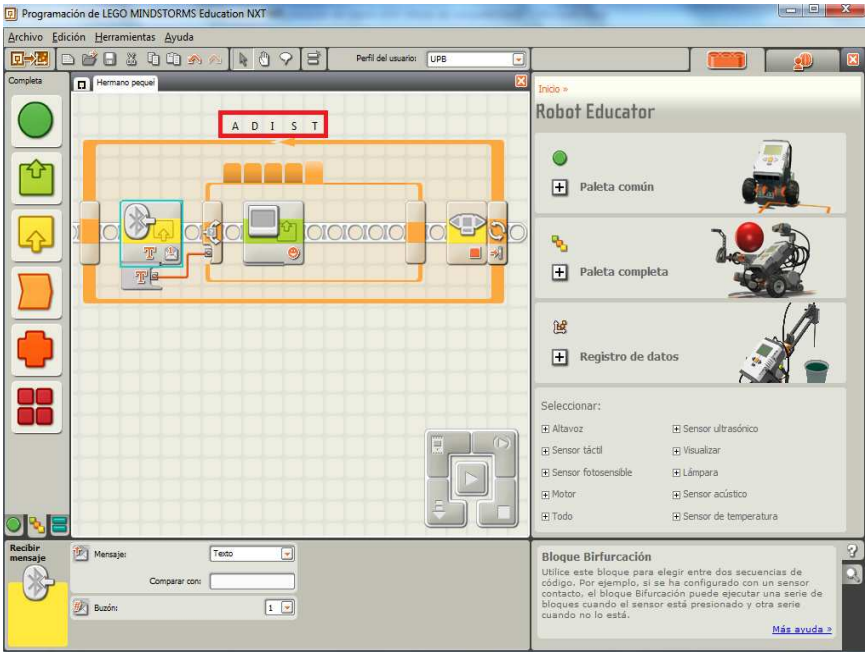



**Figura 246:** Código del Hermano pequeño.

Guarde el programa y descárguelo en el Tribot Hermano pequeño.

Encienda el *Hermano pequeño*, y luego encienda el *Gran Hermano*. Tenga en cuenta que el *Hermano pequeño* imitará el comportamiento del *Gran hermano*, y mostrará los comandos recibidos en la pantalla del controlador NXT. Oprima los botones naranja de los controladores NXT para detener los programas.

**Taller:** Utilice un Tribot como un control remoto para otros Tribots.

El objetivo de este taller es la creación de un mando a distancia de un Tribot que pueda controlar a otro Tribot para llevar a cabo los siguientes comandos:

- ✓ Giros a la izquierda y a la derecha.

- ✓ Acelerar la velocidad por lo menos en la dirección de avance.

Uno de los beneficios de tener la capacidad de comunicación Bluetooth es la creación de un mando a distancia de un Tribot con otro.

Aquí hay un plan sugerido para este taller:

Investigue algunos otros diseños de control remoto para obtener algunas ideas.

Decida qué características del Tribot utilizará para llevar a cabo cada comando.

Use un diagrama de flujo para organizar la estructura general del programa.

Cree un esqueleto del programa antes de escribir en las funciones. Utilice las marcas temporales (es decir, bloques de sonido) para probar el código del esqueleto.

Una vez que la estructura del código funciona correctamente, rellene los huecos con las operaciones de código final.

Trate de controlar dos o más Tribots utilizando sólo un mando a distancia.

# USANDO EL SENSOR FOTOSENSIBLE COMO INTERRUPTOR

En el cuadro desplegable *Perfil de usuario*, seleccione su perfil.

Seleccione *Archivo >> Nuevo* para crear un nuevo programa.

Coloque un bloque *Sensor luz* que se encuentra en la *paleta sensor.* Haga clic en el bloque *Sensor luz*. La ventana de configuración de la Figura 247 debe aparecer en la parte inferior de la interfaz del software NXT.

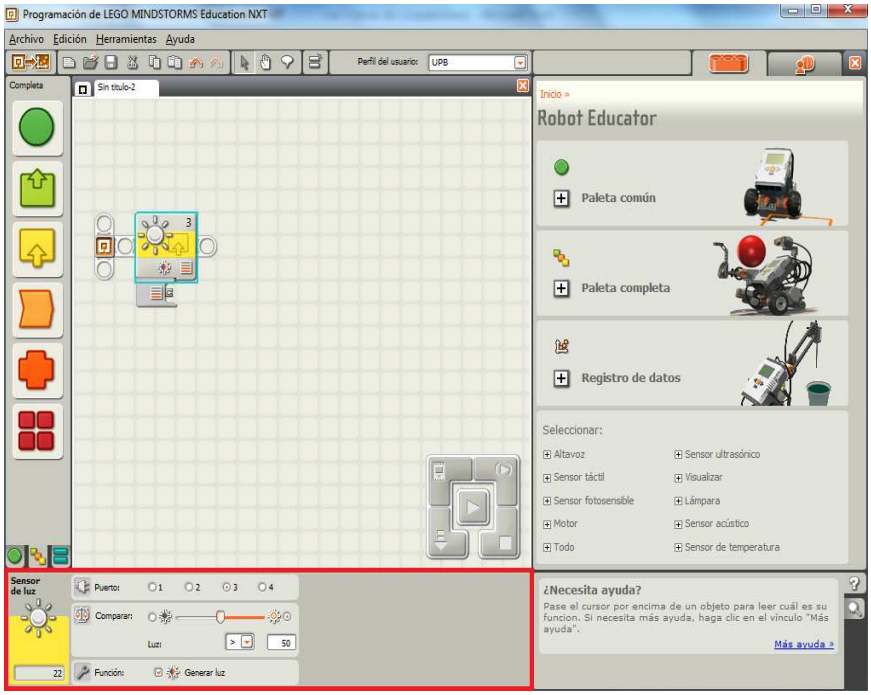



**Figura 247:** Ventana de configuración del bloque Sensor luz.

**Usando el sensor de luz como interruptor.**

Hay dos diferentes maneras en que el sensor de luz se puede utilizar para proporcionar información a un sistema de control de movimiento. Una forma es usar el sensor de luz como interruptor de conmutación digital. Esto se puede lograr mediante la comprobación si el valor del sensor de luz está por encima o por debajo de cierto umbral. A modo de ejemplo, cualquier valor del sensor que está por encima de 50 se considera como un estado "encendido" (superficie blanca), y cualquier valor por debajo de 50 se considera como un estado "off" (superficie de color negro.)

Una forma sencilla de seguir una línea con un sensor de luz es que el sensor sigue a lo largo de un camino, como se muestra a continuación:

**Figura 248:** Seguidor de líneas.

Tenga en cuenta que el sensor no está viajando dentro de la línea, sino a lo largo del borde de la línea. La ejecución del sensor de luz a través del borde de la línea proporciona la información adecuada a un sistema de control de movimiento porque la transición se detecta. La transición de la luz a la oscuridad y de la oscuridad a la luz es la única manera para que el robot sepa si está siguiendo la línea. Si el robot se desplaza a lo largo del centro de la línea o en una hoja grande de papel negro, su lectura siempre será "apagado" durante todo el tiempo que se mueva. Esto daría lugar a opiniones de mala posición para el robot, y haría que  vagará sin rumbo.

Suponiendo que el robot sigue el borde superior de la línea en todo momento, se puede hacer ciertos supuestos:

- ✓ Cada vez que una transición de "on" a "off" se detecta, el robot debe haber cruzado la línea.
- ✓ Cada vez que una transición de "off" a "on" se detecta, el robot debe estar a la izquierda de la línea.

Por lo tanto, él puede ordenarle al robot para que se encienda y en consecuencia permanezca a lo largo del borde de la línea para:

- Girar a la izquierda hasta que una transición de "off" a "on" se detecta.
- Girar a la derecha hasta que una transición de "on" en "off" se detecta

El diagrama de la Figura 249 muestra los comandos que deben ser ejecutados para girar a la izquierda o a la derecha conforme el sensor de luz toma lecturas.

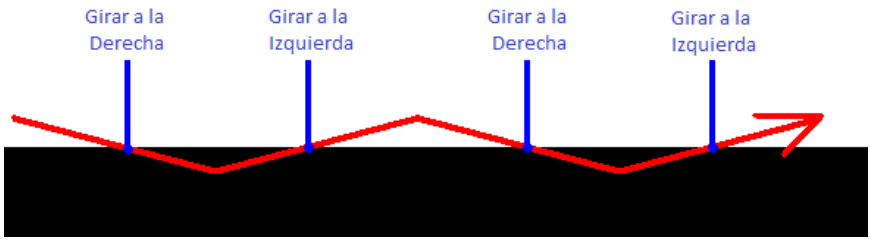



**Figura 249:** Lecturas tomadas por el sensor de luz.

Tenga en cuenta que estos comandos sólo son válidos si el robot está en el borde izquierdo (borde superior, en la Figura 249) de la línea. Si el robot sigue el borde derecho (borde inferior, en la Figura 249) de la línea no funcionará, a menos que la lógica para dar vuelta se invierta.

Para el desarrollo óptimo de este capítulo usted requiere del Lego NXT 8527/8547 Mindstorms Test Pad x1624, que es un circuito de prueba como el que se muestra en la Figura 250.

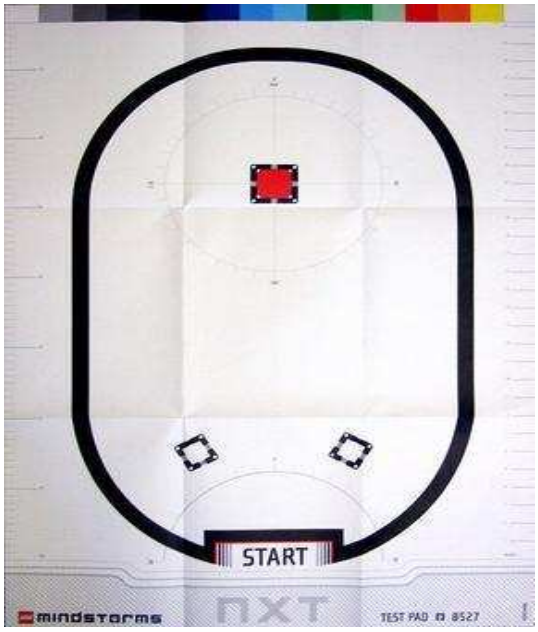



**Figura 250:** Lego NXT 8527/8547 Mindstorms Test Pad x1624

Con todo esto dicho, ¡construya un programa seguidor de líneas!

Cree un nuevo programa en el software NXT 2.1 Programming.

Guarde el programa como *"Seguidor de líneas"*.

Una variable lógica se utiliza para alternar entre girar a la izquierda y a la derecha. Coloque un bloque *Variable* que se encuentra en la *paleta Datos*. Este bloque servirá para establecer el estado inicial de la lógica como "verdadero". Configurar el bloque como se muestra en la Figura 251.

En la práctica de programación, siempre es una gran idea inicializar todas las variables en un estado conocido antes de que sean utilizadas para evitar que los datos basura que pueden estar presentes generen una confusión en el programa.

Coloque un bloque *Bucle* después del bloque *Variable*. Configúrelo como se muestra en la Figura 252.

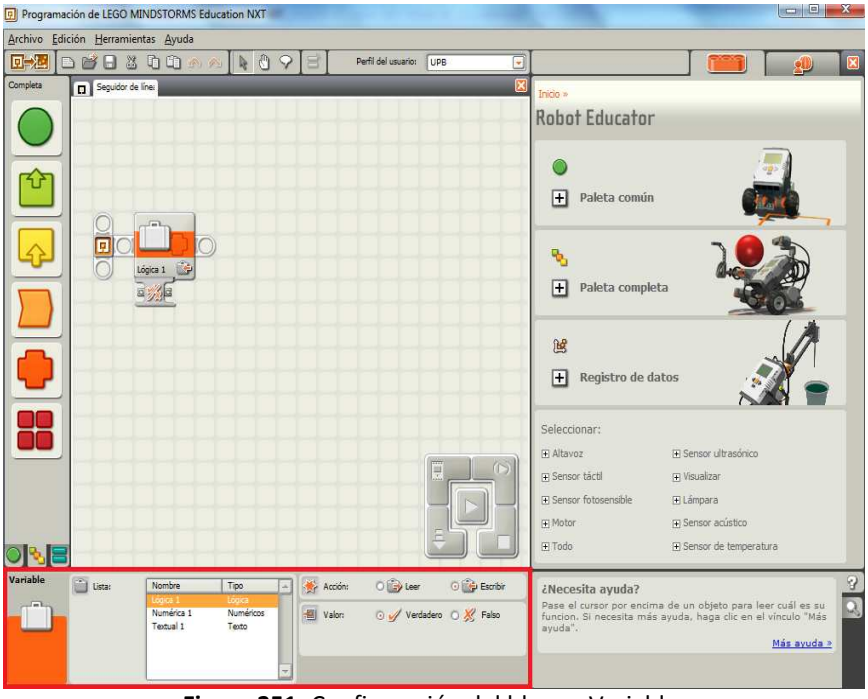



**Figura 251:** Configuración del bloque Variable.

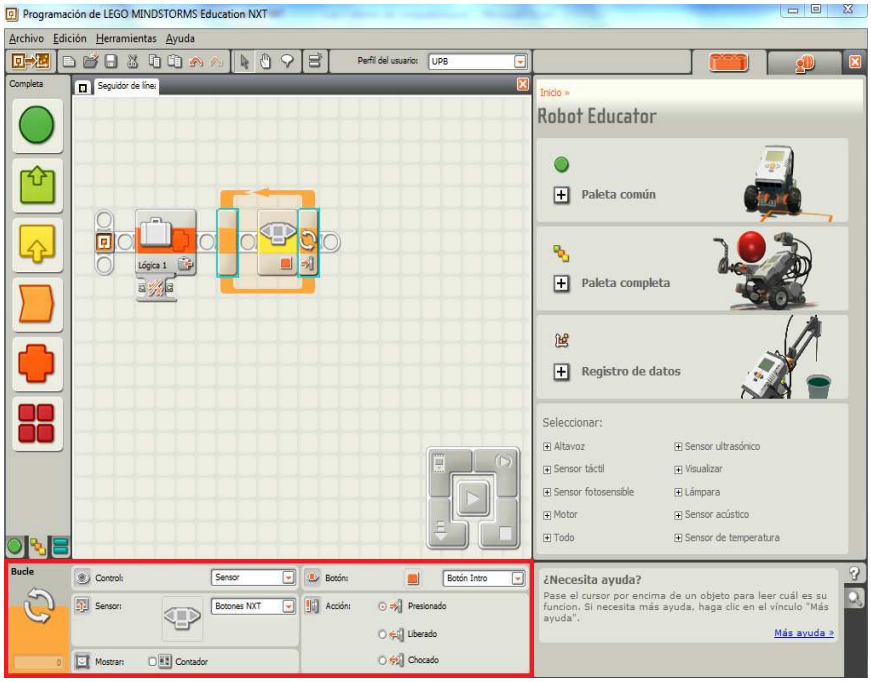



**Figura 252:** Configuración del bloque Bucle.

Coloque un bloque *Variable* dentro del bloque *Bucle*. Configúrelo para leer el contenido de Lógica 1.

Inserte un bloque *Bifurcación* después del bloque *Variable* y configúrelo como se muestra en la Figura 253.

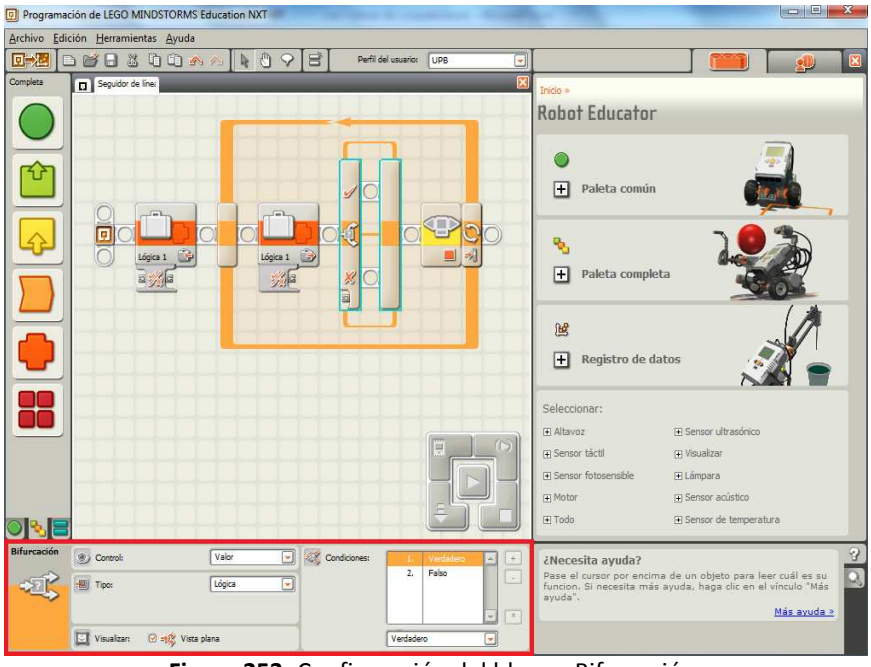



**Figura 253:** Configuración del bloque Bifurcación.

Cablee la salida del bloque *Variable* con la entrada en el borde inferior izquierdo del bloque *Bifurcación*. Esto permite que la *Variable* pueda alternar entre los casos dentro del bloque *Bifurcación*.

Coloque un bloque *Mover* dentro del caso *"verdadero"* del bloque *Bifurcación*, y configúrelo como se muestra en la Figura 254.

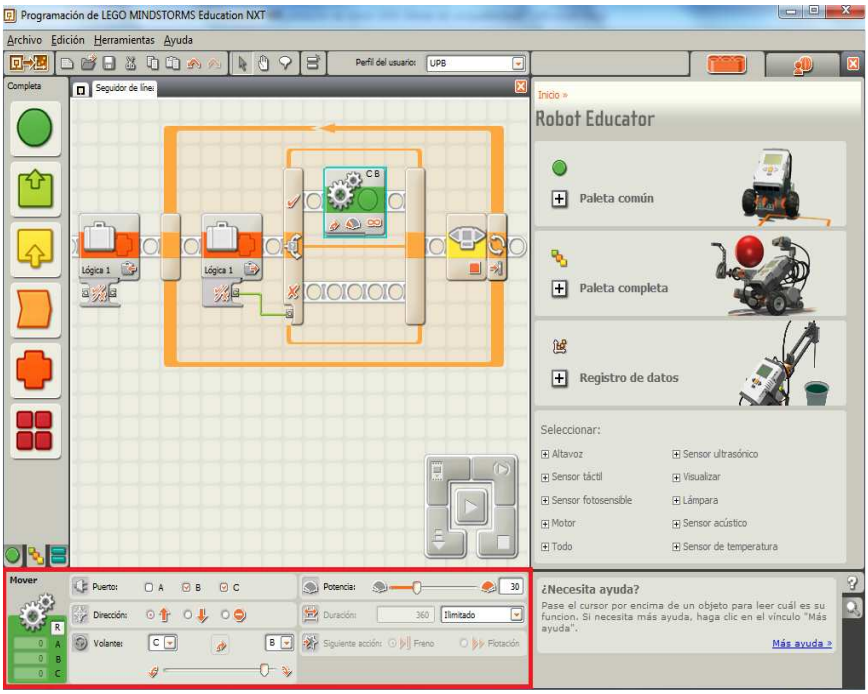



**Figura 254:** Configuración del bloque Mover.

Coloque otro bloque *Mover* dentro del caso *"falso"* del bloque *Bifurcación*, y configúrelo como se muestra en la Figura 255.

Tenga en cuenta que el bloque de la izquierda tiene más potencia que el bloque de la derecha. Esto fue intencional para que responda mejor a las curvas de la línea.

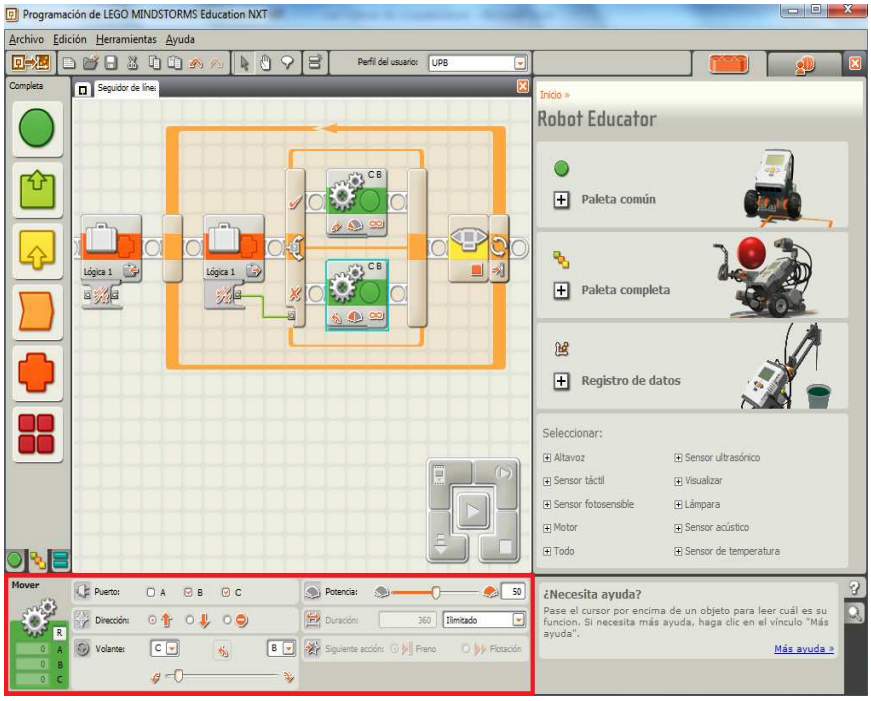



**Figura 255:** Configuración del segundo bloque Mover.

Tenga en cuenta que este programa de ejemplo está diseñado con las siguientes características:

- ✓ El robot se pondrá a prueba en un circuito que se extrae del Lego NXT 8527/8547 Mindstorms Test Pad x1624, que es un circuito como el que se muestra en la Figura 250.
- ✓ El Tribot va a seguir a lo largo del borde exterior del circuito.
- ✓ El Tribot se mueve alrededor de la malla en el sentido de las agujas del reloj.
- ✓ El Tribot se alinea en el comienzo de una parte recta del circuito.
- ✓ La orientación inicial del Tribot será paralela a la línea del circuito.

Coloque un bloque *Espera* después del bloque *Mover* en el caso *"verdadero"* del bloque *Bucle*, y configúrelo como se muestra en la Figura 256.

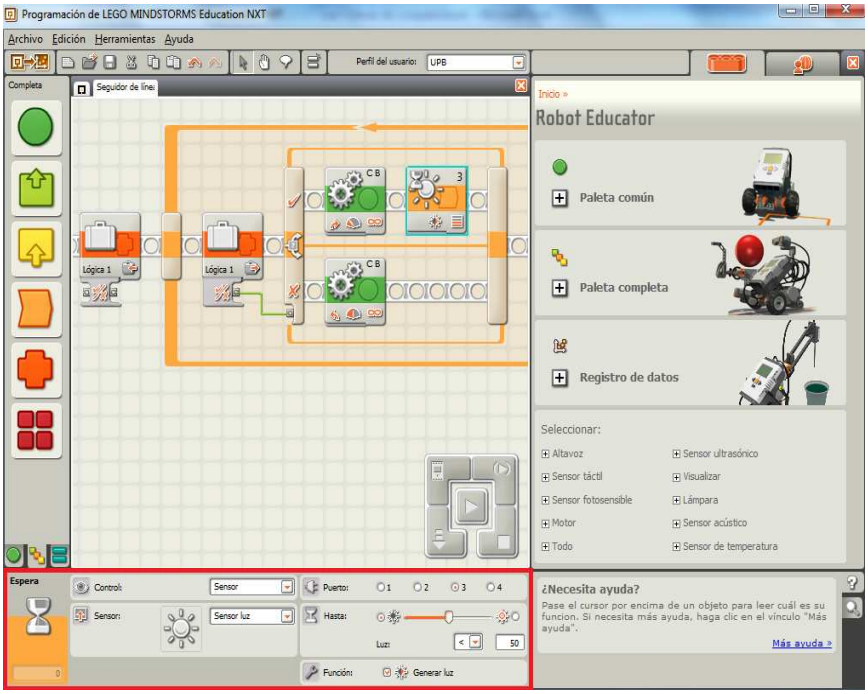



**Figura 256:** Configuración del bloque Espera.

Coloque otro bloque *Espera* después del bloque *Mover* en el caso *"falso"* del bloque *Bifurcación*, y configúrelo de la misma manera que el bloque *Espera* anterior, pero cambie el símbolo Menor que ( < ) por un Mayor que ( > ).

Coloque un bloque *Variable* después de cada uno de los bloques *Espera* en ambos casos del bloque *Bifurcación*. Ambos bloques *Variable* escriben un valor Lógica 1. El caso "verdadero" del bloque *Bifurcación* escribirá una lógica con valor *Falso*, y el caso falso del bloque Bifurcación escribirá una lógica con valor *Verdadero*.

El código del programa se muestra en la Figura 257.

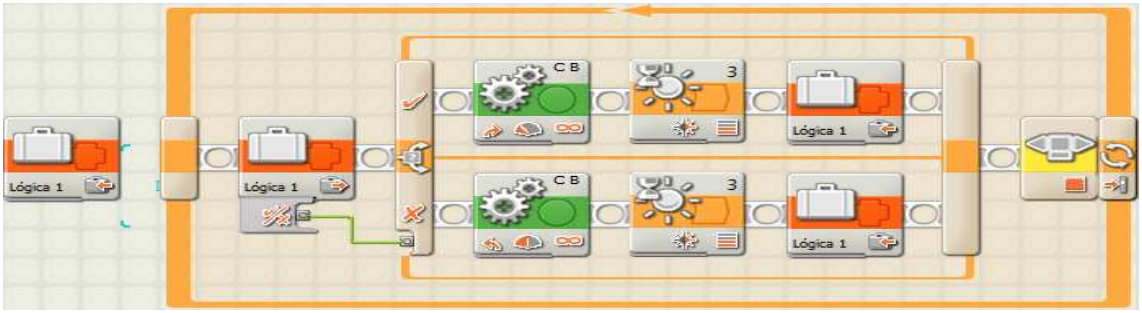



**Figura 257:** Código del programa.

Guarde su programa y descargarlo en el Tribot.

Coloque el Tribot en el Lego NXT 8527/8547 Mindstorms Test Pad x1624, a lo largo de un tramo recto.

Ejecute el programa, y vea cómo el robot intenta seguir el borde de la línea.

**Usando el sensor de luz como sensor analógico.**

Para utilizar la lectura analógica adecuada del sensor de luz, es importante entender cómo se decide cuál es el valor que va a utilizar. Para mantener este simple concepto, supongamos que el sensor de luz tiene la intensidad media de luz que se detecta en un área pequeña. Por lo tanto, como el sensor pasa por encima del borde de una línea negra, la línea cubrirá esta pequeña área en el tiempo. Esto efectivamente reduce el valor de la intensidad devuelto por el sensor. Lo mismo ocurrirá cuando se mueve de un área oscura a un área de luz. La Figura 258 ilustra este concepto.


**Figura 258:** Funcionamiento del sensor de luz de forma analógica.

Este comportamiento permite un control más preciso apuntando a un valor que esta entre blanco y negro. Supongamos que el rango de valores posibles de los sensores es representado como sombras de gris. La Figura 259 es una interpretación de la orilla de una línea hasta el sensor de luz.

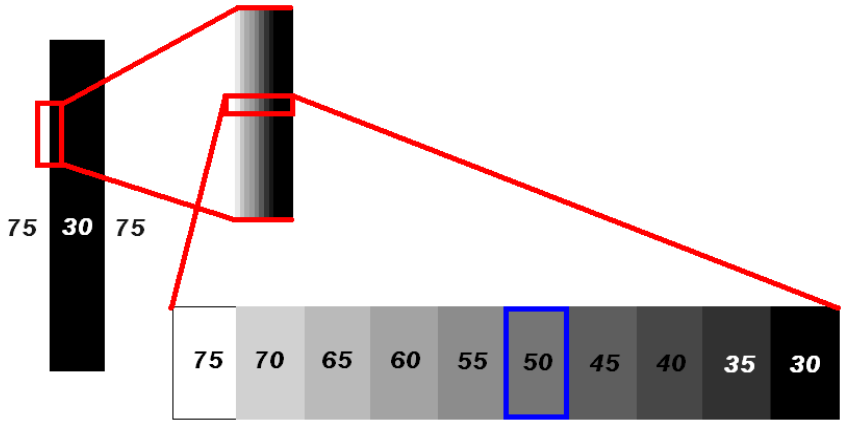



**Figura 259:** Escala de grises.

Los números representan los valores de intensidad que sería devuelto si el sensor de luz se coloca sobre determinado matiz de gris. Escoger un valor medio de 50 como el límite objetivo sería conceder un mayor control del robot debido a que más información sobre la ubicación del sensor se devuelve al programa.

Para ofrecer un rendimiento más preciso, todo lo que hay que hacer es una resta entre la lectura actual del sensor y la lectura ajustada para el sensor. En este caso, la lectura del sensor se establece en 50.

A continuación se construirá un programa sencillo que muestra este concepto.

Cree un nuevo programa de NXT, y guardarlo como "Sensor de luz analógico"

Coloque un bloque *Bucle*, y deje su configuración predeterminada.

Coloque un bloque *Sensor luz* en el interior del bloque *Bucle*, y déjelo en su estado predeterminado.

Coloque un bloque *Matemáticas* después del bloque *Sensor luz*, y configúrelo para que realice una resta.

Conecte la salida de la intensidad del bloque *Sensor luz* a la entrada A del bloque *Matemáticas*.

Seleccione el bloque *Matemáticas*, y coloque *"50"* para la entrada de B. 50 será la lectura deseada del sensor, o punto de referencia, para el algoritmo de la siguiente línea.

El resultado de este cálculo será la diferencia entre el lugar donde está el robot, y el lugar en el que debería estar. Esta información puede ser usada para controlar directamente la dirección y la potencia de las ruedas. Desde esta resta puede producir una respuesta negativa, la dirección de las ruedas también se puede activar y desactivar.

Coloque otro bloque *Matemáticas* después del bloque *Matemáticas* resta, y configúrelo para realizar una multiplicación.

Conecte la salida de Resultados del bloque *Matemáticas* resta a la entrada A del bloque *Matemáticas* multiplicación.

Seleccione el bloque *Matemáticas* multiplicación, y el coloque B igual a *"2"*. Esto actuará como un amplificador de la fuerza con la que las ruedas deben girar para corregir su trayectoria.

Con el fin de estar seguros de que el robot se mueve siempre hacia adelante, coloque otro bloque *Matemáticas* después del bloque *Matemáticas* multiplicación, y configúrelo para que se agreguen *"30"* al resultado de multiplicar.

El último paso es determinar si las ruedas deben girar hacia adelante o hacia atrás. Coloque un bloque *Comparación* después del bloque *Matemáticas* agregar, y conecte el resultado del bloque agregar a la entrada A del bloque *Comparación*. Configure el bloque *Comparación* para que se compruebe si A es mayor que 0.

Coloque un bloque *Motor* después del bloque *Comparación*. Determine que la salida del motor corresponde con la rueda izquierda (por lo general su motor C) y configure el bloque *Motor* a la rueda de comandos.

Ampliar el gabinete del bloque *Motor* haciendo clic en el borde inferior izquierdo del bloque.

Conecte la salida de Resultados del bloque Comparación a la entrada de Dirección del bloque del motor.

Conecte la salida de Resultados del bloque *Matemáticas* agregar a la entrada de potencia del bloque *Motor*.

Coloque otro bloque *Motor* después del bloque *Motor* anterior, y configúrelo para el control de la otra rueda (por lo general el motor B) para ir siempre hacia adelante con una potencia de *"30"*.

El programa final debería observarse como la Figura 260.

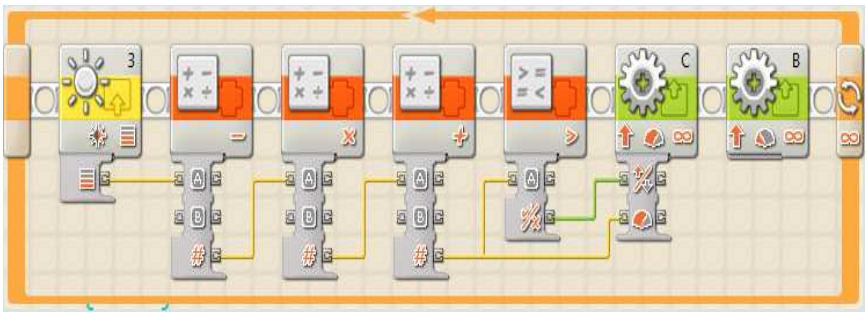



**Figura 260:** Código del programa para usar el sensor de luz como un sensor analógico.

Guarde el programa y descárguelo en el robot.

Coloque el robot en el terreno de prueba, en el borde izquierdo de la línea negra, y ejecute el programa.

# PROGRAMACIÓN CON LEJOS Y NETBEANS

LeJOS es una pequeña JVM (Máquina Virtual para Java) que permite ejecutar código Java dentro del controlador del Lego Mindstorms. Esta implementación de Java nació producto de TinyVM una maquina diseñada para ejecutar código en la versión RCX, que luego se convirtió en leJOS. Adicional a la máquina virtual se cuenta con una API que implementa el núcleo de clases en Java, algunas de J2ME y otras adicionales para la comunicación a través de Bluetooth, acceso a GPS y sensores y actuadores.

La versión para el NXT llamada leJOS NXJ cuenta con:

- ✓ Soporte para programación orientada a objetos.
- ✓ Multitarea.
- ✓ Arreglos multidimensionales.
- ✓ Recursión.
- ✓ Sincronización.
- ✓ Excepciones.
- ✓ Tipos de datos Java.
- ✓ Un API para robots.

También cuenta con herramientas para subir el código al controlador NXT.

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de Java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software.

La Plataforma NetBeans es una base modular y extensible usada como una estructura de integración para crear aplicaciones de escritorio grandes. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones.

La plataforma ofrece servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación. Entre las características de la plataforma están:

- ✓ Administración de las interfaces de usuario.
- ✓ Administración de las configuraciones del usuario.
- ✓ Administración del almacenamiento.
- ✓ Administración de ventanas.
- ✓ Framework basado en asistentes.

# BIBLIOGRAFÍA

Bernat Romani. "Jugar con las máquinas". Ed. Ters Torres/Edunsa. 1999.

Dra. Martha N. Cyr. “ROBOLAB. Empezando Guía del profesor para el Software ROBOLAB” Ed. LEGO Dacta 1999.

Dra. Martha N. Cyr. “ROBOLAB. Empezando 2 Guía del profesor para el Software ROBOLAB” Ed. LEGO Dacta 1999.

Antonio M. Lázaro. “LabVIEW 6i. Programación Gráfica para el Control de Instrumentación”. Ed. Paraninfo. 2001.

José Mª Angulo, Susana Romero e Ignacio Angulo. “Microbótica”. Ed. Paraninfo. 1999.

Clive Gifford. “Robots. Descubre cómo funcionan estas máquinas”. SM Saber. 1998.

Mario y Giulio Ferrari. "Building Robots with LEGO MindStorms". Syngress. 2002.

Robótica LEGO – LEGO Robotics, disponible en: http://www.julio.sandria.org/robotica/lego-mindstorms/31-robotica-lego-lego-robotics.html

NXTMastery Mastering LabVIEW with LEGO Mindstorms, disponible en: http://nxtmastery.com/