

## MANUAL DE PRÁCTICAS DE LABORATORIO DE: INTERFACES HOMBRE MAQUINA

**NOMBRE:** \_\_\_\_\_

**CARRERA Y CODIGO:** \_\_\_\_\_

**FECHA:** diciembre del 2020

**PROFESOR:** Ing. Francisco Javier Flores Gómez

## **Práctica 1:** Introducción a la programación de LabVIEW

**Objetivo:** identificar, conectar y comprobar las funciones básicas reconociendo los diversos tipos de controles e indicadores en las dos ventanas de programación.

**Introducción:** se clasifican según su estilo y pueden ser *modern*, *system* y *classic*, con submenús según el tipo de datos. En los controles e indicadores éstos son los elementos que permiten operar como entradas y salidas de información para variar o modificar valores, son de tipo *numérico*, *booleano*, *textos* y *tipos compuestos*.

*Los datos numéricos*, se dividen en enteros y de coma flotante con diferentes tamaños en cada uno. *Los booleanos*, sólo pueden ser verdadero (true) o falso (false) y son apropiados para crear botones. *Los tipos de datos compuestos* con *arrays* (arreglos) de números y booleanos. Son listas ordenadas de valores y los *cluster* son un conjunto desordenado de otros datos

En los controles e indicadores se pueden seleccionar las terminales que permiten su conexión de manera directa según su tipo, que servirán para interactuar con el usuario., y se dividen en controles o entradas e indicadores salidas.

Para graficar las más importantes son *waveform chart* o con memoria se anexa el nuevo dato a los existentes y los *waveform graph* estos son sin memoria que dibujan totalmente la gráfica cuando llegan nuevos datos. En los ejes los datos pueden ajustarse o modificarse los valores máximos.

Se recomienda colocar una etiqueta en todos los controles e indicadores con el nombre de la función a desempeñar o para identificar al elemento.

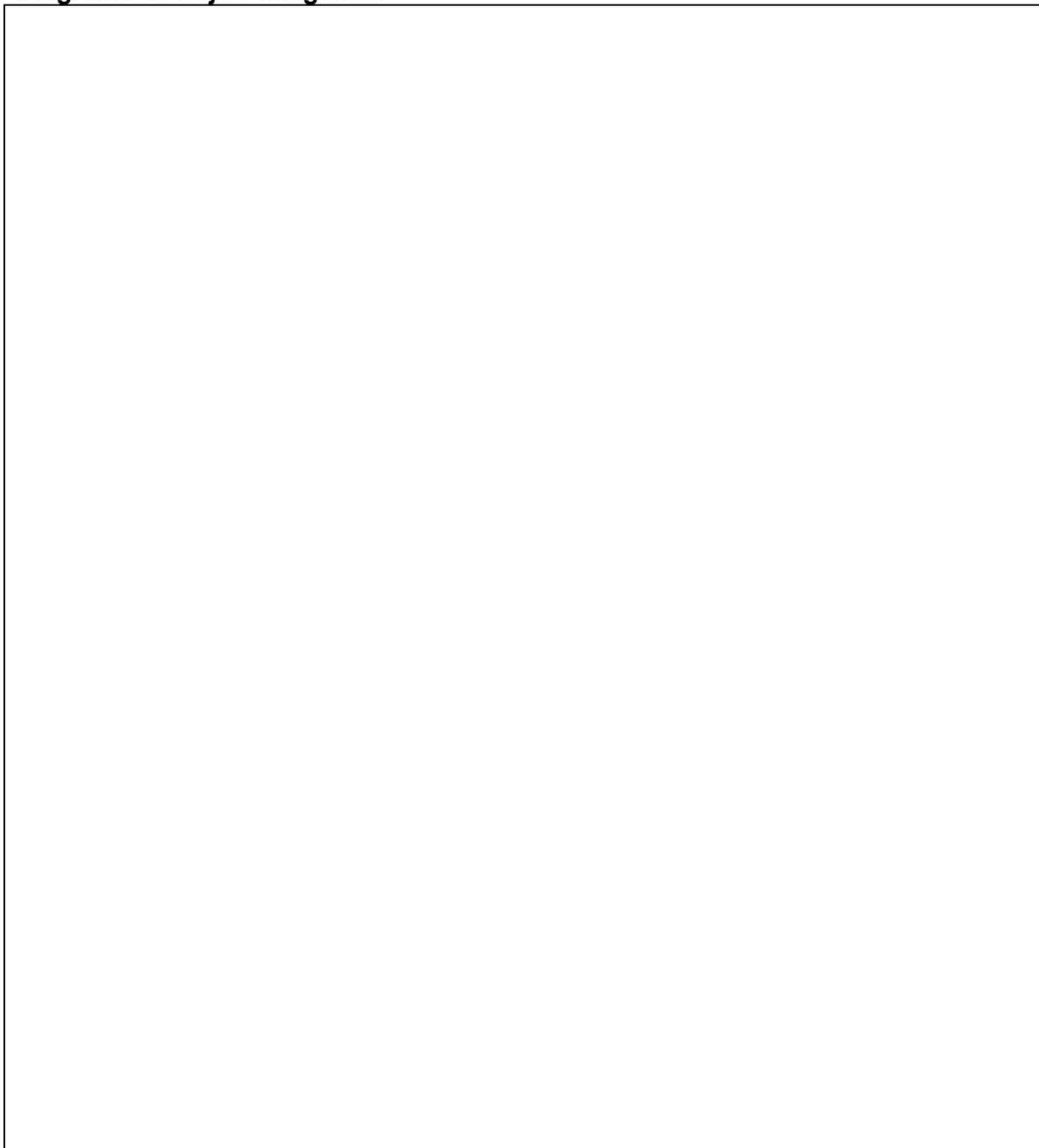
**Bibliografía:** LabVIEW 8.20 Entorno gráfico de programación, José Rafael Lajara Vizcaíno y José Pelegrí Sebastián, Alfaomega-Marcombo

**Procedimiento:** el alumno conectará las entradas y salidas a partir de los controles e indicadores de los diversos tipos, en programación gráfica realizando al menos cinco operaciones básicas agregando las conexiones e introducirá los valores numéricos y lógicos comprobando su funcionamiento. Partiendo de las funciones y menús.

### **Material necesario:**

- Computadora
- Software LabVIEW
- Diagrama de identificación
- Manual

**Diagrama de flujo e imágenes:**



**Conclusión:**

**Práctica 2:** Control de ejecución a la programación de LabVIEW

**Objetivo:** identificar, conectar y comprobar la secuencia de programación y ejecución en el software correspondiente.

**Introducción:** se clasifican según en controles e indicadores éstos son los elementos que permiten modificar los valores en entradas y salidas de información o datos y en estructuras que controlan bajo alguna cualidad según sea, las cuales se ejecutan de arriba hacia abajo y de izquierda a derecha, de acuerdo al formato de edición.

En los controles e indicadores se pueden seleccionar las terminales que permiten su conexión de manera directa según su tipo, que servirán para interactuar con el usuario.

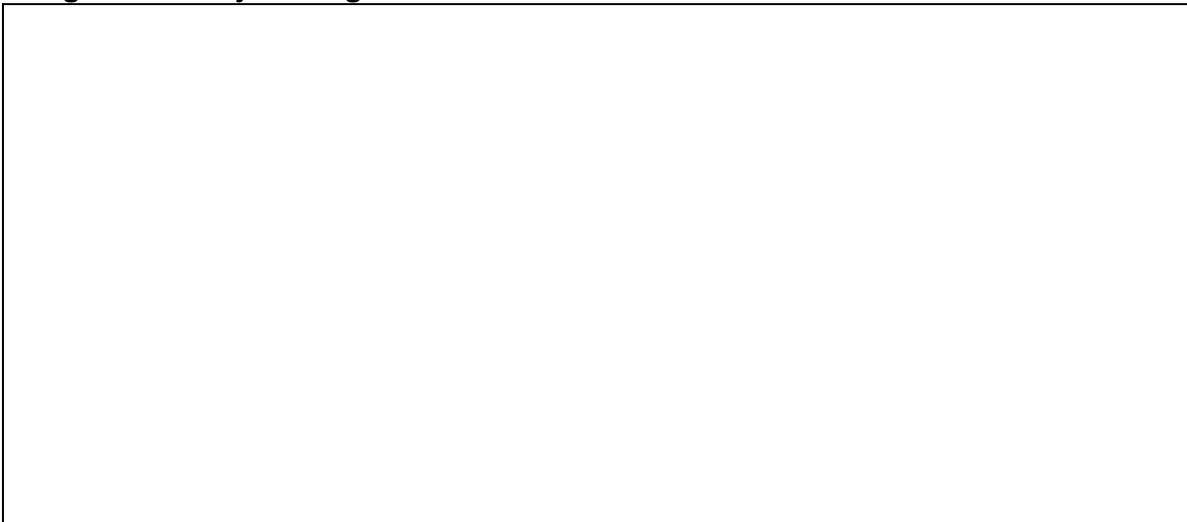
**Bibliografía:** LabVIEW 8.20 Entorno gráfico de programación, José Rafael Lajara Vizcaíno y José Pelegrí Sebastián, Alfaomega-Marcombo

**Procedimiento:** el alumno conectará las entradas y salidas a partir de los controles e indicadores de los diversos tipos, en programación gráfica realizando modificaciones en el orden de edición y conexión comprobando su funcionamiento. Partiendo de las funciones y menús.

**Material necesario:**

- Computadora
- Software LabVIEW
- Diagrama de identificación
- Manual

**Diagrama de flujo e imágenes:**



**Conclusión:**

**Práctica 3:** Programación de LabVIEW con estructuras.

**Objetivo:** conexión y comprobación de las estructuras básicas que permiten un control en la ejecución del programa.

**Introducción:** las funciones que operan en el diagrama de bloques con dos ventanas, una que opera como panel frontal o de control para el usuario, y la segunda que opera como zona de conexión donde el programador realiza la función a realizar, con varios submenús que se dividen en función de su aplicación como estructura, que controlan el flujo de ejecución. Contiene elementos que son equivalentes a las *instrucciones de control* de los lenguajes convencionales de programación.

La estructura **secuencia** (o sequence) sirve para *secuenciar* el orden de ejecución del código que está en su interior, con un orden. En ésta estructura, una función se ejecuta cuando tiene disponibles *todos los datos* de entrada, y en los lenguajes convencionales se ejecutan en orden de aparición. Se produce de esta manera una dependencia de datos que hace que la función que recibe un dato directo (o indirectamente) de otra se ejecute siempre después, creando un flujo de programa. Mediante Stacked Sequence en la cual sólo aparece un marco de varios a la vez, Flat Sequence en ésta aparecen todos los marcos en la ventana de programación simultáneamente.

La estructura **caso** (o case) se *usará en aquellas situaciones en el que el número de alternativas disponibles sean do o más*. Según qué valor tome el selector dentro de los  $n$  valores posibles. Se ejecutará en correspondencia uno de los  $n$  subdiagramas.

La estructura case consta una ventana de las cuales está dentro de un case o suceso y etiquetado por un identificador del mismo tipo que el selector, y contiene dos sentencias: uno cuando la condición es verdadera para ejecutar el caso de otra manera si el caso es falso. Dos cuando el selector permite ejecutar caso 0,1, 2, 3,...  $n$ : ejecutar caso  $n$  ( $2^{31} - 1$ ).

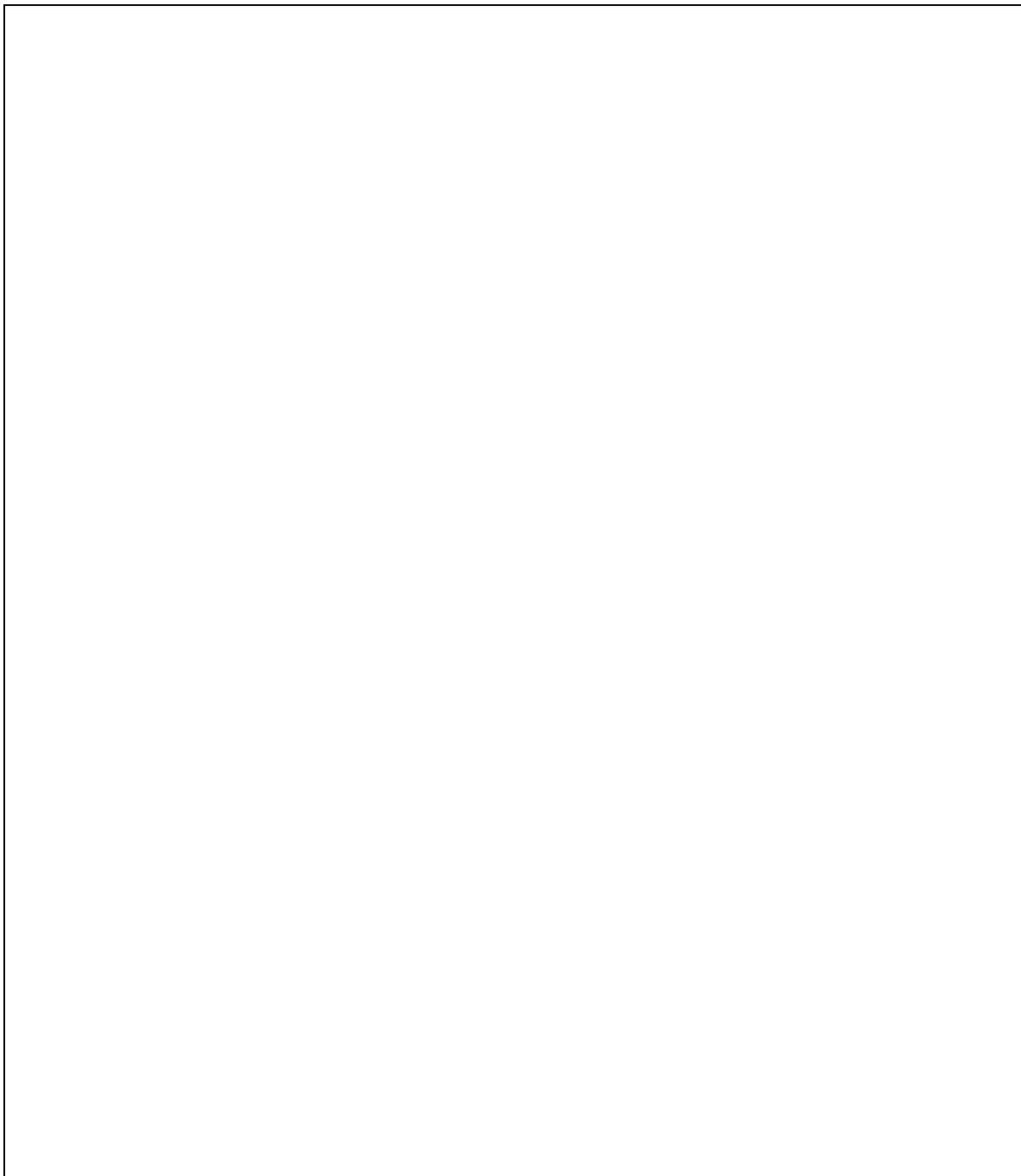
**Bibliografía:** LabVIEW 8.20 Entorno gráfico de programación, José Rafael Lajara Vizcaíno y José Pelegrí Sebastián, Alfaomega-Marcocombo.

**Procedimiento:** el alumno programará la estructura case con verdadero-falso y en varios casos, Sequence staked y flat, con sus entradas y salidas comprobando su aplicación.

**Material necesario:**

- Computadora
- Software LabVIEW
- Diagrama de identificación

**Diagrama de flujo e imágenes:**



**Conclusión:**

**Práctica 4:** Programación de LabVIEW continuación.

**Objetivo:** conexión y comprobación de las estructuras básicas que permiten un control en la ejecución del programa.

**Introducción:** La estructura **While Loop** que *repetirá el código en su interior hasta que se cumpla una condición*, la cual es evaluada en cada iteración, ejecuta un subprograma hasta que se cumpla una condición, y siempre se ejecuta al menos una vez. En la ejecución del subprograma hasta que el terminal condiciona (entrada) "Stop If True", reciba un valor booleano dado.

La estructura **For Loop** se utilizará el bucle que repite el código de su interior un número de veces *previamente establecido*. Se usa cuando queremos que una operación se repita un número determinado de veces. Su equivalente en lenguaje de programación es For i=0 to N-1.

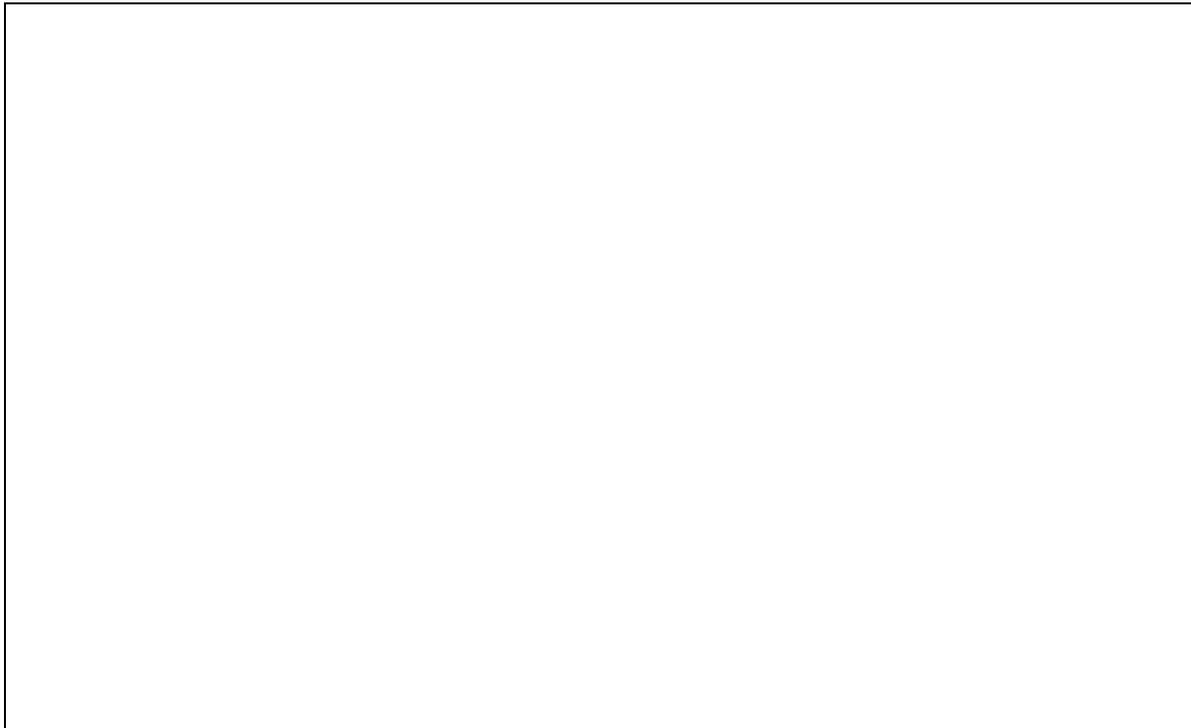
**Bibliografía:** LabVIEW 8.20 Entorno gráfico de programación, José Rafael Lajara Vizcaíno y José Pelegrí Sebastián, Alfaomega-Marcombo.

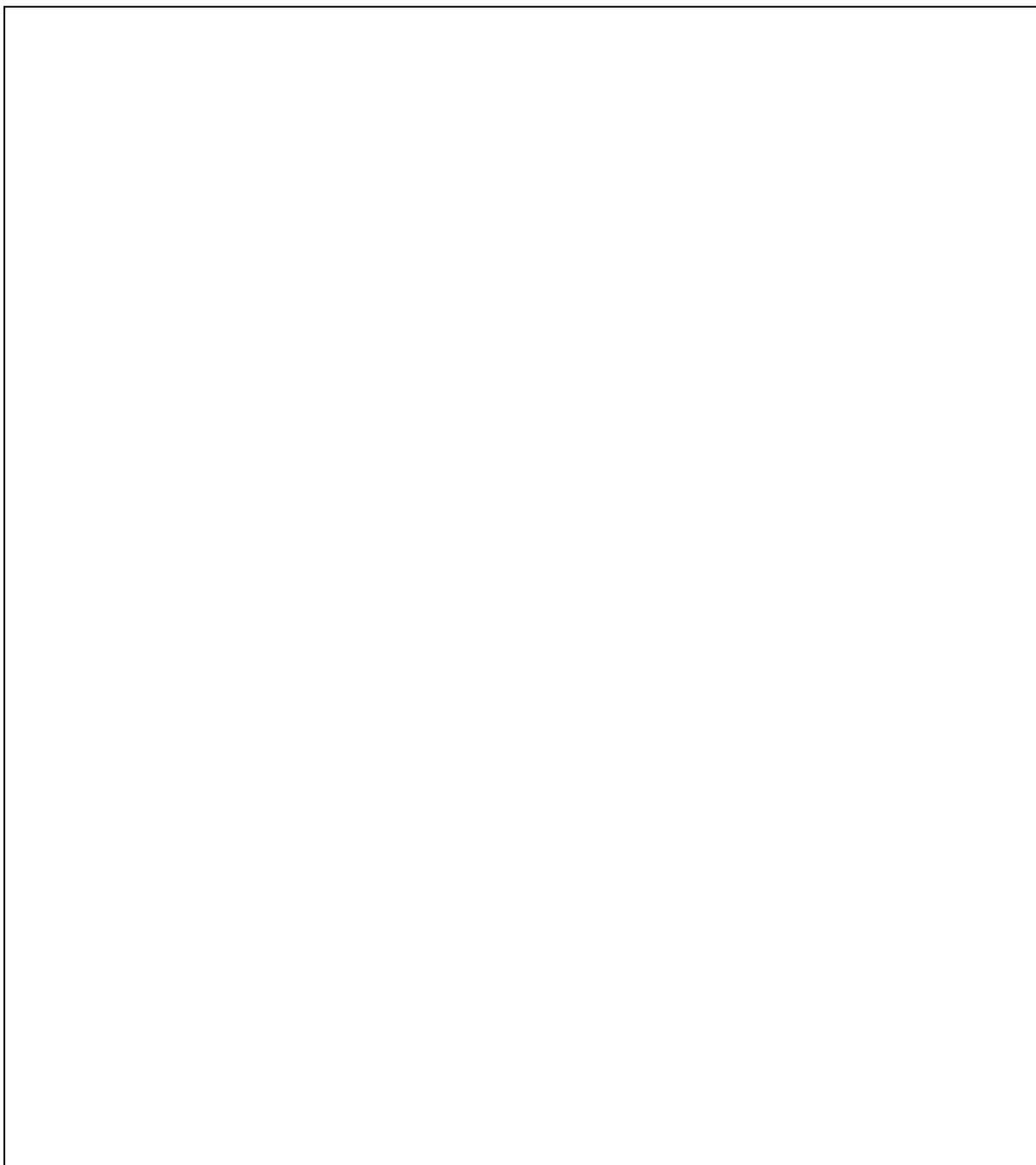
**Procedimiento:** el alumno programará las estructuras **While** y **For loop** con sus entradas y salidas comprobando su aplicación.

**Material necesario:**

- Computadora
- Software LabVIEW
- Manual

**Diagrama de flujo e imágenes:**





**Conclusión:**

**Práctica 5:** Programación de LabVIEW continuación

**Objetivo:** conexión y comprobación de las estructuras básicas que permiten un control en la ejecución del programa.

**Introducción:** La estructura **evento** (o Event) este tipo al igual que case tienen varios subdiagramas en este menú también se tiene una condición que hace que el código del diagrama correspondiente se ejecute. Evento *detiene la ejecución del programa hasta que se da esa condición*, es decir, congela el programa hasta que ocurre un evento o transcurre un intervalo de tiempo.

Utiliza un **formula node** (nodo) cuando se requiere ejecutar fórmulas matemáticas, es una función de características similares a las anteriores, pero en lugar de contener un subdiagrama, contiene una o más fórmulas separadas por un punto y coma. Se usará Formula Node cuando queramos ejecutar fórmulas matemáticas que serían complicadas de crear utilizando las diferentes herramientas matemáticas que LabVIEW incorpora en sus librerías. La fórmula node no controla el flujo de ejecución, sino que evalúa una expresión matemática escrita como texto con una sintaxis parecida al lenguaje C. Las sentencias son asignaciones que usan operadores o funciones, declaración de variables, sentencias de condición.

**Bibliografía:** LabVIEW 8.20 Entorno gráfico de programación, José Rafael Lajara Vizcaíno y José Pelegrí Sebastián, Alfaomega-Marcombo.

**Procedimiento:** el alumno programará las estructuras Event y formula nodo con sus entradas y salidas comprobando su aplicación.

**Material necesario:**

- Computadora
- Software LabVIEW

**Diagrama de flujo e imágenes del programa:**



**Conclusión:**

**Práctica 6:** Programación de LabVIEW continuación

**Objetivo:** conexión y comprobación de las estructuras básicas que permiten un control en la ejecución del programa.

**Introducción:** Utiliza un **Array** (arreglo) es una colección de datos del mismo tipo, que pueden ser numérico, booleano, directorios, cadenas y formas de onda. Puede tener una o más dimensiones y hasta  $2^{31}$  elementos por dimensión según la memoria disponible. Permite simplificar las conexiones cuando utilice un grupo de datos similares y cuando ejecute datos repetitivos. Los array son ideales para almacenar los datos generados en ciclos, donde cada interacción del ciclo produce un elemento del array. La creación de **arrays** de control o indicadores en el panel frontal según se requiera. El acomodo se hace mediante filas *n* y columnas *m*.

Utiliza un **Clusters** cuando se agrupan elementos de diferentes tipos de datos como un grupo de alambres trenzados, donde cada alambre en el cable representa un elemento diferente del cluster. Elimina el desorden en el cableado en el diagrama de bloques y reduce el número de terminales del panel conector que los subVIs necesiten, con un máximo de 28 terminales. Es una colección ordenada de uno o más elementos, pueden contener cualquier combinación de tipos de datos. Se acceden sus elementos todos a la vez y no uno a uno. Pero no pueden contener combinación de indicadores y controles.

**Bibliografía:** LabVIEW 8.20 Entorno gráfico de programación, José Rafael Lajara Vizcaíno y José Pelegrí Sebastián, Alfaomega-Marcocombo.

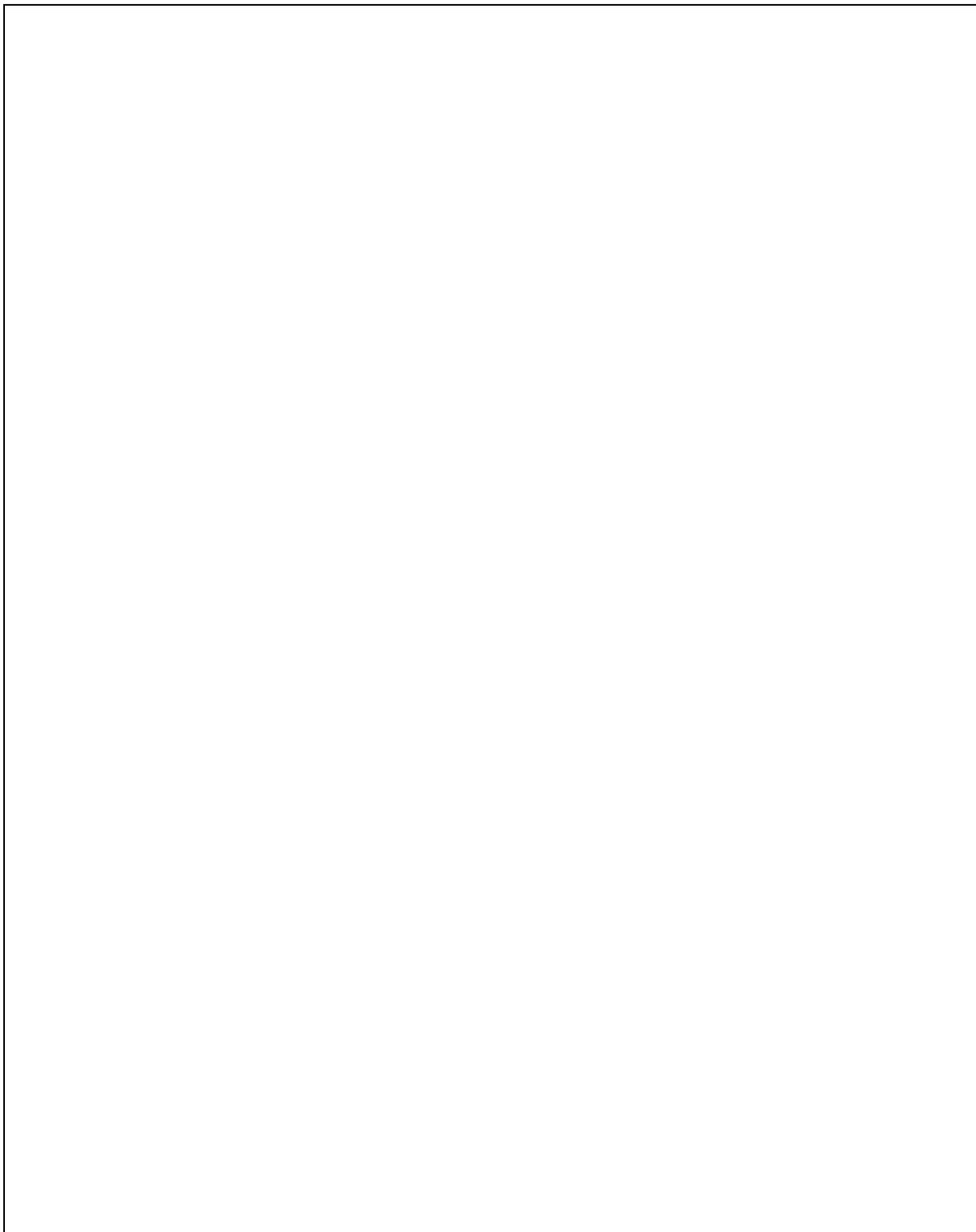
**Procedimiento:** el alumno programará las estructuras mediante array y cluster con sus entradas y salidas comprobando su aplicación.

**Material necesario:**

- Computadora
- Software LabVIEW
- Manual

**Diagrama de flujo e imágenes del programa:**





**Conclusión:**

**Práctica 7:** Programación de LabVIEW continuación

**Objetivo:** el alumno conocerá algún modelo, tipos y parámetros de las tarjetas DAQ.

**Introducción:** La generación y obtención de señales eléctricas es uno de los principales usos que se le da a LabVIEW, a través de tarjetas de adquisición de datos DAQ.

Las funciones comunes que suelen tener las DAQ son la *transferencia* en:

- Adquisición de señales analógicas
- Generación de señales analógicas
- Generación y adquisición de señales digitales
- Contadores y temporizadores (timers)
- Triggers (pre-trigger y post-trigger)
- Auto calibración, sensores, etc.

Los programas de aplicación como LabVIEW, envían los comandos de los controladores, para que adquiera y retorne una lectura. También presentan y analizan los datos adquiridos para su procesado virtualmente.

LabVIEW incluye un grupo de VI's que le permite configurar, adquirir y enviar los datos a los dispositivos DAQ (dispositivos de adquisición de datos) o NI-DAQ (dispositivos de medición e instrumentación). Cada dispositivo soporta diferentes plataformas de hardware y sistemas operativos específicos como las plataformas tipo VISA, GPIB, USB.

**Bibliografía:** LabVIEW 8.20 Entorno gráfico de programación, José Rafael Lajara Vizcaíno y José Pelegrí Sebastián, Alfaomega-Marcocombo.

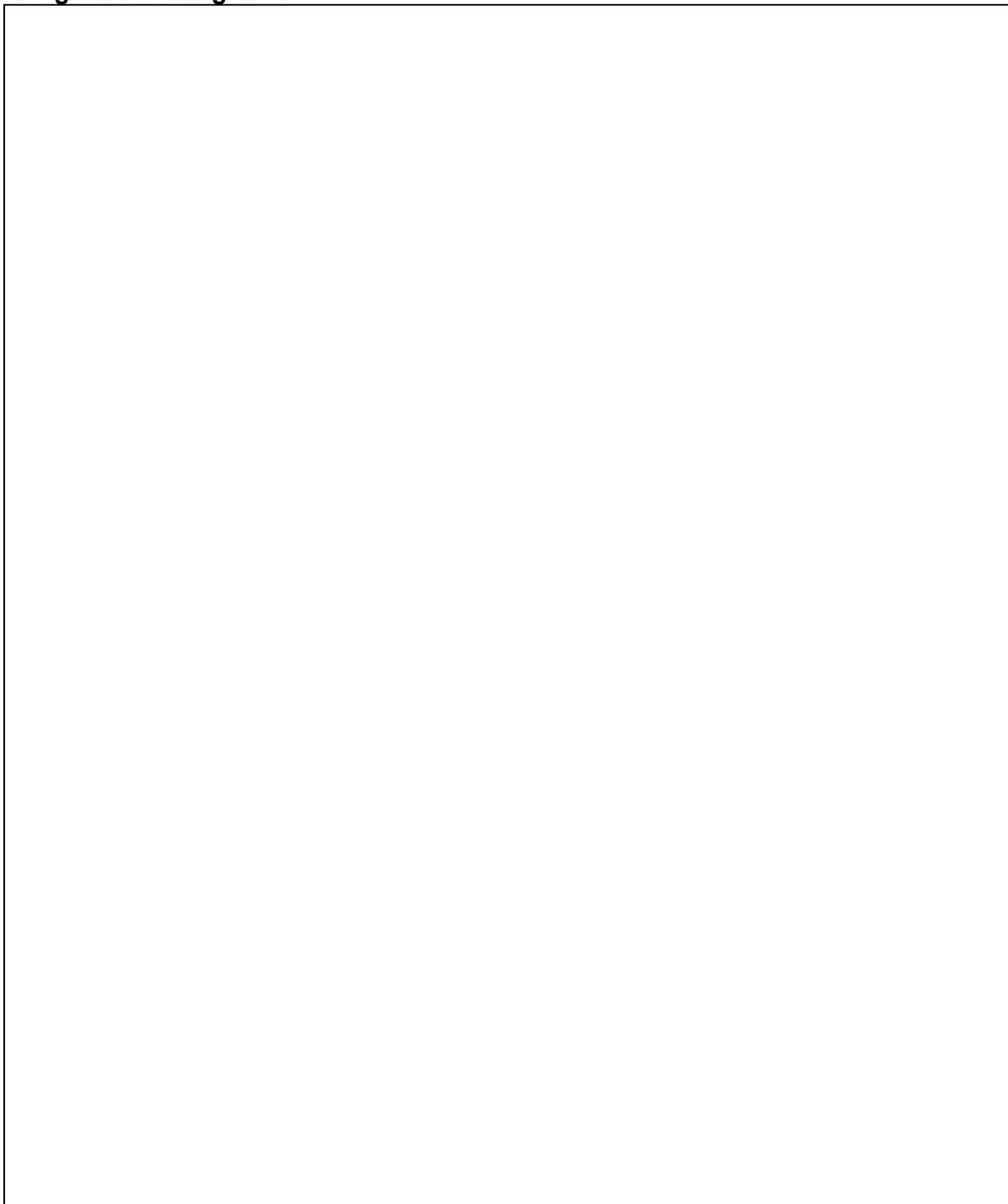
**Procedimiento:** el alumno programará y conectará por medio de la DAQ6009, algunas entradas/salidas analógicas y digitales que le permitan establecer una comunicación como si fueran sensores y actuadores con la interfaz a partir del dipsw y un potenciómetro como **entradas**, leds y como **salidas**, realizando las conexiones de acuerdo al diagrama e introducirá los valores lógicos comprobando su funcionamiento.

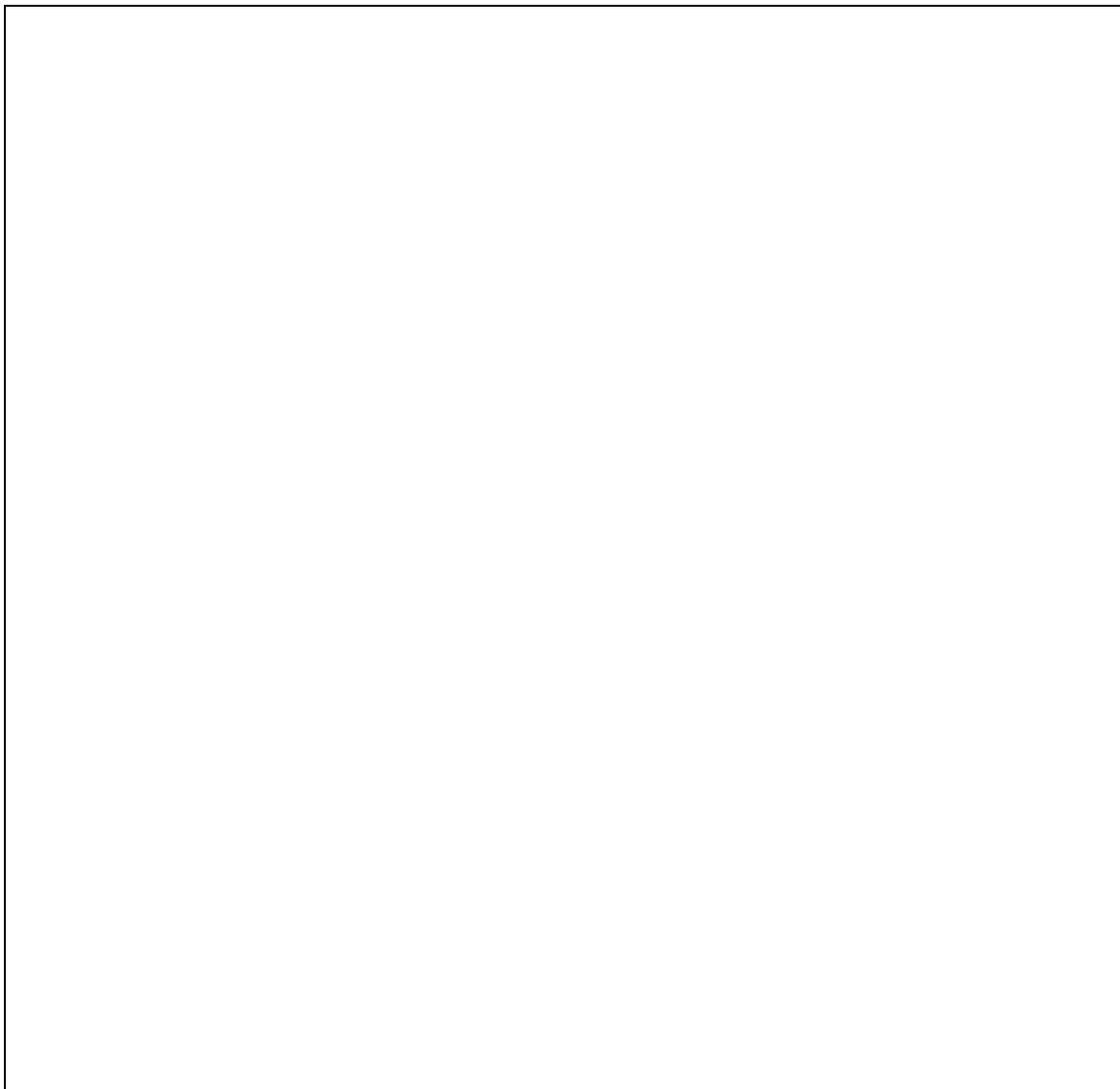
Las estructuras a través de **Measurement I/O / NI DAQmx/ DAQ assist** que permita comunicar y controlar variables de entrada y salida de tipo analógico y digital mediante la conexión de los pines correspondientes en concordancia a la declaración en el software, mediante **arrays** se introducen controles y LEDs tipo Booleano, la conexión de un LED y un potenciómetro de acuerdo al diagrama, comprobando su aplicación.

**Material necesario:**

- Protoboard
- Cables conectores (jumper)
- La DAQ (USB6009)
- Computadora (descargar el driver)
- Software LabVIEW
- Potenciómetro 10K $\Omega$
- Dipsw
- Leds (9) y resistencias de 330 ohm/ 250Mw (16)
- Manual (<http://www.ni.com/es-mx/shop.html>)

**Diagramas e imágenes:**



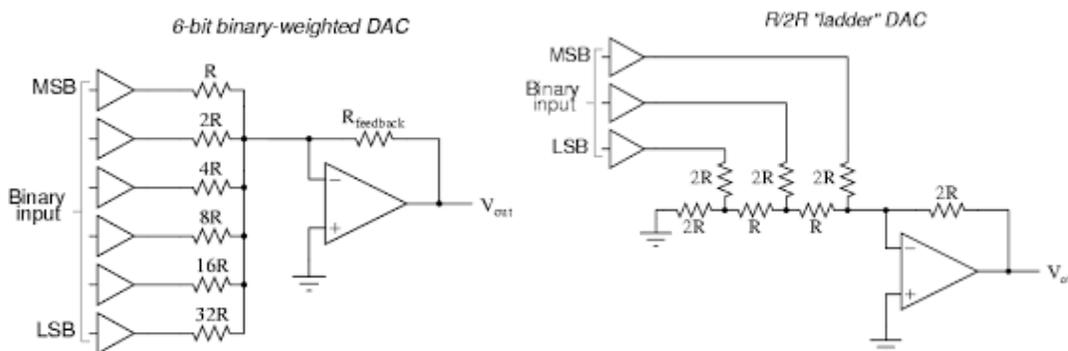


**Conclusión:**

**Práctica 8:** Introducción a los DAC

**Objetivo:** conexión y conocimiento de los parámetros del convertidor DAC 0800.

**Introducción:** un DAC es un circuito integrado electrónico que básicamente realiza un proceso de tomar un valor representado en el código digital (como binario o BCD) y convertirlo en un voltaje o corriente que sea proporcional al valor digital. Se utilizan siempre que un circuito integrado digital debe proporcionar como respuesta con un voltaje o corriente analógico para manejar un dispositivo, tales como el control, automatización, reconstrucción de señales.



La *precisión* en los DAC comúnmente se le llama error de escala completa (%FS) y error de linealidad, se presenta como porcentaje de la salida a FS del convertidor. El *error de desplazamiento* "offset": en el caso ideal, la salida de un DAC será de cero volts cuando la entrada binaria sea cero en todas. El *tiempo de establecimiento* es la velocidad de operación de un DAC por lo general se especifica como el tiempo requerido para que la salida del DAC cambie de cero a su valor a escala completa (cuando todos los bits cambian de 0 a 1). En la práctica se mide como el tiempo necesario para que la salida del DAC se establezca dentro de  $\pm 1/2$  el tamaño de paso de su valor final. Los valores más comunes se encuentran entre 50 ns a 10 us.

**Bibliografía:** Sistemas Digitales Principios y aplicaciones, Ronald J. Tocci, Prentice Hall

**Procedimiento:** 1º el alumno conectará el CI DAC 0800 con la pulsera antiestática con sus entradas a partir del dipsw para introducir ceros y unos lógicos, de acuerdo al diagrama.

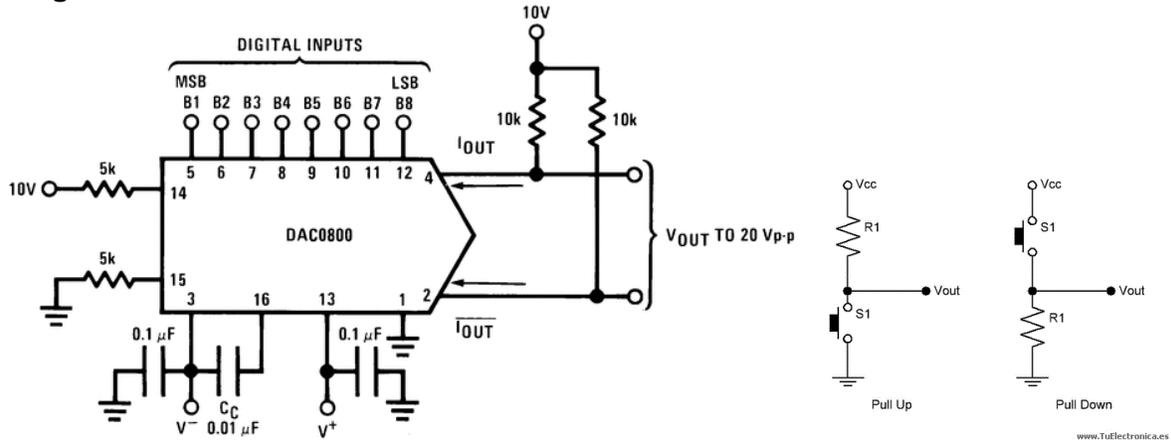
2º Medirá el voltaje con el voltímetro a la salida de modo diferencial (conectando los pines 2 a 4) y por separado a cada una (conectando los pines 4 a tierra y de 2 a tierra), e introducirá los valores lógicos del MSB al LSB, acuerdo a su configuración anotando los valores medidos en la tabla siguiente y comprobando su veracidad.

**Material necesario:**

- Protoboard
- Cables conectores (jumper)
- Pulsera anti estática
- El CI DAC 0800 (CMOS)
- Interruptores (Dipsw 8 resistencias (R1) 330Ω TTL o de 1kΩ CMOS)
- Resistencias 10kΩ (4)

- Condensadores de  $0.1\mu\text{F}$  (2) y  $0.01\mu\text{F}$
- Voltímetro
- Fuente de alimentación simétrica ( $\pm 10\text{Volts}$ )
- Diagrama de identificación de los CI
- Manual

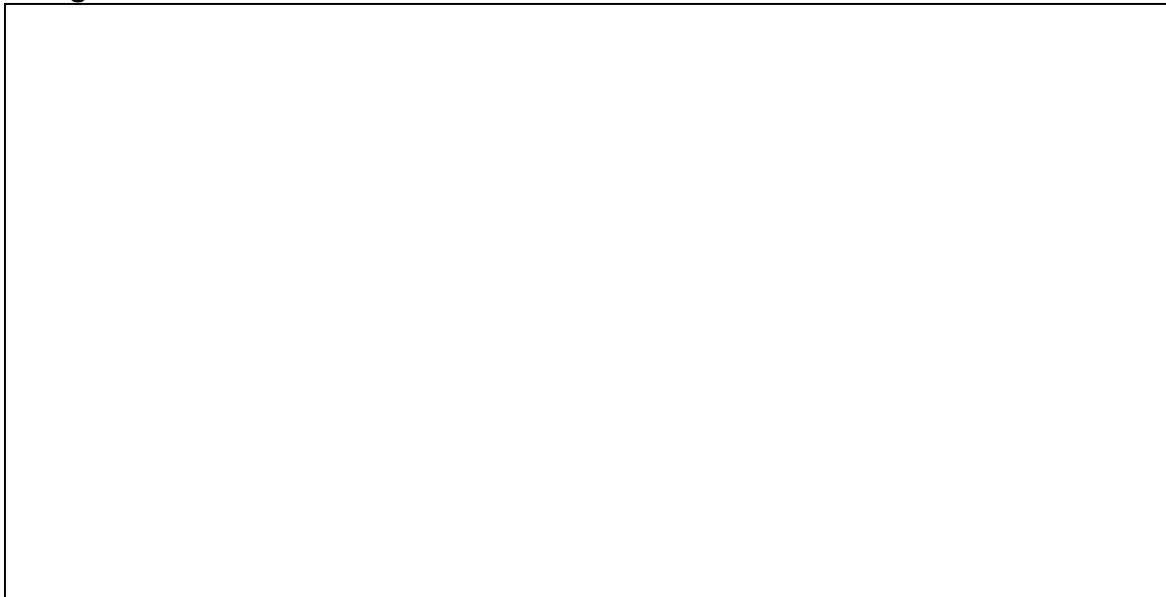
**Diagrama:**

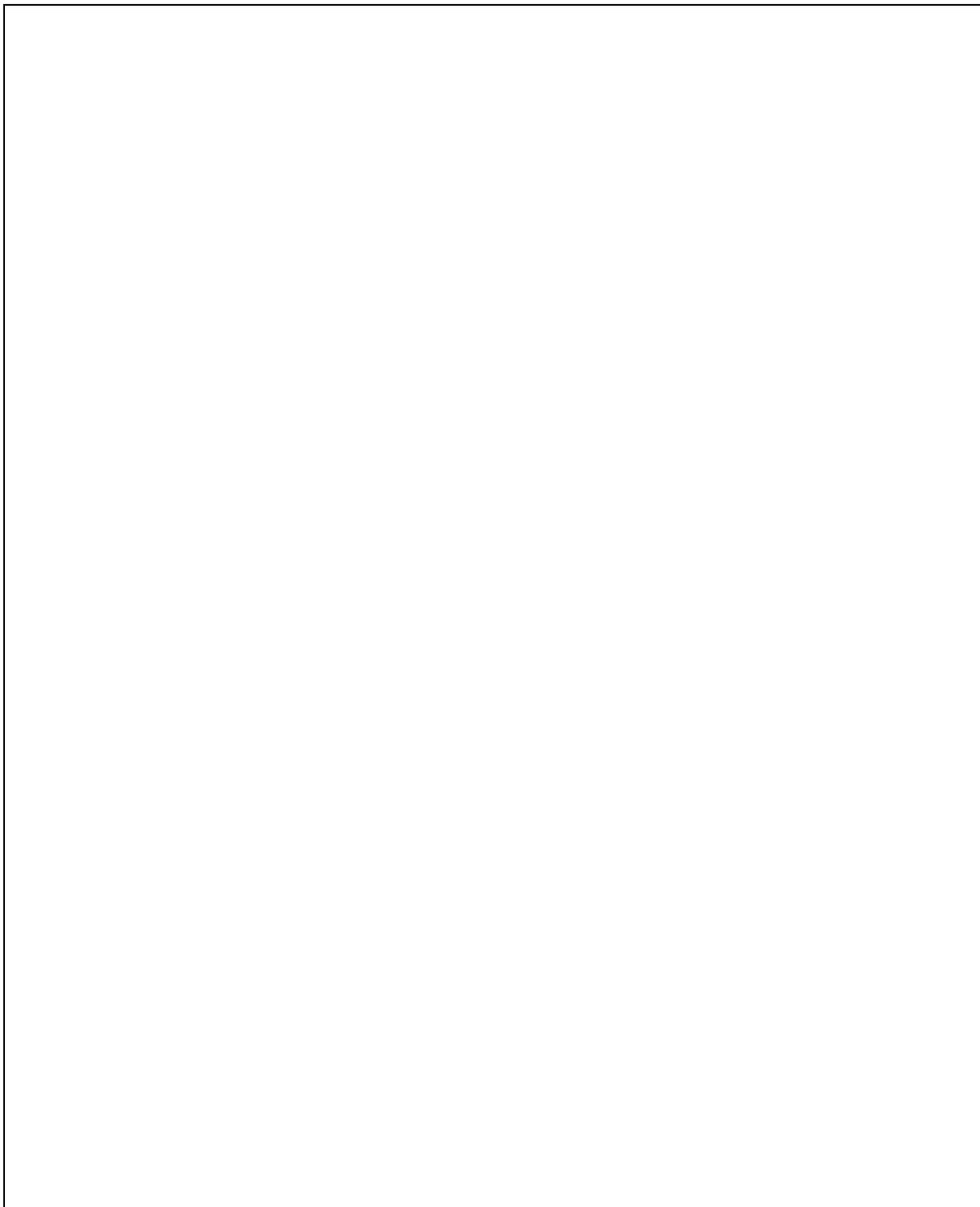


**Mediciones:** registre los valores obtenidos conectando un multímetro entre los terminales del circuito DAC

Pines	MSB B1	B2	B3	B4	B5	B6	B7	LSB B8
De 2 a 4								
De 2 a GND								
De 4 a GND								

**Imágenes:**





**Conclusión:**

**Práctica 9:** Introducción a los ADC 0804

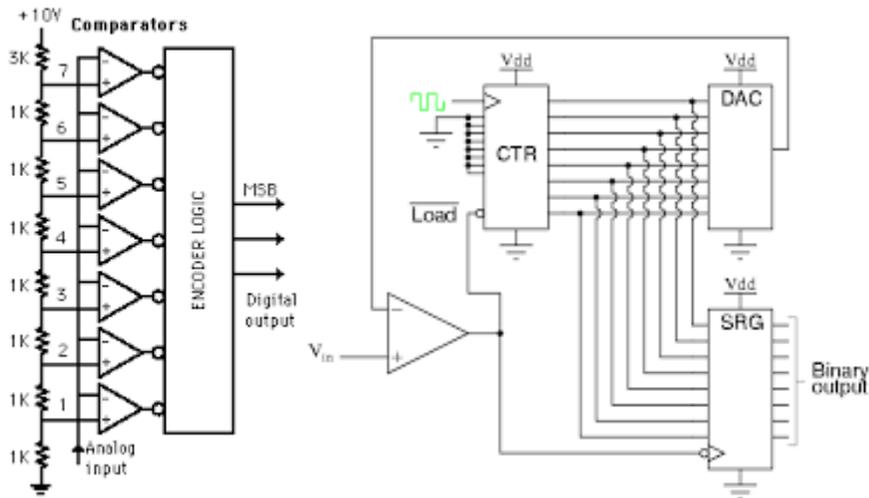
**Objetivo:** conexión y conocimiento de los parámetros del convertidor ADC 0804.

**Introducción:** un conversor o convertidor de señal analógica a digital CAD; *Analog-to-Digital Converter*, es un dispositivo electrónico capaz de convertir una señal analógica, ya sea de tensión o corriente y *después de cierto tiempo*, obtiene una señal digital mediante un cuantificador y codificándose en muchos casos en un código binario en particular, que representa la entrada analógica de manera proporcional a la entrada. Donde un código es la representación unívoca de los elementos, en este caso, cada valor numérico binario hace corresponder a un solo valor de tensión o corriente.

Utiliza el *muestreo* como el proceso de tomar muestras de la señal a intervalos periódicos y es una modulación por amplitud. La *cuantificación* y *codificación* en general el proceso de cuantificación y codificación es realizado en el mismo paso salvo que se necesite realizar una codificación específica.

En la cuantificación de la señal se produce pérdida de la información que no puede ser recuperada en el proceso inverso, es decir, en la conversión de señal digital a analógica y esto es debido a que se truncan los valores entre 2 niveles de cuantificación, mientras mayor cantidad de bits mayor resolución y por lo tanto menor información pérdida.

Se utiliza en equipos electrónicos como computadoras, grabadores de sonido y de vídeo, y equipos de telecomunicaciones



El *tiempo de conversión* ( $t_c$ ) es el intervalo entre el *fin* y el *inicio* con la activación de la salida como Fin De conversión *FDC*, que depende de *V<sub>A</sub>* voltaje analógico ocasiona un valor mayor requerirá más escalones. El tiempo de conversión máximo ocurrirá cuando *V<sub>A</sub>* esté apenas por debajo del límite de escala, de modo que *V<sub>A</sub>* deba pasar al último

escalón para activar FDC. Para un convertidor de  $N$  bits esto es:  $t_c (máx.) = (2^{N-1})$  ciclos de reloj

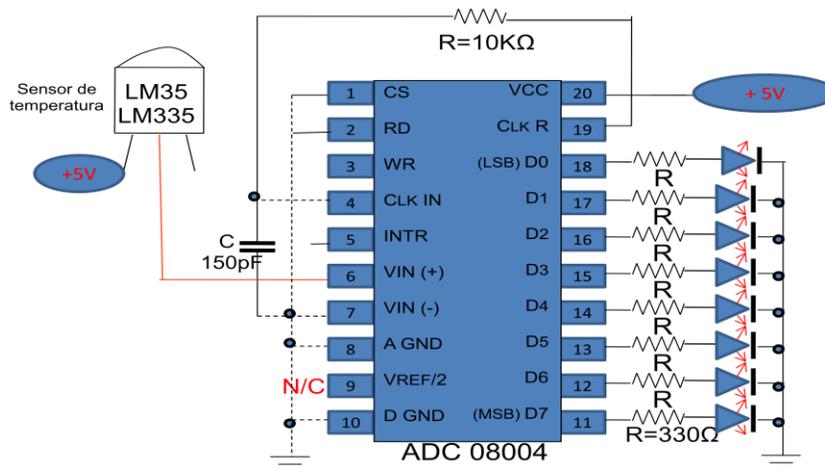
**Bibliografía:** Sistemas Digitales Principios y aplicaciones, Ronald J. Tocci, Prentice Hall

**Procedimiento:** el alumno realizará la conexión del convertidor analógico - digital (ADC) de acuerdo a la identificación de los terminales del diagrama comprobando su funcionamiento. Para la calibración se ajusta el potenciómetro a un voltaje justo a la mitad del valor máximo que entrega el sensor, con la finalidad de obtener el código a escala completa a la salida. El alumno conectará el CI con sus entradas a partir del sensor LM35 y a la salida agregando las conexiones a partir de Led comprobando su funcionamiento.

**Material necesario:**

- Fuente de alimentación (5 Volts)
- Protoboard
- Cables conectores (jumper)
- Pulsera antiestática
- Sensor de temperatura LM35 (o foto resistencia)
- Potenciómetro 10kΩ
- ADC 0804
- Resistencia 10kΩ
- Condensador 150 pF
- Leds y resistencias de 330Ω (8)
- Voltímetro
- Datasheet o manual

**Diagramas:**



**Imágenes:**

**Conclusión:**

**Práctica 10:** Proyecto final

**Objetivo:** identificación, conexión y comprobación de lo aprendido mediante un proyecto.

**Introducción:** la simulación virtual de un proceso de producción en base a un producto real, el cual contendrá la fórmula con ingredientes, procesado (dosificación de material prima), dosificación para su envasado y empaquetado. De al menos tres sabores, presentaciones, preparaciones, variedades. Que incluya la simulación en base a un pedido y muestre controles e indicadores dicho proceso.



**Bibliografía:** LabVIEW 8.20 Entorno gráfico de programación, José Rafael Lajara Vizcaíno y José Pelegrí Sebastián, Alfaomega-Marcombo.

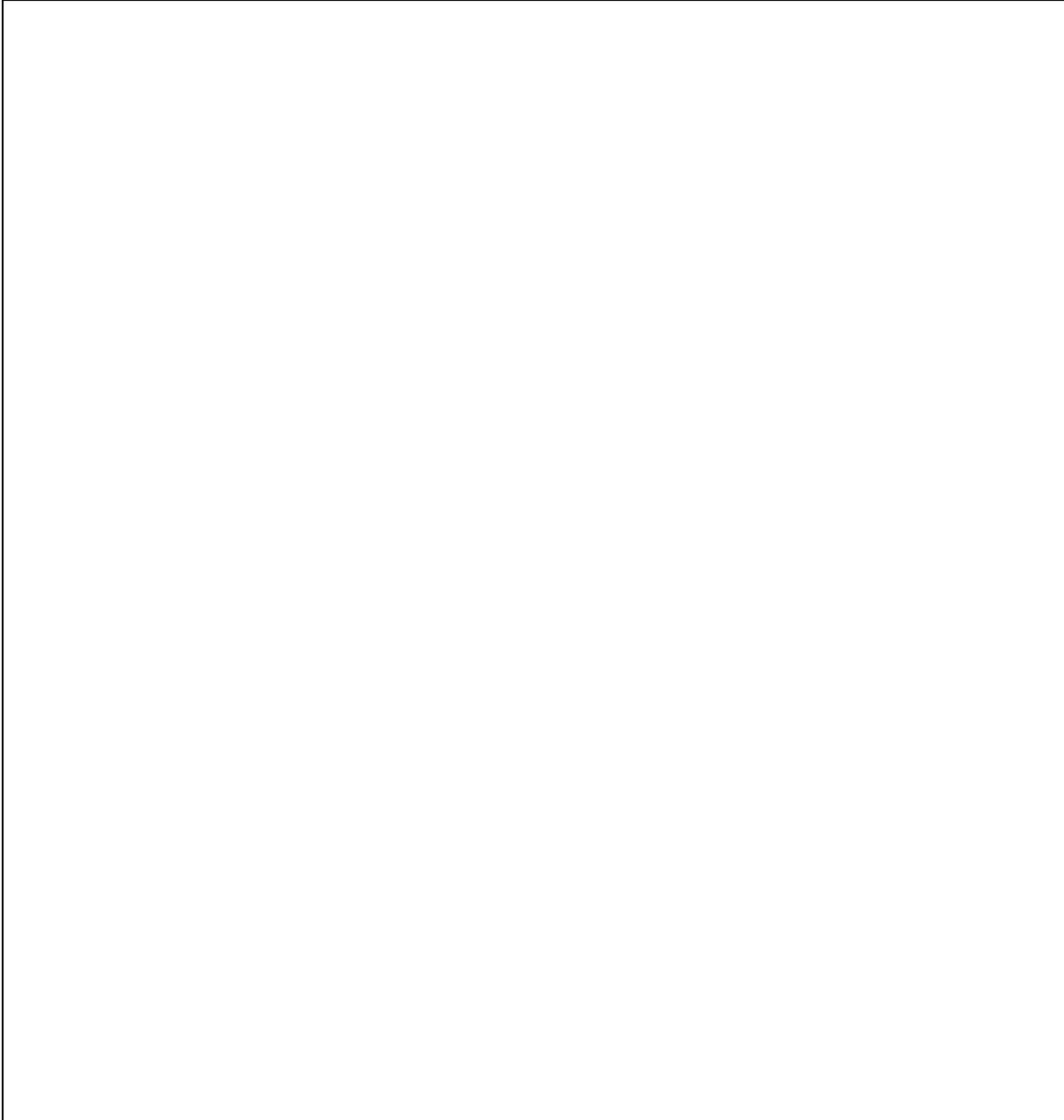
**Procedimiento:** el alumno programará un diseño a través del cual simule el control de un proceso de fabricación con entradas de materia prima, el envasado y empaquetado de varias medidas.

**Material necesario:**

- Computadora

- Software LabVIEW
- Manual

**Diagramas e imágenes:**



**Conclusión:**