Chapter 1 Homework Set: Problems 5, 6, 8 and 9

The VI you have written is associated with the first chapter of this book. Thus, within the *YourName* folder, create a subfolder named *Chapter 1* and save the VI there as follows: With *YourName* open, click on the *Create New Folder* button. In the highlighted box that appears next to the folder icon, name the new folder *Chapter 1*, and click *Open* to open this folder. Finally in the *File name*: box, type *Sine Wave Chart (While Loop)*, and then press *OK*. Your VI will be saved in the *Chapter 1* folder, which is within the *YourName* folder (a location we will denote as *YourName|Chapter 1*) with the extension *.vi* appended to its name.

### DO IT YOURSELF

Write a VI called *Stopwatch* that functions as a stopwatch with a precision of 0.01 second. Design *Stopwatch* so that it continuously displays the elapsed time from when its *Run* button is pressed until its *Stop Button* is pressed. The front panel of *Stopwatch* should appear as shown below. Here, the elapsed time is displayed on a *Numeric Indicator*
(found in Controls>>Modern>>Numeric) labeled Elapsed Time (second). Choose the Stop Button's Mechanical Action appropriately.

Problems

1. Write a VI called Greater Than Ten that lights a Round LED indicator (found in Controls>>Modern>>Boolean) on its front panel when the iteration terminal of a While Loop on its block diagram has a value of greater than 10. You may find the following icons useful: Select and Greater? in Functions>>Programming>>Comparison.

2. Write a VI called Single Sine Cycle that stops automatically after the While Loop on its block diagram has iterated exactly 1000 times. As the While Loop performs these 1000 iterations, make a Waveform Chart plot one cycle of a sine wave (where the final data point is at the equivalent point on the sine wave as the initial point). You may find the following icons useful: Equal? in Functions>>Programming>>Comparison and Pi Multiplied By 2 in Functions>>Programming>>Numeric>>Math & Science Constants.

3. On the block diagram of Sine Wave Chart (While Loop), pop up on the While Loop's conditional terminal and select its Continue if True mode. Then, modify the program as needed so that it executes in the original manner (i.e., charts the sine wave until the user clicks the Stop Button).

4. Write a VI called Metronome, whose front panel includes a Numeric Control (found in Controls>>Modern>>Numeric) labeled Beats Per Minute, as shown below.
When run, construct *Metronome* so that it produces $N$ beeps every minute until the *Stop Button* is clicked, where $N$ is the value entered in the *Beats Per Minute* control and a beep is a 264-Hz sound wave of 100-ms duration. To create a beep, use *Beep.vi*, which is found in **Functions>>Programming>>Graphics and Sound**, with its use system alert? input set to FALSE.

5. The odd integers are given by $2i + 1$, where $i = 0, 1, 2, \ldots$ Write a program called *Odd Integer Search* that answers the following question: What is the smallest odd integer that is divisible by 3 and also, when cubed, yields a value greater than 4000? Display the answer to this question in a front-panel *Numeric Indicator*. You may find icons in the following palettes useful: **Functions>>Programming>>Numeric** (e.g., *Quotient & Remainder*), **Functions>>Programming>>Comparison, Functions>>Programming>>Boolean**, and **Functions>>Mathematics>>Elementary & Special Functions>>Exponential Functions**.

6. Construct a VI named *EvenOdd* whose front panel has two *Round LED* indicators and a *Stop Button* (all found in **Controls>>Modern>>Boolean**). Label one indicator *Even* and the other *Odd*, as shown below. Place a While Loop on the block diagram, which iterates once every 0.5 seconds, and construct a subdiagram within it so that the *Even* indicator is lit (and *Odd* is unlit) during iterations for which the value of the iteration terminal $1$ is even, while the *Odd* indicator is lit (and *Even* is unlit) during iterations for which the value of $1$ is odd. You may find the following icons useful: *Select and Equal To 0?* in **Functions>>Programming>>Comparison** as well as *Quotient & Remainder* in **Functions>>Programming>>Numeric**.

7. Explore the impact of While Loop execution on CPU usage. Open the *Windows Task Manager* by right-clicking an empty area on the taskbar at the bottom of your monitor and then selecting *Task Manager*, or by pressing $<\text{Ctrl} + \text{Shift} + \text{Esc}>$. You will find that the *Task Manager* displays CPU usage (expressed as a percentage of your computer's maximum value) at the bottom of its window.

(a) Open *Sine Wave Chart (While Loop)*, and run this VI with *Wait (ms)* programmed to produce one While Loop iteration every 100 ms. What is the approximate increase in CPU usage that results when the program is running?
(b) Next, program **Wait (ms)** to produce one While Loop iteration every 1 ms, then run the VI. What is the approximate increase in CPU usage that results when the program is now running?

(c) Finally, delete **Wait (ms)** from the VI's block diagram so that the While Loop will iterate as fast as possible on your computing system (e.g., millions of iterations per second), then run the program. What is the approximate increase in CPU usage that results when the program is now run? (Note that such "memory hogging" should be avoided unless your program is executing a high-speed task, that is, not simply updating a plot, as in this case.)

8. Write a program called **Iterations Until Integer Equals Five**, which randomly generates an integer in the range from 1 to 10, and iterates this random process until the integer equals 5. The number of iterations required until the randomly generated integer equals 5 is then displayed on the front panel in a **Numeric Indicator** (found in **Functions>>Programming>>Numeric**), which is labeled **Required Iterations**, as shown below.

![Iterations Until Integer Equals Five](image)

To create a random integer in the range from 1 to 10, use **Random Number** (0−1), to generate a random floating-point number in the range from 0 up to (but not including) 1, multiply this number by 10, and then use **Round Toward + Infinity** to round the floating-point number to the next highest integer (note **Round Toward + Infinity** rounds x.000 to x). All of these icons are found in **Functions>>Programming>>Numeric**. Think carefully about how to determine the value of **Required Iterations**. Also, you will need to use the **Equal?** icon, which is found in **Functions>>Programming>>Comparison**.

Knowing that the integers are created randomly (i.e., equal probability for producing integer 1 through 10 each iteration), what do you expect the value of **Required Iterations** to be on average? Run **Iterations Until Integer Equals Five** 20 times and record the value of **Required Iterations** resulting from each run. Is the average of these 20 values close to the value that you expected?

9. A characteristic of the Waveform Chart can be controlled within a program using a **Property Node**.

   (a) In this chapter, you set a Waveform Chart's **X-Axis Multiplier** equal to 0.2 using the **Properties** dialog window. Alternately, this property of the Chart can be set on the block diagram via a Property Node as follows: With **Sine Wave Chart** (While
Loop) open, pop up on the Chart's icon terminal and select Create>>Property Node>>X Scale>>Offset and Multiplier>>Multiplier, then place the resulting Property Node on the block diagram. This Property Node is created as an indicator, hence the small outward-directed black arrow at its right side. Change this icon to a control by popping up on its lower section and selecting Change To Write. The black arrow will now be inward-directed on its left side. Using Create>>Constant, wire a value of 0.2 to the XScale Multiplier property control as shown below.

Waveform Chart

0.2

XScale.Multiplier

Run Sine Wave Chart (While Loop) and assure yourself that the Property Node is indeed setting the X-Axis Multiplier to 0.2.
(b) Program Sine Wave Chart (While Loop) so that its Waveform Chart is cleared at the start of each run. The appropriate Property Node is made by selecting Create>>Property Node>>History Data. After changing this Property Node to a control, Create>>Constant will produce the needed input called an Empty Array as shown below.

Waveform Chart

In order to clear the Chart at the start of each program execution (i.e., immediately after the Run button is pressed), should this Property Node be placed inside or outside of the While Loop? Run Sine Wave Chart (While Loop) several successive times and demonstrate that the Property Node performs as intended.
(c) Change the background color Sine Wave Chart (While Loop)'s Waveform Chart during runtime as follows: Place a Property Node within the While Loop using Create>>Property Node>>Plot Area>>Colors>>BG Color, then change this Property Node to a control. On the front panel, place a Framed Color Box, found in Controls>>Modern>>Numeric, then on the block diagram wire its terminal to the Property Node. Run Sine Wave Chart (While Loop) and demonstrate that the Chart's background color can be controlled using the Operating Tool.