

LabVIEW Lesson 3 – Structures


Lesson Overview

- What structures are available in LabVIEW.
 - How to use a while loop.
 - How to use a for loop.
 - How to use data/values from previous loops.
-

I. Structures available in LabVIEW.

- a. A structure is a graphical representation of a loop or case structure used in text-based programming and is used in the block diagram to repeat blocks of code and provide conditional execution requirements.
- b. LabVIEW offers seven different types of structures including both while and for loops as well as case structures.
- c. These structures can be found under the **Functions Palette** → **All Functions** → **Structures** or **Functions Palette** → **Exec Ctrl** (*This one has limited options.*)

II. Using the **While Loop**.

- a. The while loop can be used to repeat a block of code until a certain condition is met.
 - i. Two different types of terminating conditions can be set:
 1.  **Stop if True (default setting)** – When using this condition, the while loop will run until a certain condition is met. For example, consider the following:

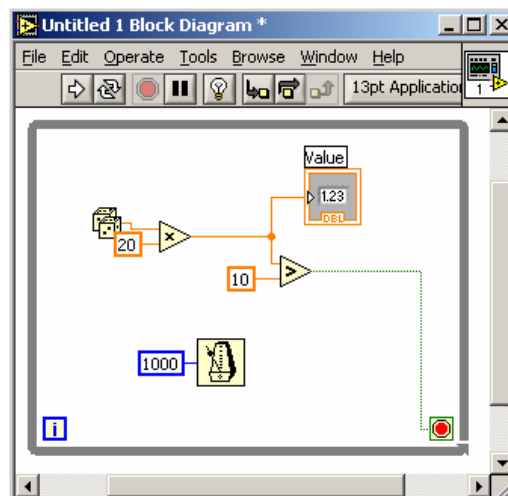








Figure 3.1: Stop if True Example




For this loop, it will continue to run until the output value is greater than 10 and then, it will terminate.

2.  **Continue if True** – When using this condition, the while loop will run until a certain condition is no longer satisfied or true. For example, if we change the terminating function to Continue if True (**Right-Click** on the stop sign terminal and select **Continue If True**) in the previous example, the loop will run until the output value is less than 10 and then, it will terminate.

- ii.  **Iteration Terminal** – The iteration terminal will keep record of how many iterations have been completed. The iteration count always begins from 0, so during the first iteration, the output from the iteration terminal will be 0.

b. **Example 3.1: Matching Dice using a While Loop**

- i. This VI will simulate the rolling of two dice and determining how many rolls it takes to roll matching dice.
- ii. Starting with the **Block Diagram**.
 1. Create a **While Loop**
 - **Functions Palette** → **Exec Ctrl** → **While Loop** → **Left-Click** and **Hold** on block diagram → **Drag** to create a fairly large box, which will represent your while loop (*be sure to make it large enough to fit several terminals inside*).
 2. Delete the **Stop Button Terminal** that is connected to the **Stop If True Terminal** (*automatically created when creating a while loop*).
 3. Insert the following inside the while loop:
 - a. **2 Random Number Generators** 
 - **Functions Palette** → **All Functions** → **Arith/Compare** → **Numeric** → **Random Num**
 - b. **2 Multiplication Functions** 
 - **Functions Palette** → **Arith/Compare** → **Numeric** → **Multiply**
 - c. **2 Round To +Inf Functions** 
 - **Functions Palette** → **Arith/Compare** → **Numeric** → **Round To +Inf**
 - d. An **Equals Function** 
 - **Functions Palette** → **Arith/Compare** → **Compare** → **Equal To**

- e. A **Wait Until Next MS Multiple Function** 
 - **Functions Palette** → **All-Functions** → **Time & Dialog**  → **Wait Until Next MS Multiple**
 - f. An **Increment Function** 
 - **Functions Palette** → **Arith/Compare** → **Numeric** → **Increment**.
4. Wire and arrange the terminals as shown in Figure 3.2, excluding the numerical indicators (Die 1, Die 2, # of Rolls). You will have to create a few constants as well, which can be done by **Right-Clicking** on the appropriate terminal and selecting **Create** → **Constant**.

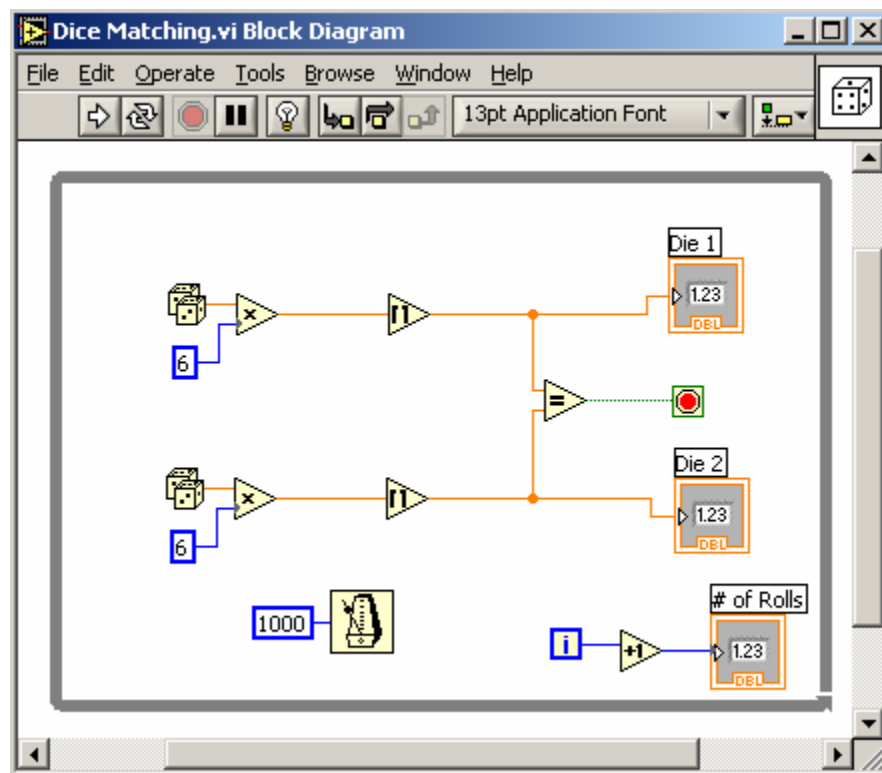


Figure 3.2: Die Matching Block Diagram

- iii. On the **Front Panel**, we need 3 **Numerical Indicators** (*two will show the number on each die and the other will indicate the number of rolls taken*). Label each output as follows: “**Die 1**”, “**Die 2**”, and “**# of Rolls**” as shown in Figure 3.3.

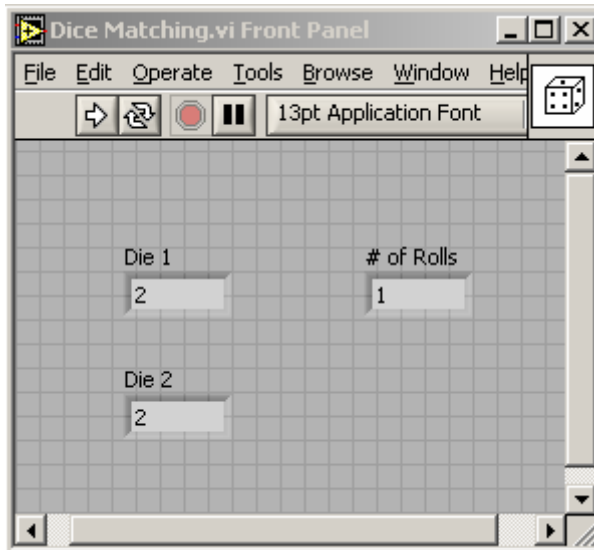












Figure 3.3: Die Matching Front Panel

- iv. Switch back to the **Block Diagram** and drag the **Numerical Indicator Terminals** into the **While Loop** and wire according to the layout shown in Figure 3.2.
- v. **Save** your VI as **DieMatch.VI** and then, **Run** a couple of times to see how it works.


III. Using the **For Loop**.

- a. The for loop can be used to repeat a block of code for a given amount of repetitions.
- b. The for loop does not use the same termination terminals; instead, it uses the **Number of Iterations Terminal** [N](#).
 - i. The number of iterations can be set permanently inside the block diagram by attaching a constant to it, or it can be set by the user through the use of a numerical control on the front panel.
 - ii. The for loop will run the pre-described number of iterations and then terminate.
- c. **Example 3.2: Timed Thermometer VI.**
 - i. **PROBLEM:** You need to design a VI that is capable of recording temperature measurements in both degrees Celsius and degrees Fahrenheit for a given time period; as well as, be being able to control the time interval between temperature measurements.
 - ii. This VI you may find very practical as it can be used to acquire temperature measurements over a designated time period. This VI also allows you to control how closely you want the measurements recorded.

iii. Starting with the **Block Diagram**.

1. Create a **For Loop** 
 - **Functions Palette** → **All-Functions** → **Structures** → **For Loop** → **Left-Click** and **Hold** on block diagram → **Drag** to create a fairly large box, which will represent your for loop (*be sure to make it large enough to fit several terminals inside*).
2. Insert **TempConv.VI** from previously created VI's
 - **Functions Palette** → **All-Functions** → **Select VI** (bottom left corner)  → **TempConv.VI**
3. Insert a **Random Number Generator** 
 - **Functions Palette** → **Arith/Compare** → **Numeric** → **Random Num**
4. Insert a **Uniform White Noise Waveform** 
 - **Functions Palette** → **All-Functions** → **Analyze**  → **Waveform Generation**  → **Uniform White Noise**
5. Insert a **Wait Until Next MS Multiple Function** 
 - **Functions Palette** → **All-Functions** → **Time & Dialog**  → **Wait Until Next MS Multiple**
6. Insert 2 **Increment Functions** 
 - **Functions Palette** → **Arith/Compare** → **Numeric** → **Increment**
7. Insert a **Multiplication Function** 
 - **Functions Palette** → **Arith/Compare** → **Numeric** → **Multiply**.

iv. Switch to the **Front Panel**

1. Insert 2 **Numerical Controls** and name them “**Input Desired Duration of Measurements in Seconds:**” and “**Input Desired Time Measurement Interval in Seconds:**”
2. Insert 2 **Numerical Indicators** and name them “**Current Elapsed Time (sec)**” and “**Current Temp (Deg F)**”
3. Insert a **Vertical Toggle Switch**
 - **Controls Palette** → **Buttons** → **Vertical Toggle Switch** 
 - Insert the following text: **Left of the Switch:** “**Indicate Temperature Scale of Measuring Device:**”, **Above the**

Switch: “Input Temp Scale”, and Right of the Switch: “Deg F” and “Deg C” as shown in Figure 3.4

4. Insert a **Waveform Chart**
 - **Controls Palette** → **Graphical Indicators** → **Waveform Chart**
 - **Edit** the axes by **Double-Clicking** on each **Axis Title** and for the **X-Axis**, type “**Time (sec)**” and for the **Y-Axis**, type “**Temp (Deg F)**”
5. Arrange the **Front Panel** as shown in Figure 3.4.
- v. Switch back to the **Block Diagram**
 1. Arrange and wire the **Block Diagram** as shown in Figure 3.5.
(NOTE: You will also have to add in some constants.)

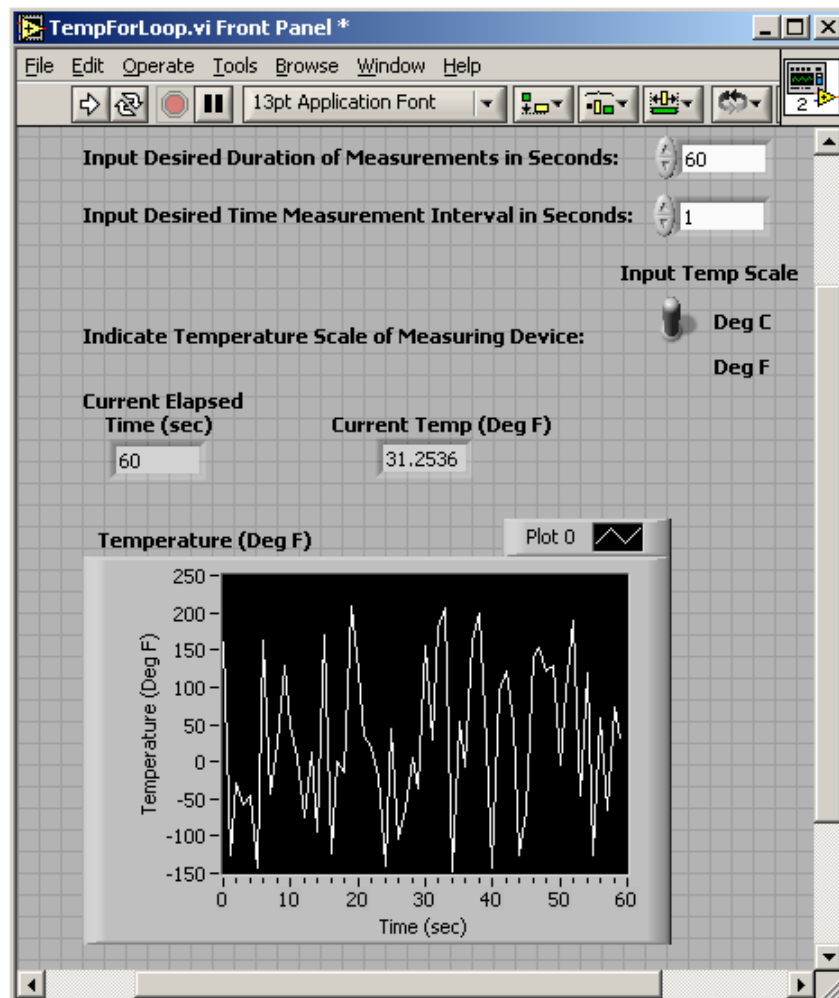


Figure 3.4: Temperature Measurement Front Panel

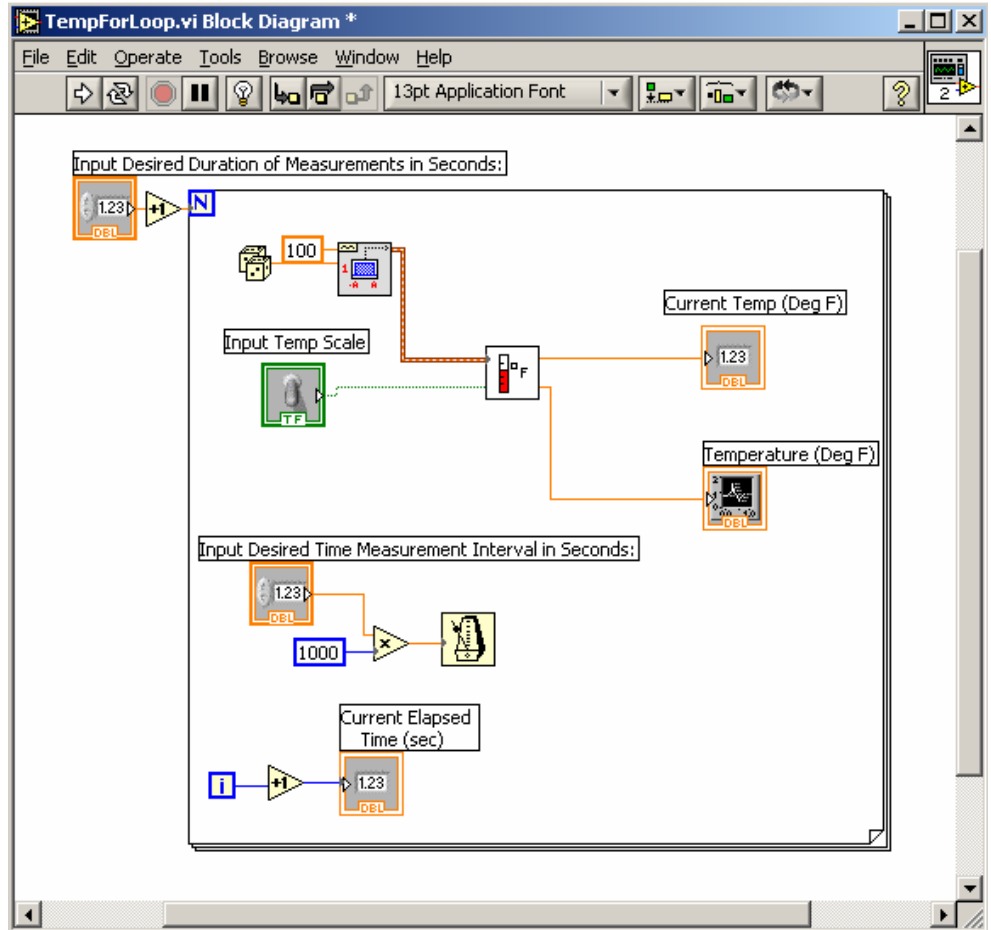

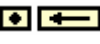


Figure 3.5: Temperature Measurement Block Diagram

2. **Save** your VI as **TempForLoop.VI**.
3. **Test** your VI using a **1-Second Interval**, **60-Second Duration**, and an **Input Temperature Scale in Deg C**.

IV. Accessing Data from Previous Loops/Cycles


- a. Often times when running data through a loop, data from the previous iteration is required. In order to access data from a previous iteration, shift registers or feedback nodes are used.
 - i. **Shift Registers**  – These are similar to static variables in text-based programming languages and are used to transfer data from one iteration to the next. Shift registers can be “stacked” on top of one another if data from multiple iterations need to be used.
 - ii. **Feedback Nodes**  - These are like shift registers as they store data when the iteration completes and then sends it to the next iteration. These nodes can transfer any kind of data type and are most often used to avoid using long unnecessary wires inside loops. The direction of the arrow indicates the flow direction of the data.

b. Example 3.3: Temperature Averaging with Shift Registers

- i. **PROBLEM:** The previously created temperature measurement VI does not include averaging. The user wants to keep a running average of the temperatures as well as a running average of just the current and previous temperature in addition to the actual readings. All of these should be plotted in a graph.
- ii. This VI uses shift registers to sum up different temperatures and calculate an average temperature.

iii. **Open** the last VI created **“TempForLoop.VI”** and switch to the **Block Diagram**.

1. Creating the Two-Point Average

- a. Add a **Set of Shift Registers**  plus an additional **Shift Register** on the **Left** (*These will be used to reference temperature values from the previous iteration as well as the one prior to that.*)
 - **Right-Click** on border of **For Loop** → **Add Shift Register**
 - To add additional element, **Right-Click** on the **Left Register** → **Add Element**

b. Insert a **Division Function** 

- **Functions Palette** → **Arith/Compare** → **Numeric** → **Multiply**

c. Insert an **Addition Function** 

- **Functions Palette** → **Arith/Compare** → **Numeric** → **Add**




d. Copy both the **“Current Temp” Numerical Indicator** and the **“Temperature” Chart Terminals**

- **Highlight** both **Terminals** → **Hold Ctrl** → **Left-Click** and **Drag Twice** to desired locations (Refer to Figure 3.7)

e. Wire the **Block Diagram** as follows (if needed, refer to Figure 3.7):

- **2 Shift Registers** on the **Left** to the **Addition Function**
- **Addition Function Output** and **Constant of 2** to the **Division Function**
- **Division Function Output** to the **Waveform Chart** and **Numerical Indicator**
- **Temperature Output** to the **Right Shift Register** (*These temperatures will be the inputs to the left shift registers.*)

2. Creating the Running Overall Average

- a. Add a **Set of Shift Registers** 
 - **Right-Click** on border of **For Loop** → **Add Shift Register**
 - b. Insert a **Division Function** 
 - **Functions Palette** → **Arith/Compare** → **Numeric** → **Multiply**
 - c. Insert an **Addition Function** 
 - **Functions Palette** → **Arith/Compare** → **Numeric** → **Add**
 - d. Copy both the “**Current Temp**” **Numerical Indicator** and the “**Temperature**” **Chart Terminals**
 - **Highlight** both **Terminals** → **Hold Ctrl** → **Left-Click** and **Drag Twice** to desired locations (Refer to Figure 3.7)
 - e. Wire the **Block Diagram** as follows (if needed, refer to Figure 3.7):
 - **Shift Register** on the **Left** and the **Current Temperature Output** to the **Addition Function**
 - **Addition Function Output** and **# of Iterations** to the **Division Function**
 - **Division Function Output** to the **Waveform Chart**, **Numerical Indicator**, and **Right Shift Register**
- iv. Switch to the **Front Panel**
1. **Label** the **Waveform Charts** and **Numerical Indicators** appropriately and organize your **Front Panel** so that it is easy to follow (Refer to Figure 3.6, if needed)
 2. **Save** as “**TempForLoopShift.VI**”.

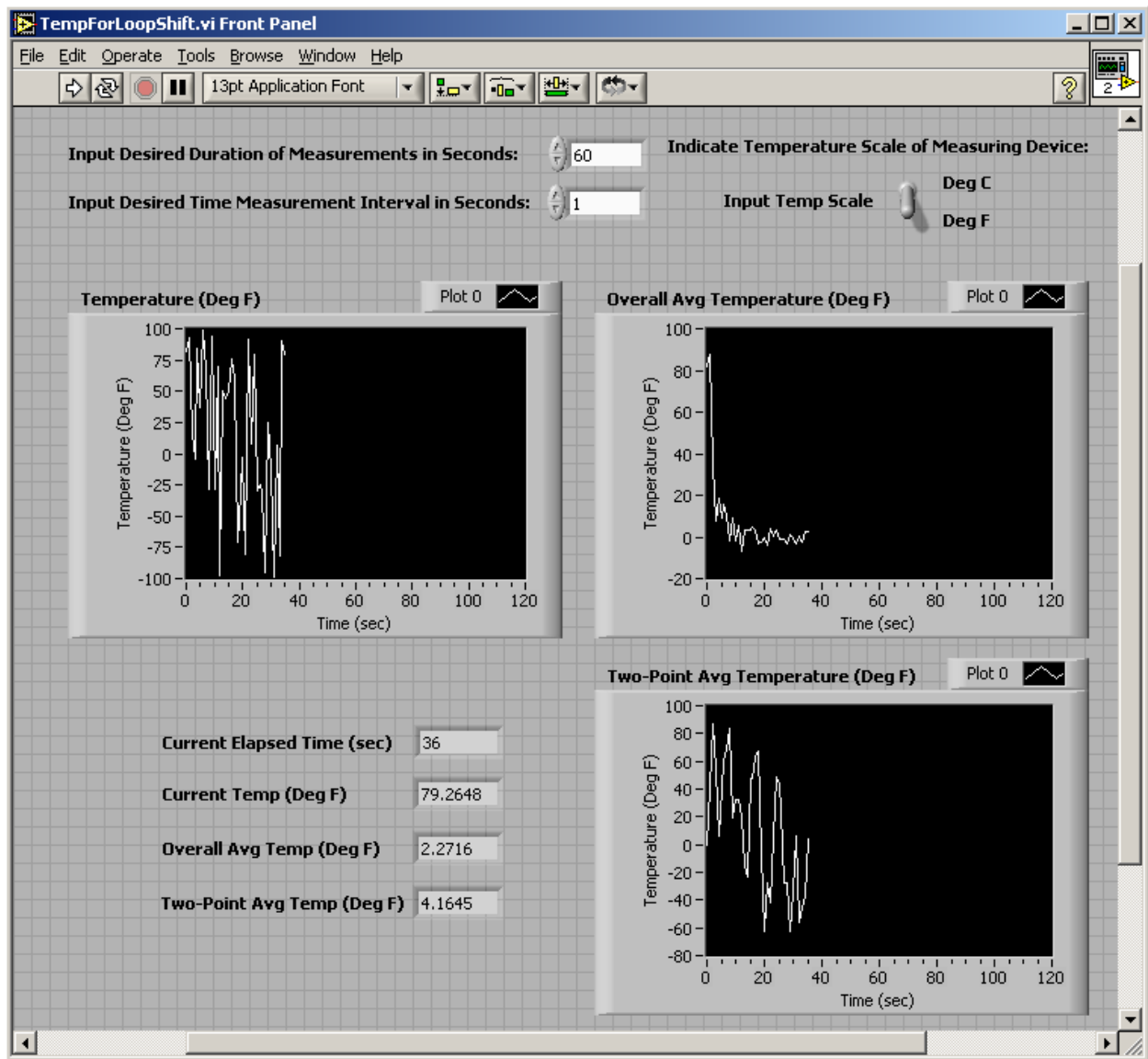


Figure 3.6: Front Panel for Temperature Averaging VI

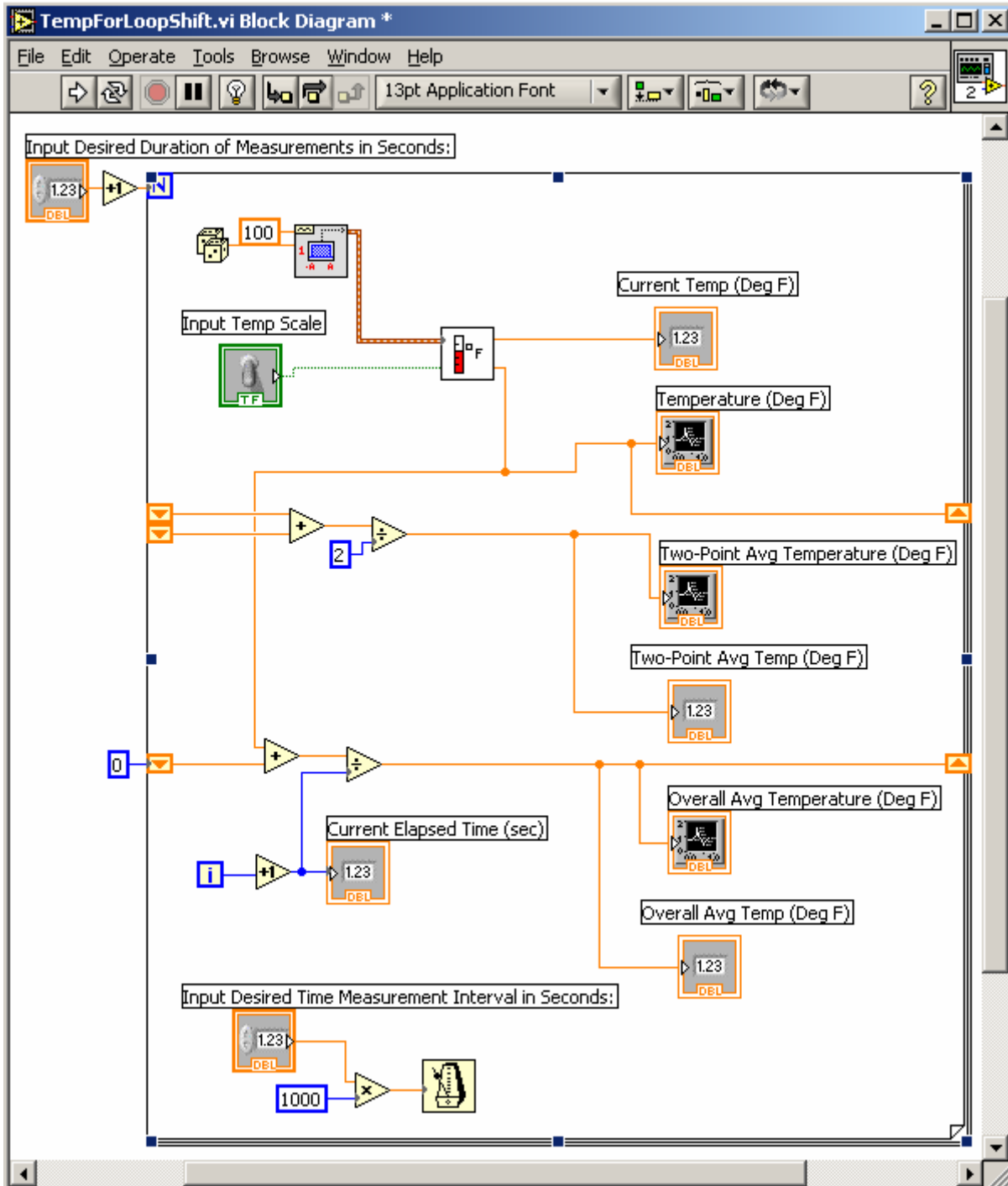


Figure 3.7: Block Diagram for Temperature Averaging VI