Introduction to LabVIEW

ES110

Graphical programming language & Data flow

- LabVIEW relies on graphical symbols rather than textual language to describe programming actions
- The principle of dataflow, in which functions execute only after receiving the necessary data, governs execution in a straightforward manner

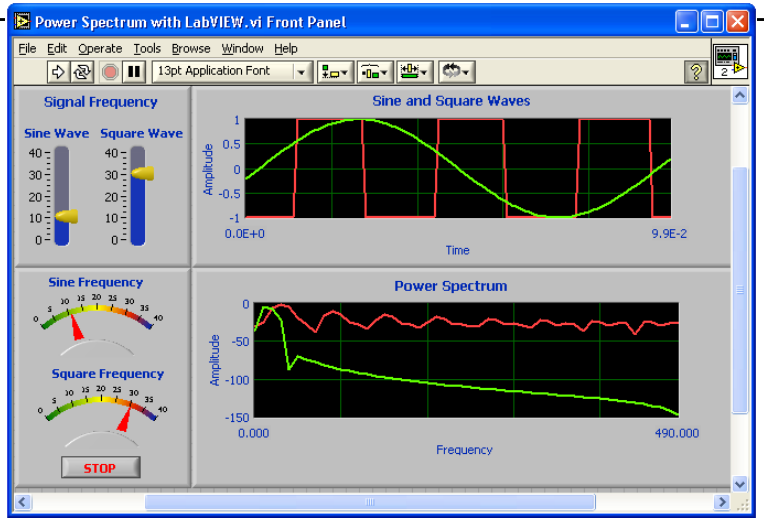
How does LabVIEW work?

- LabVIEW programs are called:
  - *Virtual Instruments (VIs)*
  - because their appearance and operation imitate actual instruments.
- However, they are analogous to main programs, functions and subroutines from popular language like C, Fortran, Pascal, …

LabVIEW Programs Are Called Virtual Instruments (VIs)

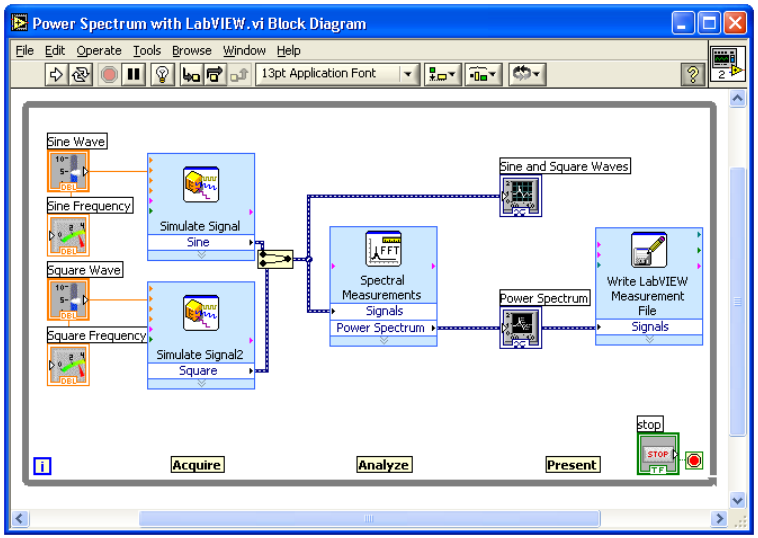
Front Panel

- Controls = Inputs
- Indicators = Outputs



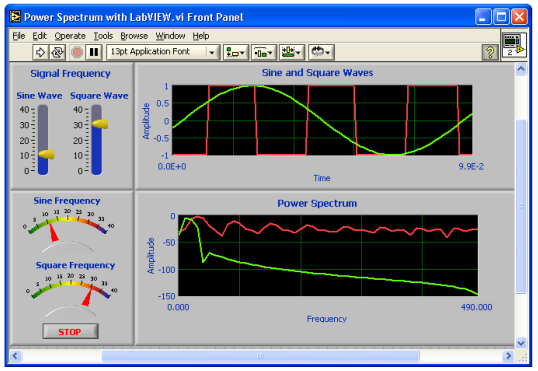
Block Diagram

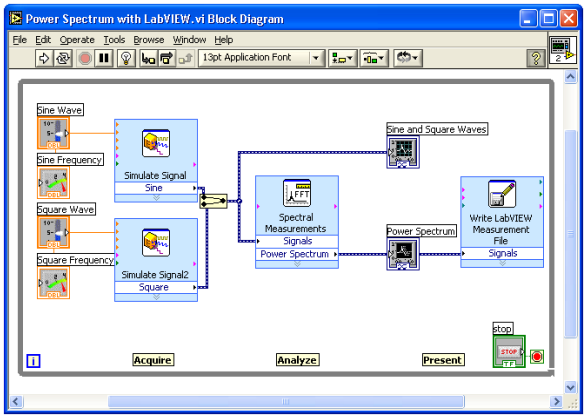
- Accompanying "program" for front panel
- Components "wired" together



LabVIEW Introduction

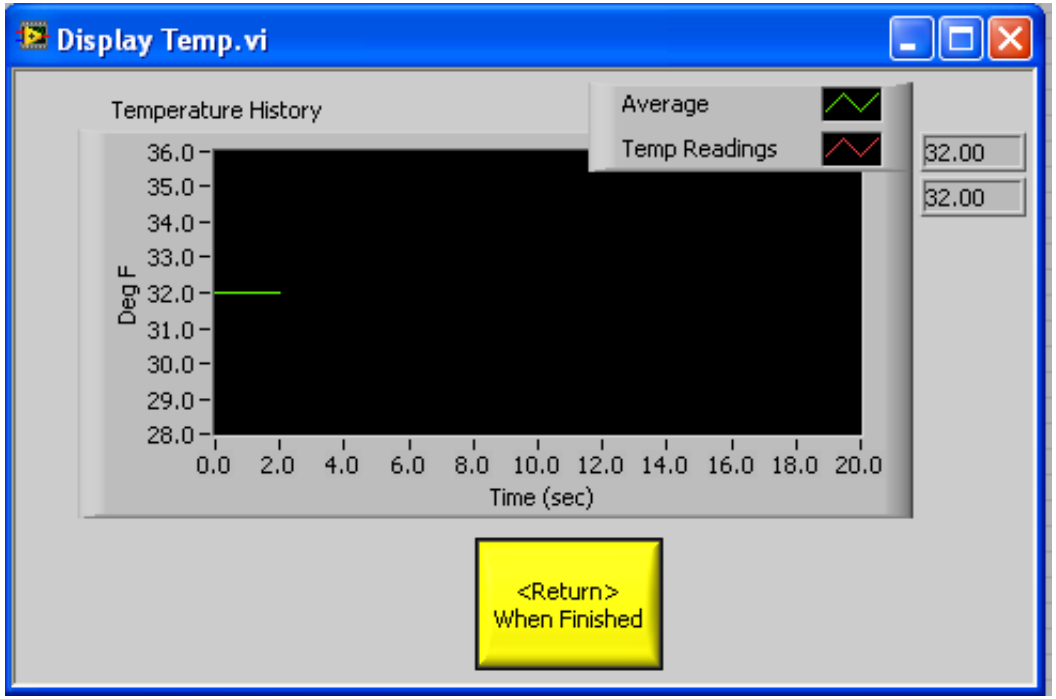
- Two “sets” for development
  - Front Panel
  - Block Diagram
- Wiring connections
- LabVIEW Conventions
- Running LabVIEW programs





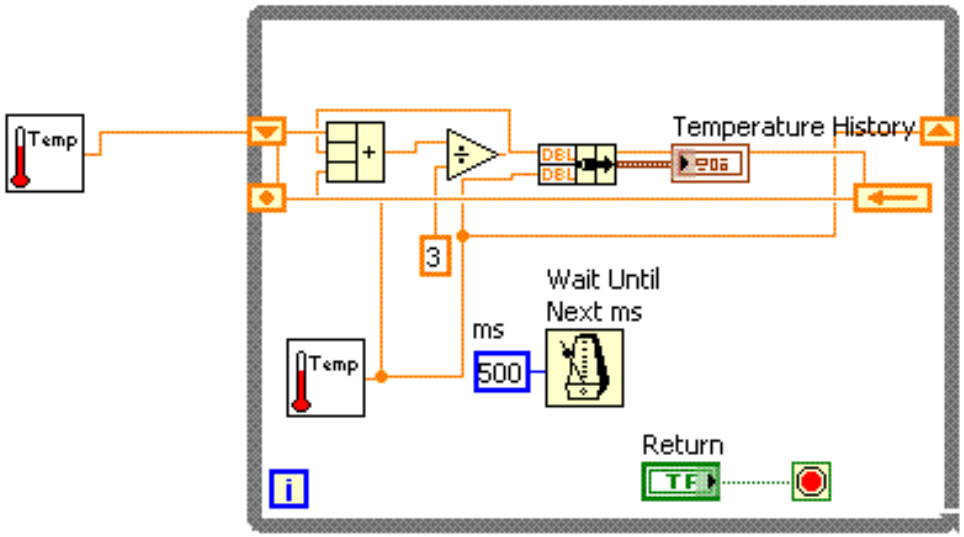
LabVIEW Front Panel

- All user interface goes here!
- Used to display Controls or Indicators
- Highly customizable



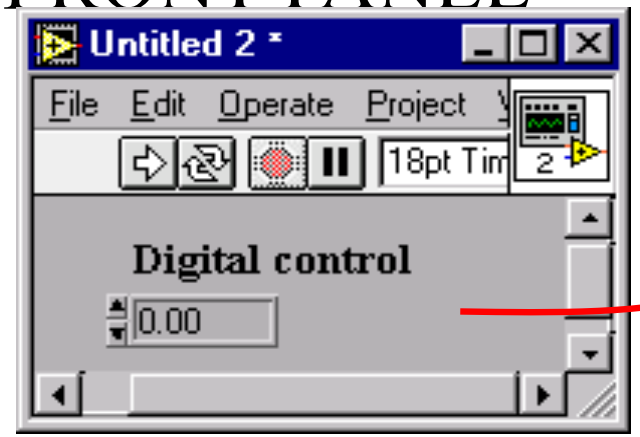
LabVIEW Block Diagram

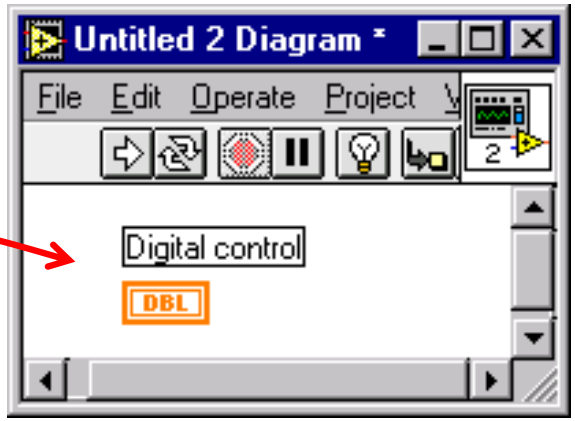
- Actual program
- Invisible to user
- Read left to right, like a book
- Where the MAGIC happens!



Terminals

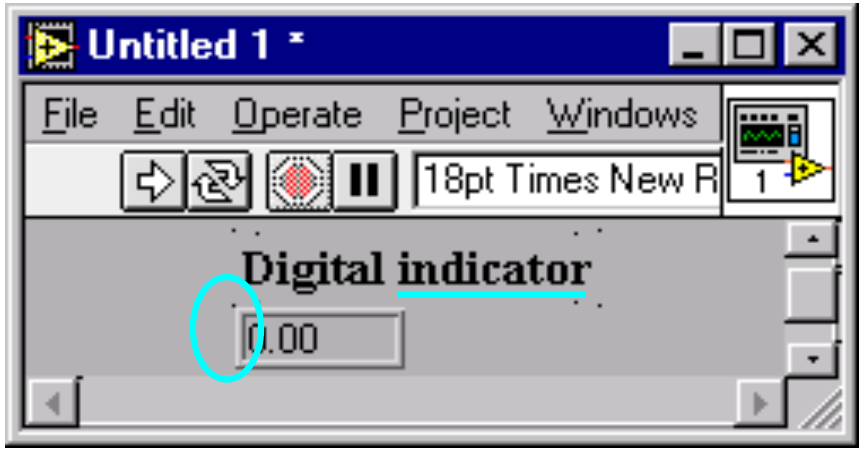
When you place a *control* (or *indicator*) on the FRONT PANEL





LabVIEW automatically creates a corresponding *control* (or *indicator*) *terminal* on the BLOCK

Control? *or* Indicator?



Controls = Inputs from the user = Source Terminals

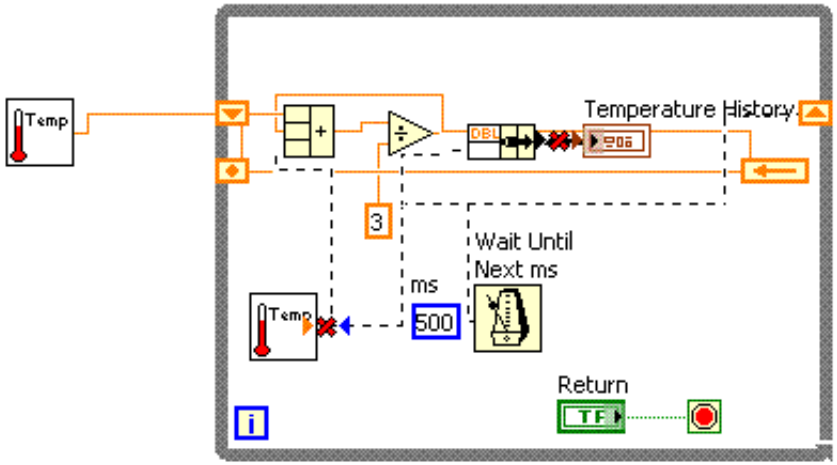
Indicators = Outputs to the user = Destinations

Manipulating Controls and Indicators

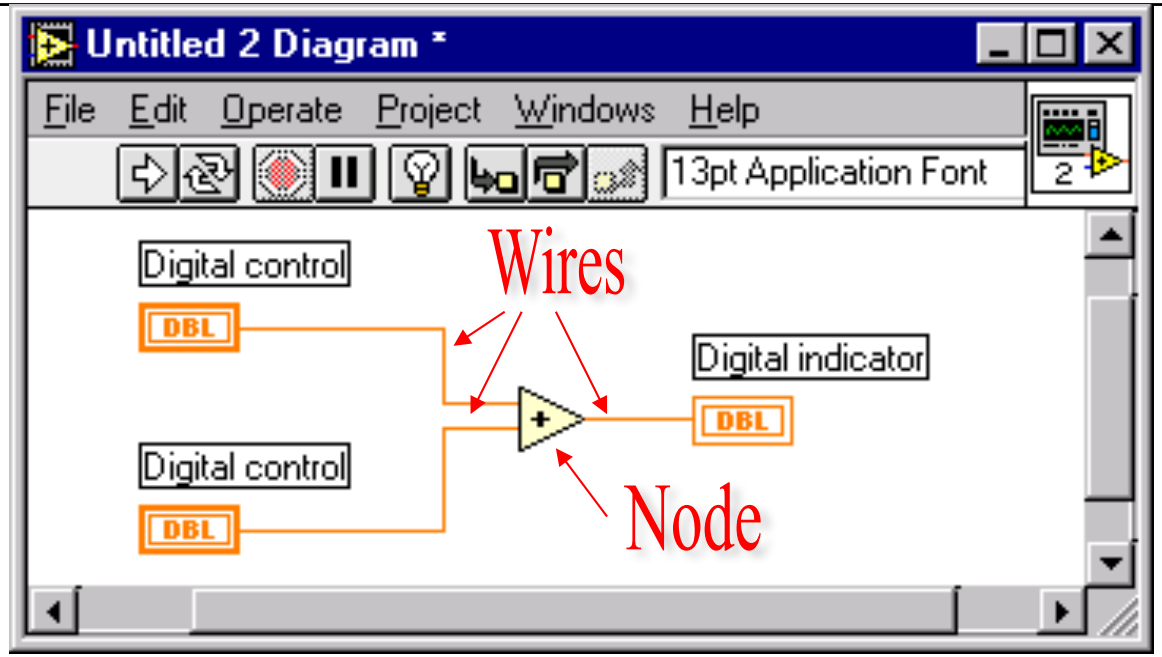
- Right click on an indicator to
  - Change to control
  - Change format or precision
- Right click on a control to
  - Change to indicator
  - Change mechanical action (whether to latch open or closed, and what to use as default…)

Wiring Connections

- Wires transport data through the block diagram
- Wire color indicates variable type
- A red “X” means something is wrong!

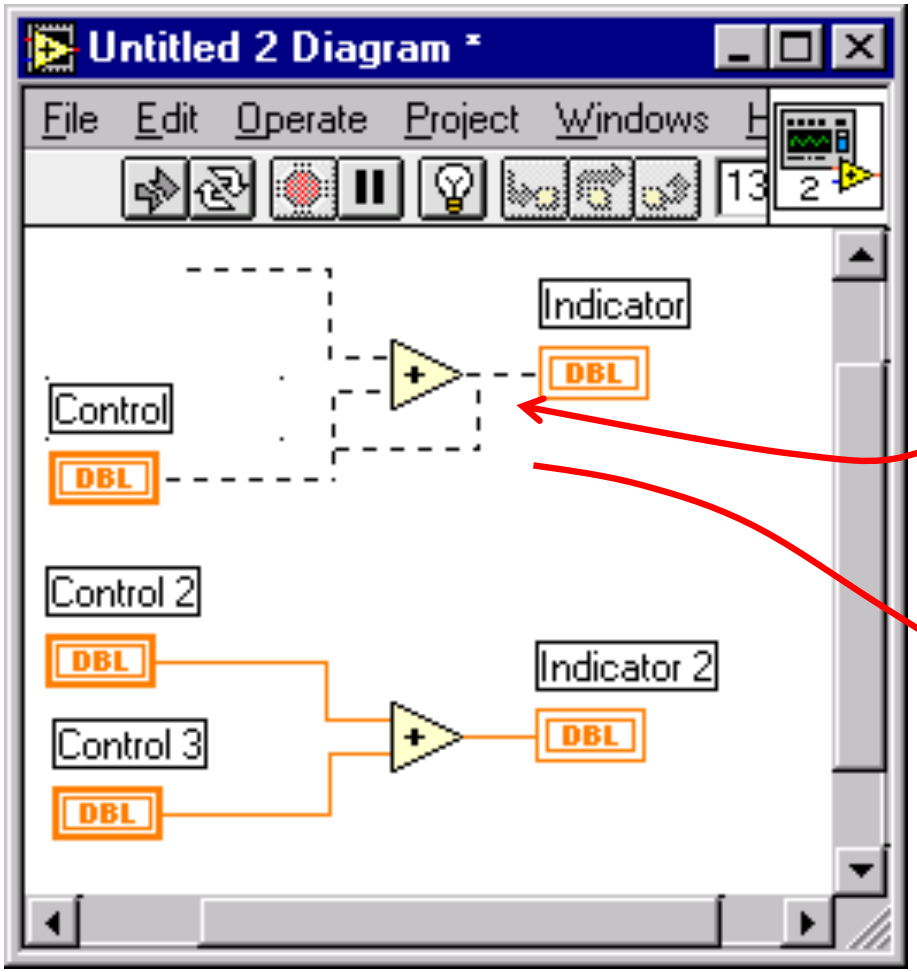


Wires



A LabVIEW VI is held together by *wires* connecting nodes and terminals; they deliver data from one source terminal to one or more destination terminals.

Broken wires

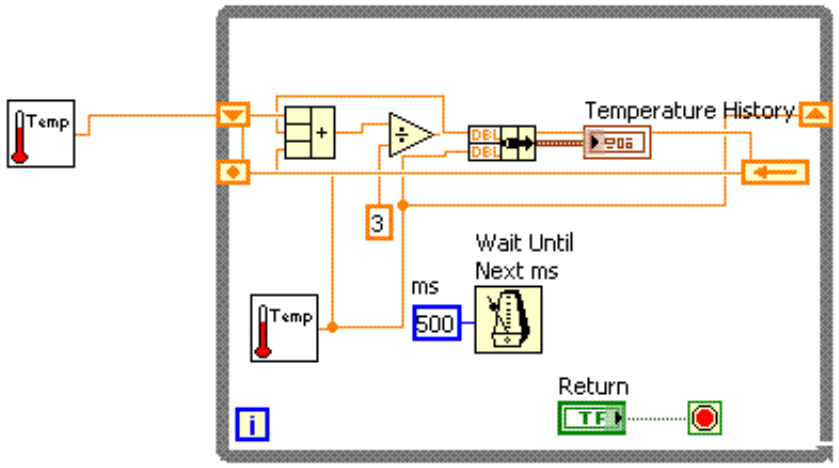


If you connect more than one source or no source at all to a wire,

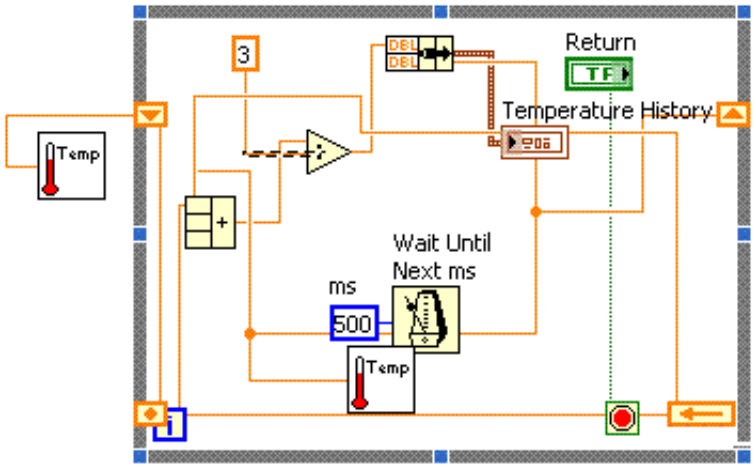
LabVIEW DISAGREES with what you're doing, and the *wire* will appear *broken*

Messy vs. Clean Wiring

CLEAN: Easy to troubleshoot



MESSY: What is going on?



Basic wires used in block diagrams and corresponding types

Each wire has different style or color, depending on the data type that flows through the wire:

	Scalar	1D array	2D array	Color
Floating-point number				orange
Integer number				blue
Boolean				green
String				pink

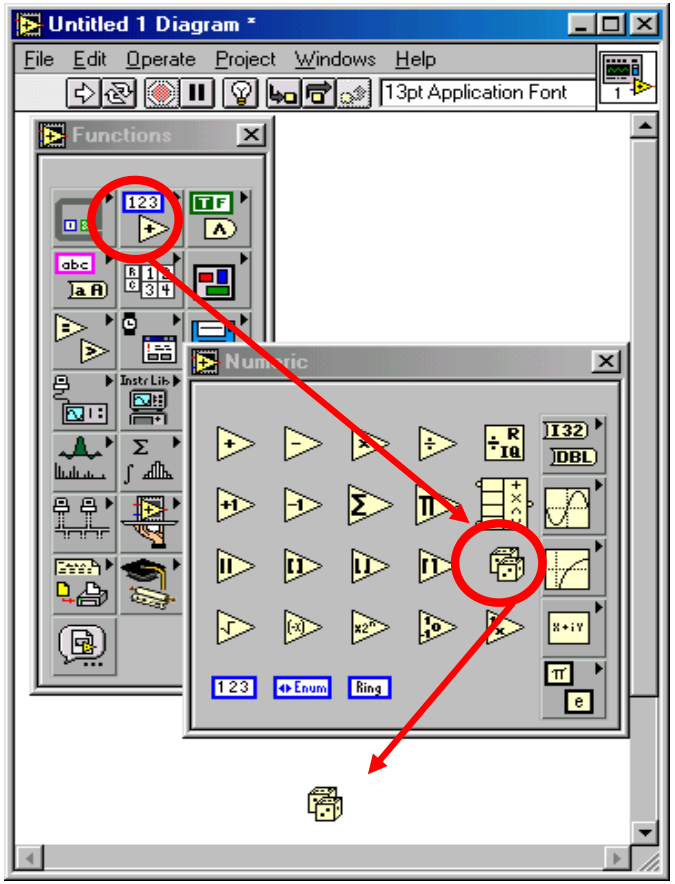
LabVIEW Conventions

- Front panel items
  - Controls and indicators
- Block diagram items
  - Program structures (loops, case structures, math, etc.)
- Controls vs. Indicators
  - Wires attach to controls on the right (give values)
  - Wires attach to indicators on the left (receive values)
- Wiring colors
  - Wires are color coded to correspond to data types

Running LabVIEW Programs

- ALMOST ALWAYS put your program in some sort of loop that can be stopped with a control

- AVOID using the red “x” to stop your program



Lab. Equipment

- Oscilloscope

Universal Measuring Instruments

- Function Generator

Signal Generator

- Digital Voltmeter (DVM)

Tools palette . . .

Select a feature to edit or move

Add/edit text

Operate a control

Probe Data (troubleshoot)

Wire features together to control flow of data

Controls palette . . .

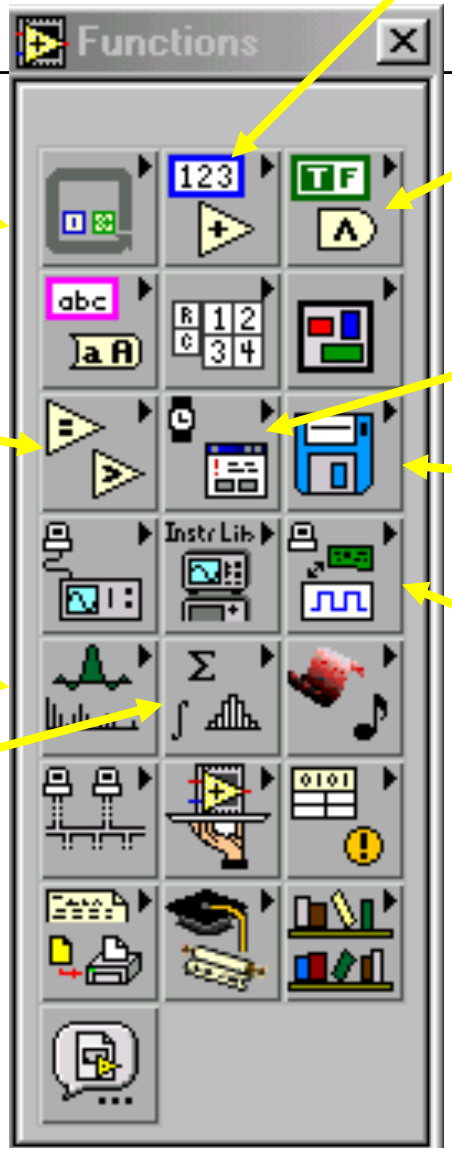
Insert a boolean control
(button or switch)

Insert a digital
indicator or control

Functions palette . . .

Add a numeric operator (+,-,…)

Add a structure such as for, while, and case statements



Add a boolean operator (and, or…)

Timing/dialog

Comparison

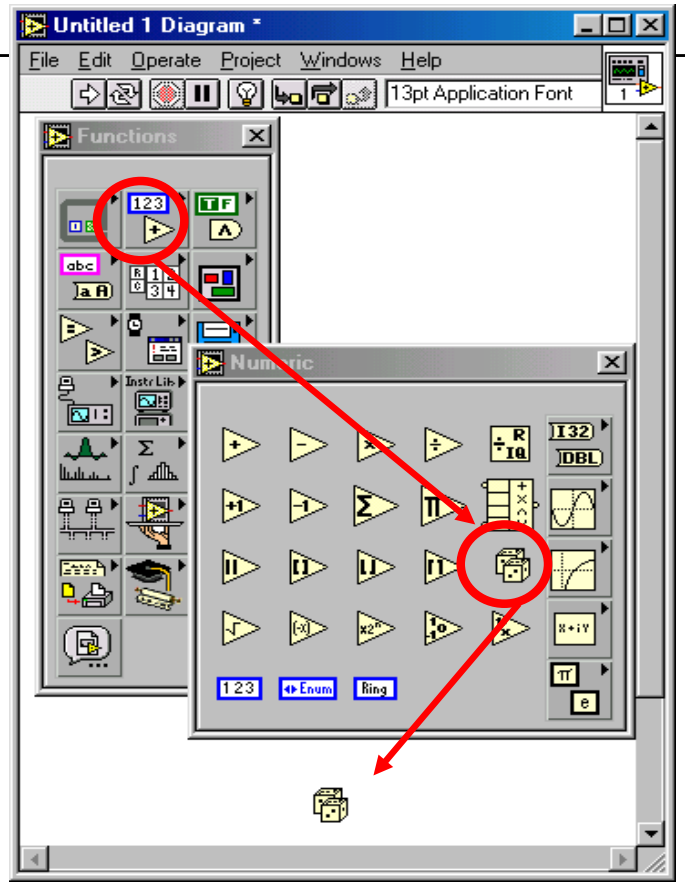
File I/O

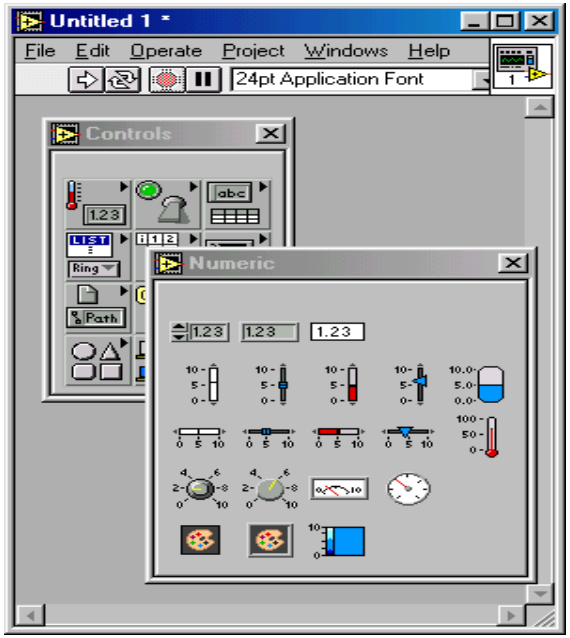
Signal analysis

Data Acquisition

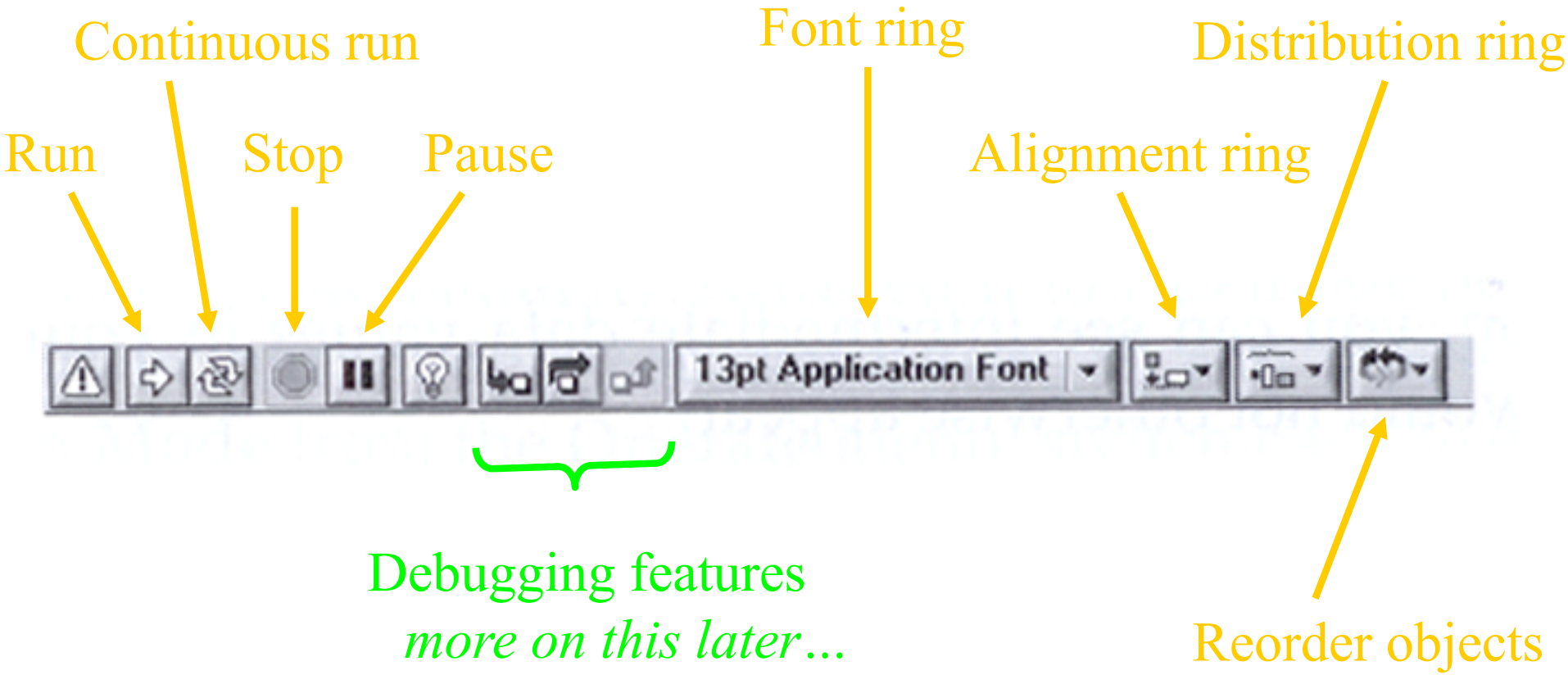
Mathematical Functions

Subpalettes . . .





Toolbar . . .


Continuous run
Font ring
Distribution ring
Run
Stop
Pause
Alignment ring
13pt Application Font
Debugging features
more on this later…
Reorder objects

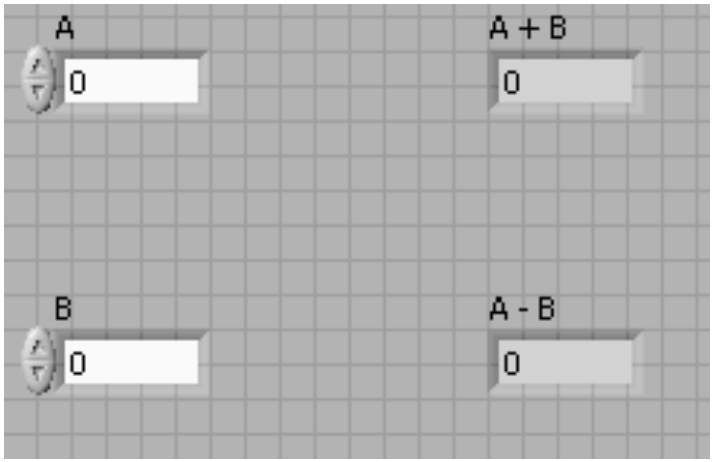
The Run Button

- The Run button, which looks like an arrow, starts VI execution when you click on it
- It changes appearance when a VI is actually running.
- When a VI won't compile, the run button is broken

Examples

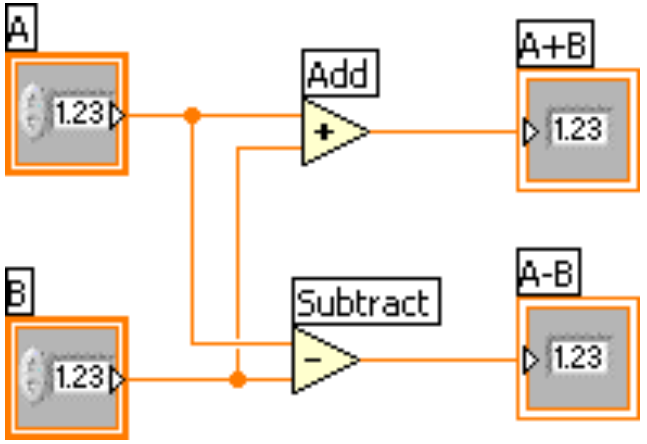
Creating a VI

Front Panel Window



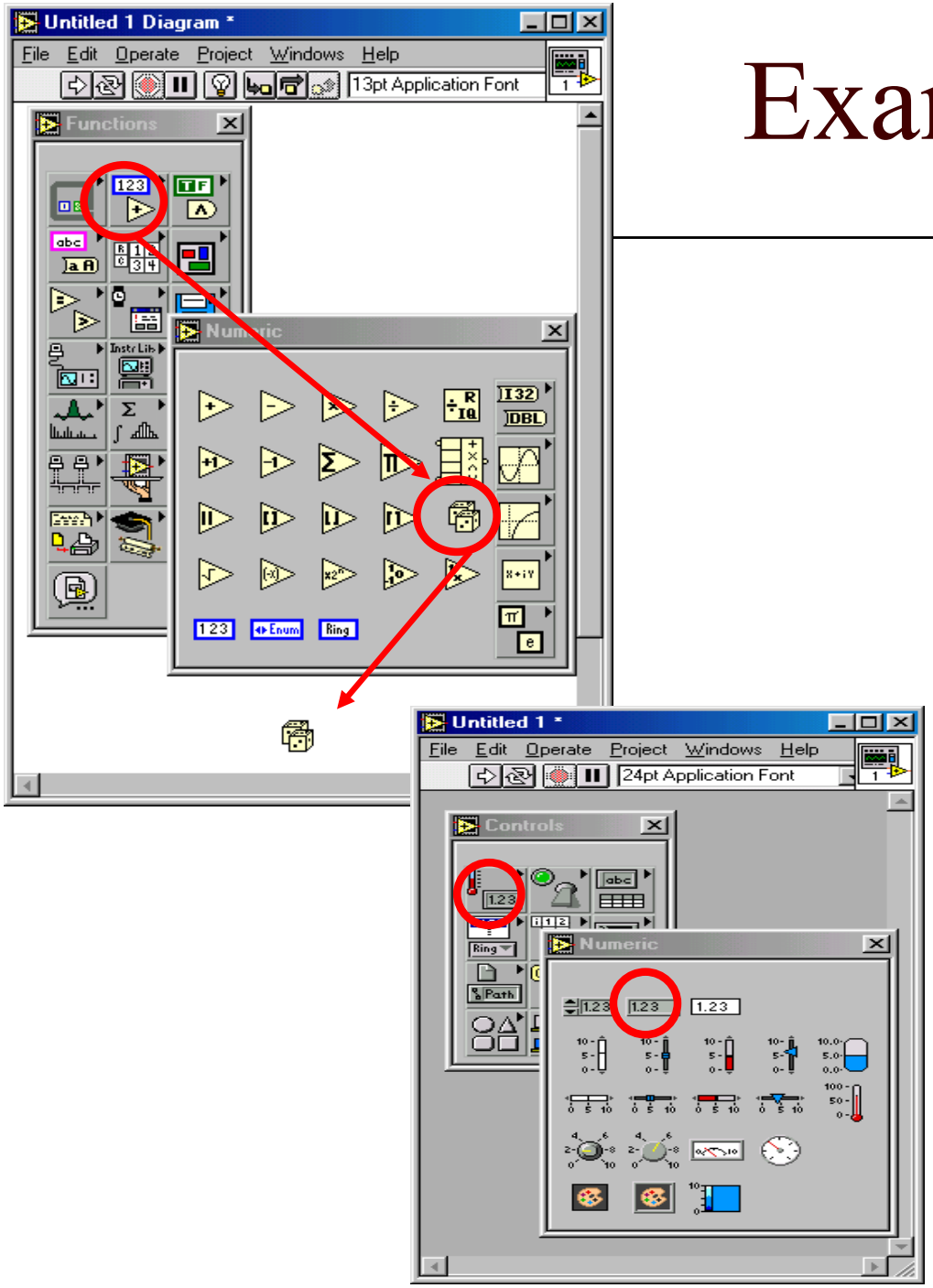
Control Terminals

Block Diagram Window



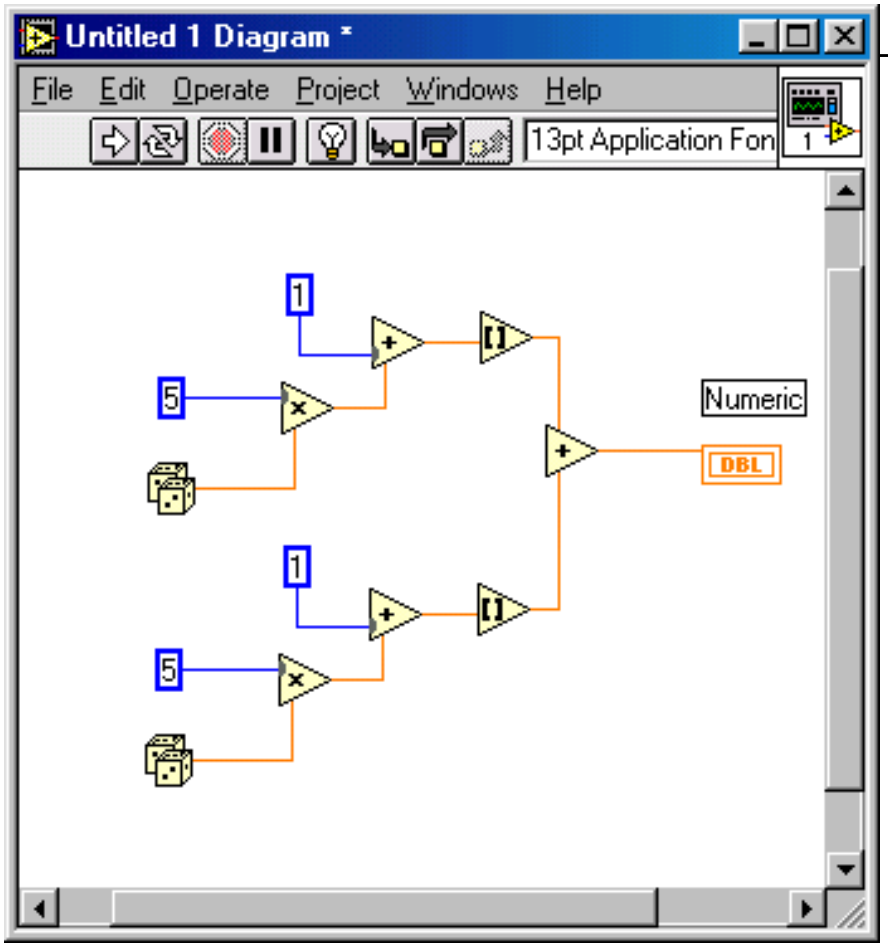
Indicator Terminals

Example 1: Craps



- From the functions – numeric panel insert a pair of dice
- From the Controls panel insert a numeric digital indicator (on the front panel)
- Use the wiring tool to connect the two (in the wiring diagram) and click the “run” button repeatedly.
- Numbers from 0.00 to 1.00 should be displayed in the front panel

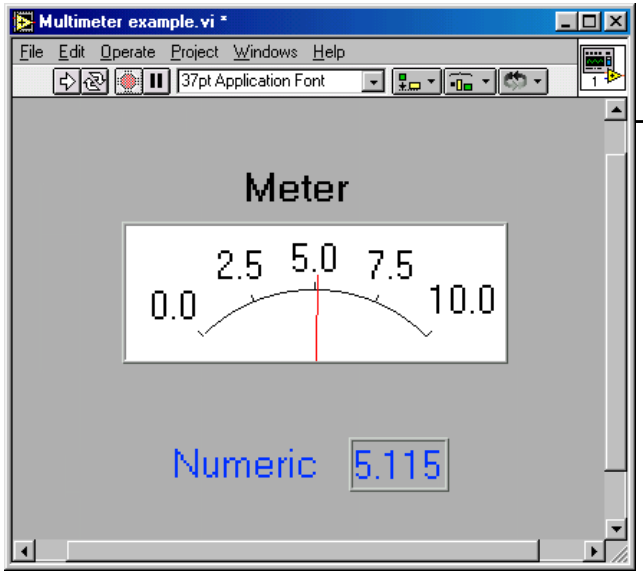
Example 1: Craps (continued)

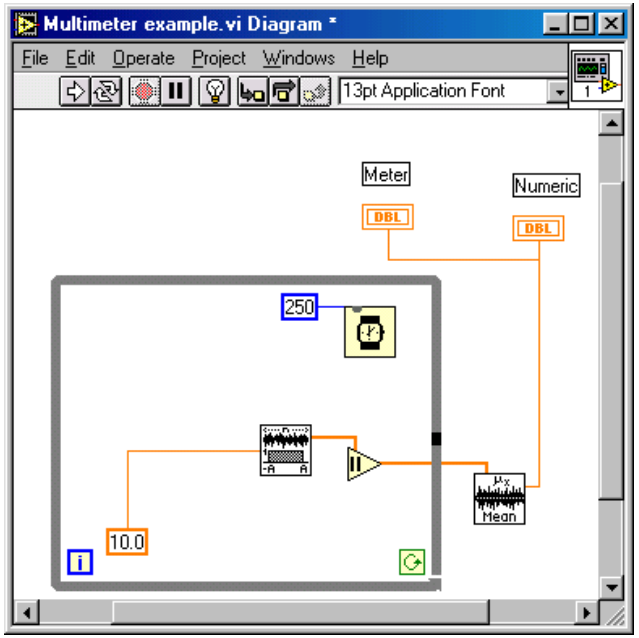


This wiring diagram simulates the rolling of 2 dice and their addition to form a number from 2 through 12.

- Delete the wire
- Add a multiplication node and a numeric constant to allow multiplication by 5
- Add an addition node and numeric constant to allow addition of 1
- Add a mathematical "Round to Nearest" node.
- Make a second copy of this structure to represent a second die and wire them together through an addition node with an output to a numeric constant

Example 2: Analog & Digital Voltmeter *(simulated signal)*

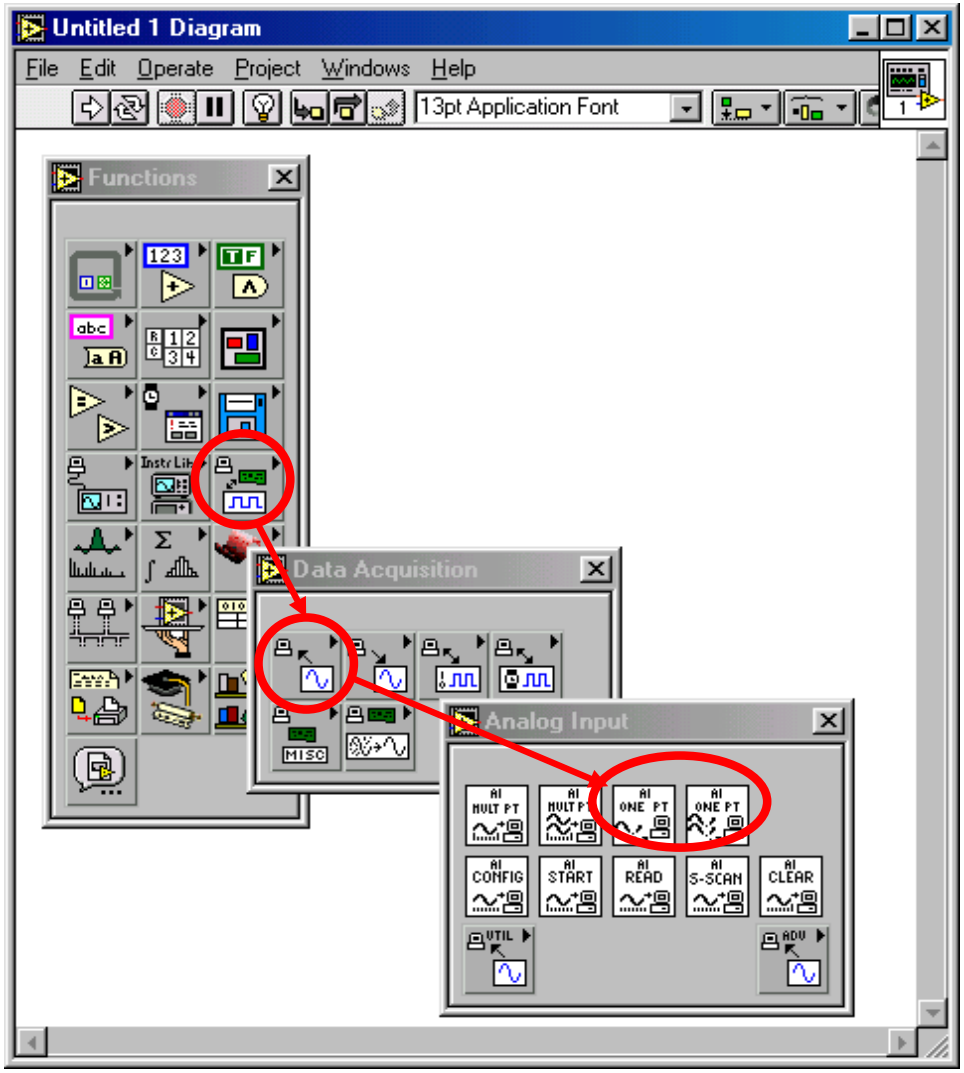




- Uniform noise used as simulated signal – Functions – Signal Processing – Signal Generation menu
- Absolute value function from functions – numeric menu
- Mean value of data series from the functions – mathematics – Probability and Statistics menu
- The 250 ms wait implemented from the time and dialog menu slows the “flutter” of the meter.

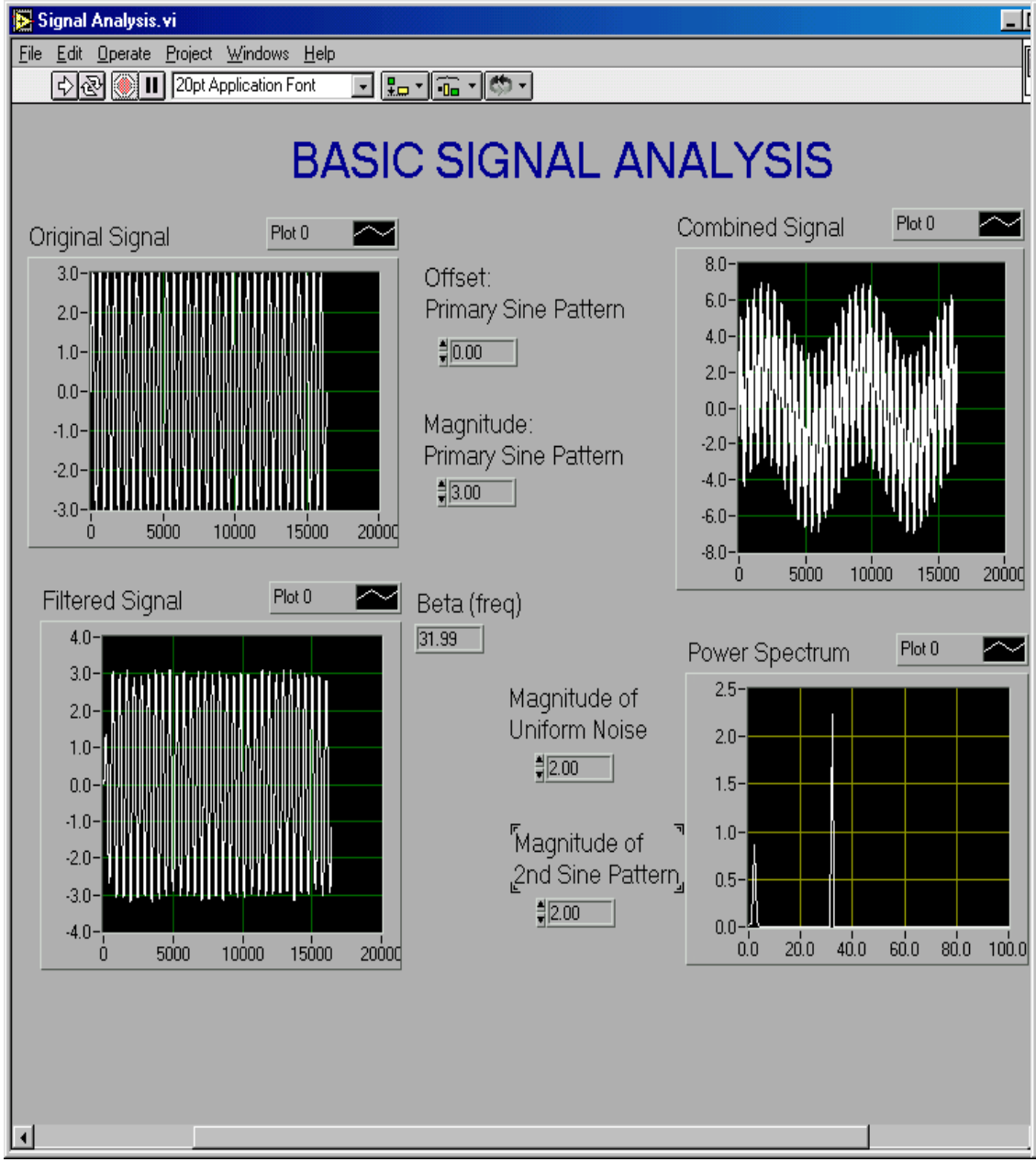
Example 3: Reading an analog input signal

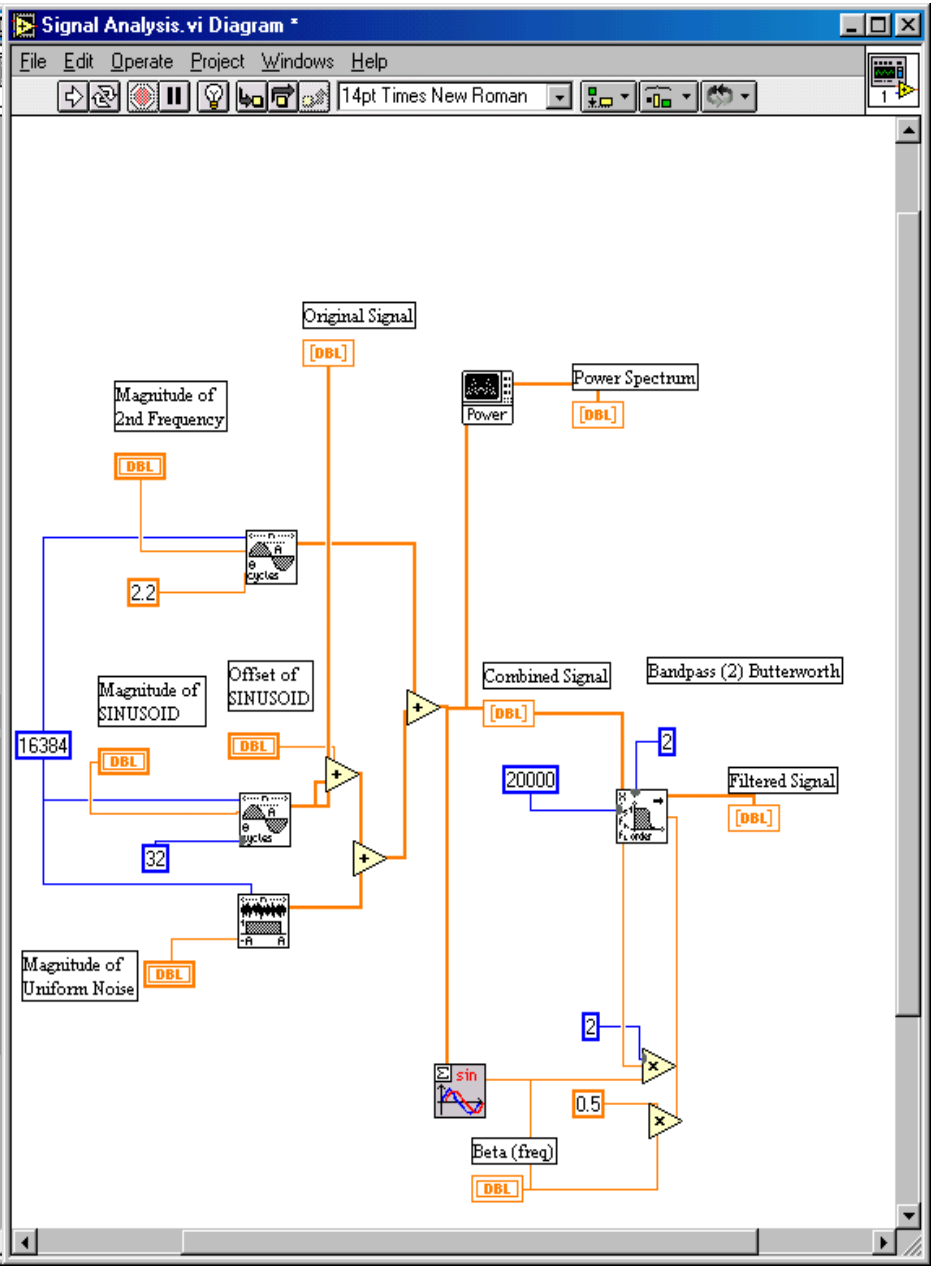
Requires A/D board to implement



- From the functions menu select data acquisition and then analog input. Then select either "Sample Channel" or Sample Channels"
- This places the sampling icon in your wiring diagram
- You then need to configure the channel(s) and wire the output to other parts of your program.

Example 4: Signal Analysis (continued)





Example 5: Creating Sub-VIs

- In wiring diagram use selection tool (mouse box) to select all items to be in the SubVI.
- From Edit menu select “Create SubVI”
- Double click on new icon and save it as a separate VI.
- Cut-and-paste it at will or insert it using “Functions – Select VI menu”