Analytisch-Physikalische Chemie

Teil Physikalische Chemie
Sommersemester 2007

Measurement and Data recording using graphic programming provided by LabView (LBV)

Matthias Geibel, *Studiengang Chemie*, *4. Semester*, mgeibel@student.ethz.ch
Daniel A. Frick, *Studiengang Chemie*, *4. Semester*, frickd@student.ethz.ch

Assistant: Nassir Mojarad

Abstract: Five different programs were programmed using LabView. The first was a simple calculator, which was then improved, so it could be choose which basic operation should be carried out. The third improvement was adding a comparison of two obtained numbers and return which is larger. The second task was to create a program that calculates the temperature (in Celsius and Kelvin) using the ideal-gas-law and the input of the pressure, volume and the number of mols. The third task was to familiarize with plots. Three random numbers were averaged and displayed using a tuneable time delay. Task 4 was to approximate a function of time by a series of sinus. Displaying the result as a plot of the value of the function and using a time delay. In the last task a program which assigns a certain randomly calculated value to a space, defined by the bin in which the random number lies, in an array. The array is then plotted to observe who the distribution of the values. By using the formula x^2, it could be observed that the distribution is raising quadratic.

Zürich ZH, 26. April 2007

Matthias Geibel

Daniel A. Frick

1. Introduction

1.1. Theory

LabView – General

LabView is a graphic programming utility, with the aim to create easy capturing, analyzing and displaying program to support scientist and engineer in their daily life.

The used function

The basic operations of LabView are addition, subtraction, division and multiplication. An input is linked with the operation; the output of the function is linked to the output.

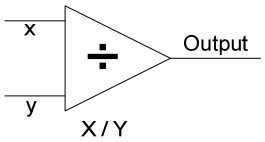


Abb. 1. Example for the division.

Other functions which were used are the comparison operator and the equality operator. To differentiate cases, a case box was used; in there different cases could be described, by enter a specific case number, the different cases could be addressed and executed. To repeat the same task several times to different loops were used. The first was a while-loop, which has ending-signal, which could be for example a switch. The second loop is a for-loop, were the ending signal is an integer which is counted up until it's the same as entered from user.

To communicate with the user there is a programmed feature which returns a box with defined information and an additional box to accept the return.

Instead of using always the basic operation, defined by LabView a formula-node could be used, which is in general a box were directly mathematical terms can be calculated by inserting values true tunnels in the node and getting values back through the tunnels.

To display information graphically there are plots which can be connected with an output of a program, together with a time delay signals, it shows the chronological sequence of a value calculated by a program.

To sort a bunch of numbers in table, an array could be used, this has to be set up first by defining the size and start values in the array. The values of an array can be picked by using the index, manipulated and than insert again at the same, or at a different position in the array.

To obtain an integer, the input has to pass through a special function called Int-32bit, the output is than an integer.

Experiment
2. Description of the Programs

2.1.1 Simple Calculator

The first task was to make a simple calculator which gets two numbers as an input. The first step was to program a calculator that gives the summation, subtraction, division and multiplication of the input at once. In the second step the program was modified so that it was possible to choose the operation. In the special case of division by zero the program gives an error.

2.1.2 Calculation and Comparison

In the second part of the first task a program was made that takes an input of two numbers. After calculating two equations with the input numbers it gives back the results as an output and compares the size of the resulting numbers.

2.2 Ideal Gas Law

A program was developed that gets as an input the values of pressure, number of moles and the volume of a gas. As the output it presents the temperature of the gas in Kelvin and Celsius.

2.3 Plotting a Graph

The devised program gives out a plot which shows the average of three random numbers varying in time. The time delay between two operations can be tuned in.

2.4 Calculation in loops

The task was to make a program that plots a sum of sine function which changes in time. The code was modified so that he number of sine functions and the frequency can be selected.

2.5 Integral of a function

The last program randomly picks up a point of a given function and calculates the corresponding value. The value is added to the bin value for this region. This procedure is repeated several times. Than the program plot the values versus the bins. The number of bins and the number of repetition can be regulated by the user

3. Analysis and Results

3.1.1 Task 1.1

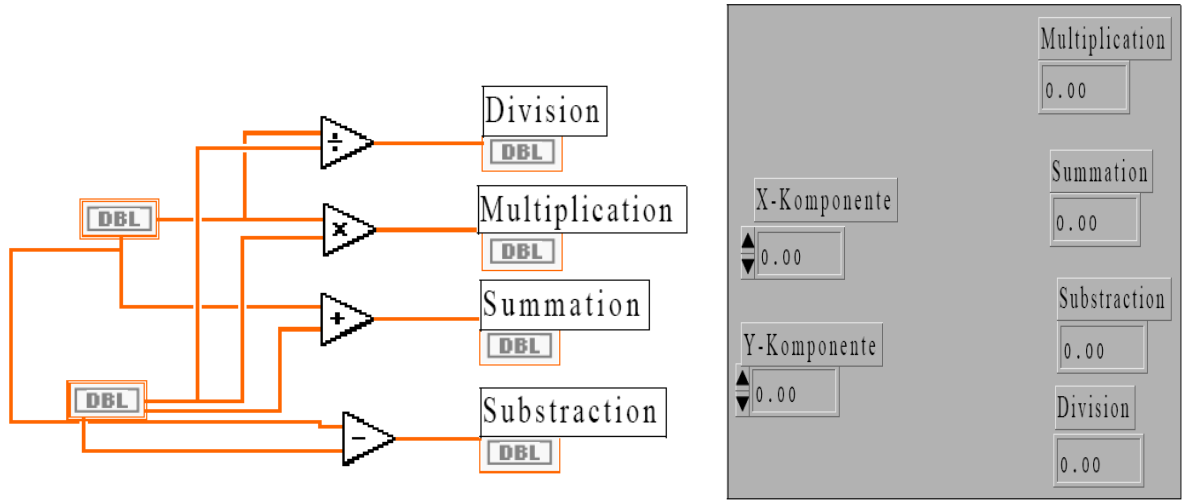
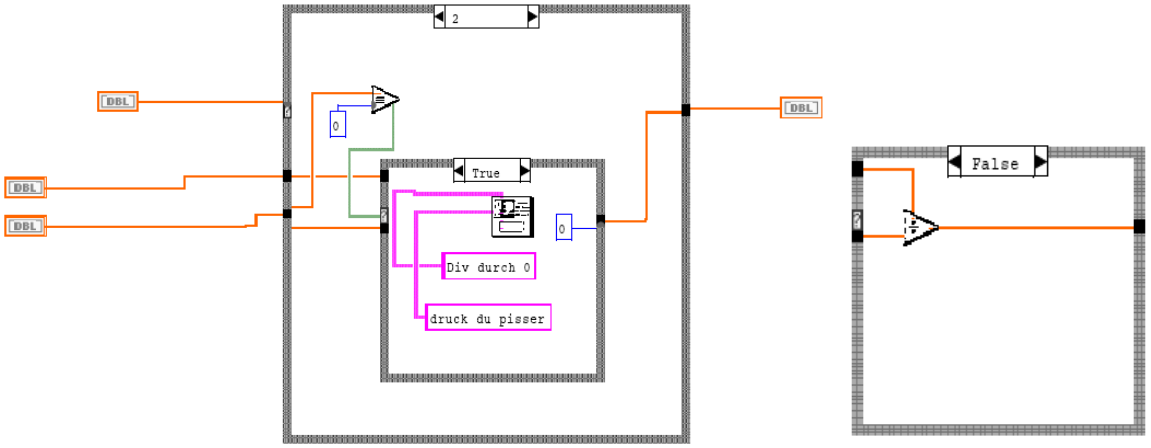


Figure 1: Block diagram (left) and front panel of task 1.1.

The input for the program is selected at the front panel with two digital controls. In the block diagram the input numbers are connected to nodes. The nodes are presenting an Add, a Multiply, a Subtract and a Divide function. The result of the functions goes to an indicator terminal and is shown in the front panel with a digital indicator. For each function there is a separate indicator.

3.1.2 Task 1.2



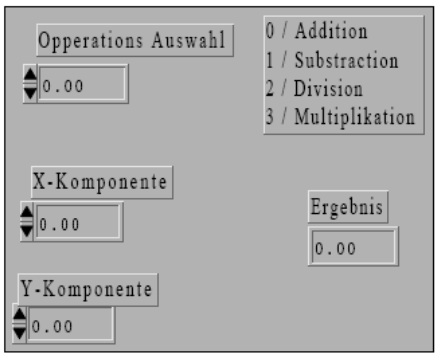


Figure 2: Block diagram for case 2 with both Boolean cases and front panel of task 1.2.

In the modified program there are 4 different cases. Each case presents a function and can be selected in the front panel. The input of the two numbers and the selected function are connected to a case structure. The input of the choosed function is connected to the numeric control of the case structure. For example the input one gives case one. The input of the numbers is connected in each case to a node for addition subtraction, division and multiplication. In the division case a Boolean case structure is integrated. First the program controls whether the input of the y-component is equal to zero. The result of this Equal node goes to the boolean control of the Boolean case structure. If the selector terminal is “True” the value zero is given back to the case structure and from there to the output. Additionally a message is sent to the front panel which says “Division by o” and the user has to a press a button. In case the input is not equal to zero the “False” frame is executed. The values of the executed cases are sent to an indicator terminal outside the case structure and displayed in the front panel with a digital indicator.

3.1.3 Task 1.3

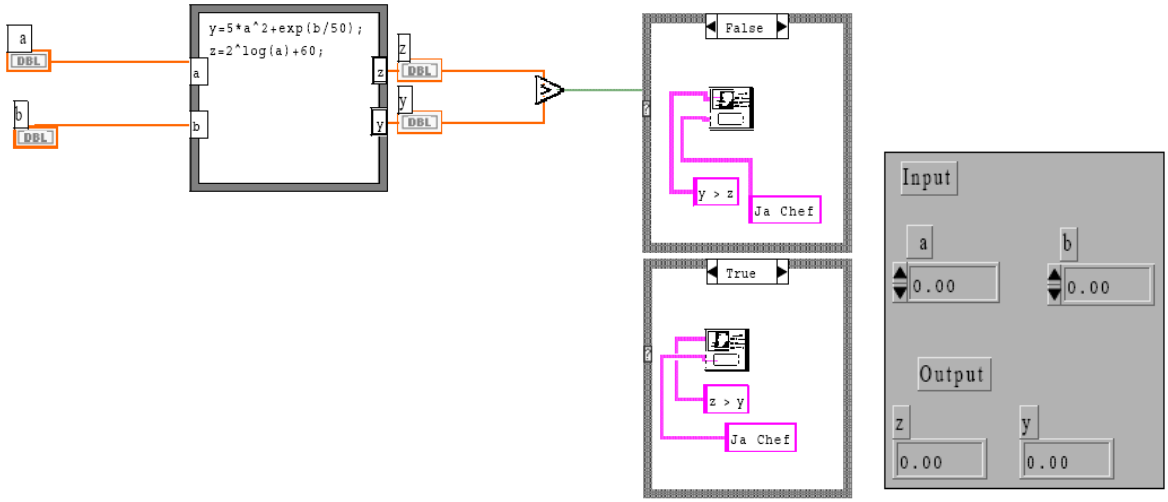


Figure 3: Block diagram and front panel of task 1.3.

The input of two numbers is connected to a formula node. In the formula node two equations are calculated. Each variable used in the formula node is declared as an input or output. The

declaration is done with an input or output box. The input boxes with the variables “a” and “b” are on the left side of the formula node and connected to the input indicators. The calculated values of the output variables “z” and “y” are sent to external indicator terminals. The indicator terminals are wired to the output boxes and to a node that compares the quantity of the numbers. The node is wired to a Boolean case. The Boolean case sends a message to the screen which informs the user which output variable it bigger.

3.2. Task 2

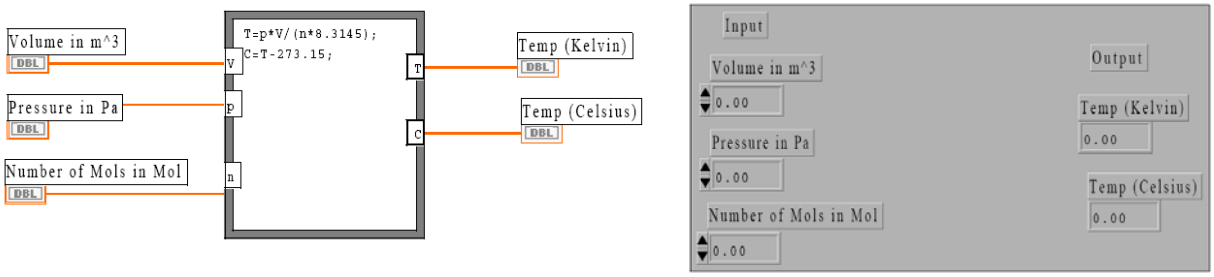


Figure 4: Block diagram and front panel of task 2.

To calculate the temperature of the gas a formula node is used. The inputs variables “v”, “p” and “n” are selected by the user in the front panel. The input indicators are connected to the input boxes of the formula node. Inside the node the temperature is calculated in Kelvin (variable T) and Degree Celsius (Variable C). The output boxes of the variables “T” and “C” are wired to separate terminal indicators and the result is shown in the front panel.

3.3. Task 3

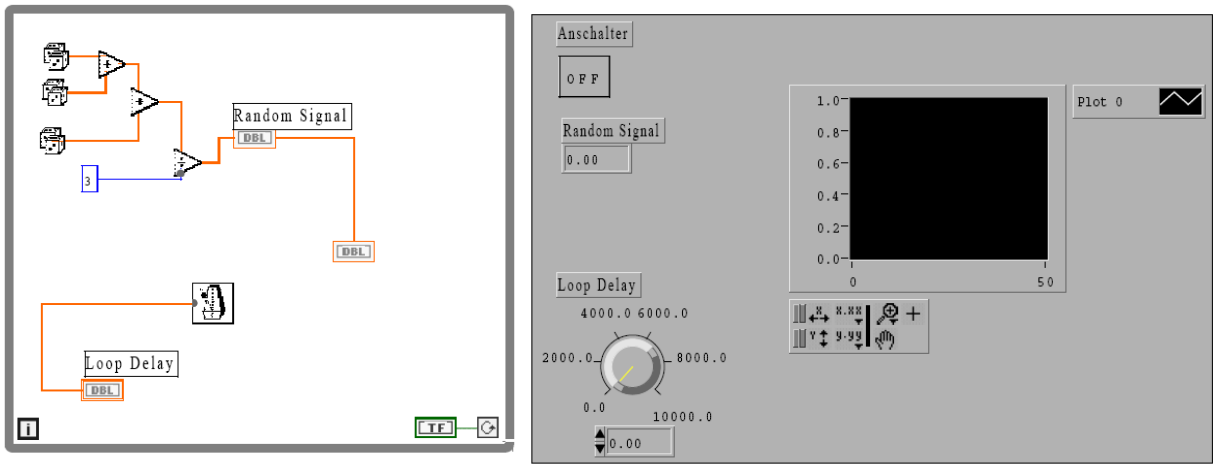


Figure 5: Block diagram and front panel of task 3.

The calculating of the average numbers is done with a while loop. Three random numbers between 0-1 are generated and added up. Afterwards the new number is divided by three with a divide node to get the average number. The divide node is connected to an indicator terminal of a waveform chart and a normal indicator terminal. The speed of the execution is controlled by using a Wait until next ms Multiple function inside the loop. So the loop iteration is executed

only once every specified number of milliseconds. The number of milliseconds can be selected in the front panel. The time node is connected to the indicator of the time input. The conditional terminal gets a Boolean input (true or false) from an Off/ON switch in the front panel. As long as the input in true (switch: ON) the loop executes.

3.4 Task 4

In Task 4 the gain was to obtain a program that could approximate a well behaving function as a sum of 10 sinus-function $f(t) = \sin(wt) + \sin(2wt) + \sin(3wt) + ... + \sin(10wt)$. The aim was to program that both the angular frequency and the time delay could be variable. A first possibility who a program could look like is Figure 6:

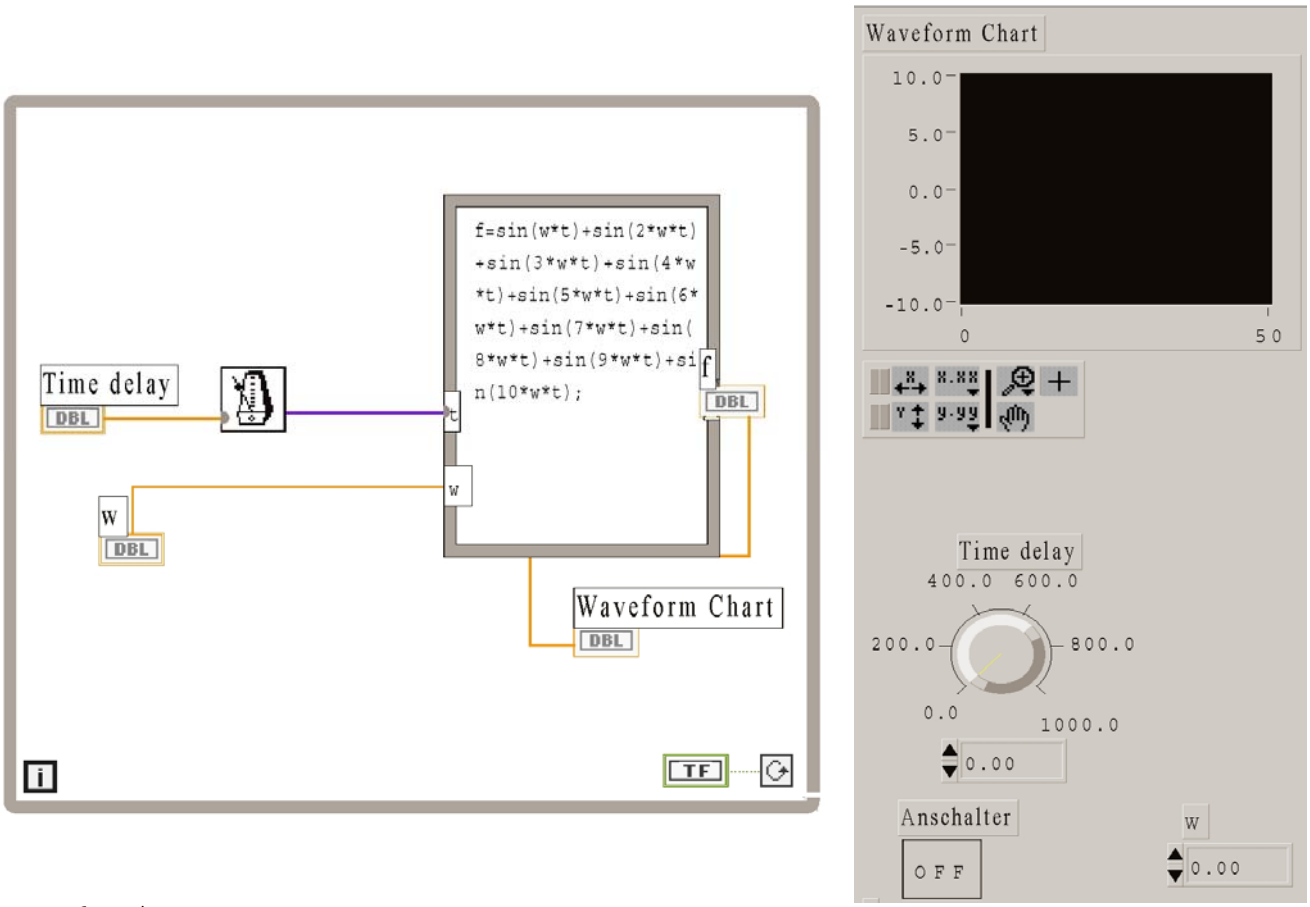


Figure 6: Task 4 – First program

As the program is dependent on the time, the condition for the while-loop is the "Anschalter", if it's in position "off" the program stops, other wise it will continue infinitely. The given values from the time delay are passed on to the timer, which produces a time t, with a certain delay. The timer signal is put in to the formula-node as variable t. The angular frequency is passed directly in the formula-node as variable w. The value of the function f is passed out of the formula-node and passed on to the chart, which will display the function as a function of time. The whole program is in the while-loop; the benefit is that the user can change the time delay or the angular frequency, without restarting the execution of the program. The disadvantage is that the numbers of sums are fixed on to ten; an adaptation to 12 sums would implement a

change in the formula-node, which is not practical. To improve the program the program was rearranged which is shown in Figure 7:

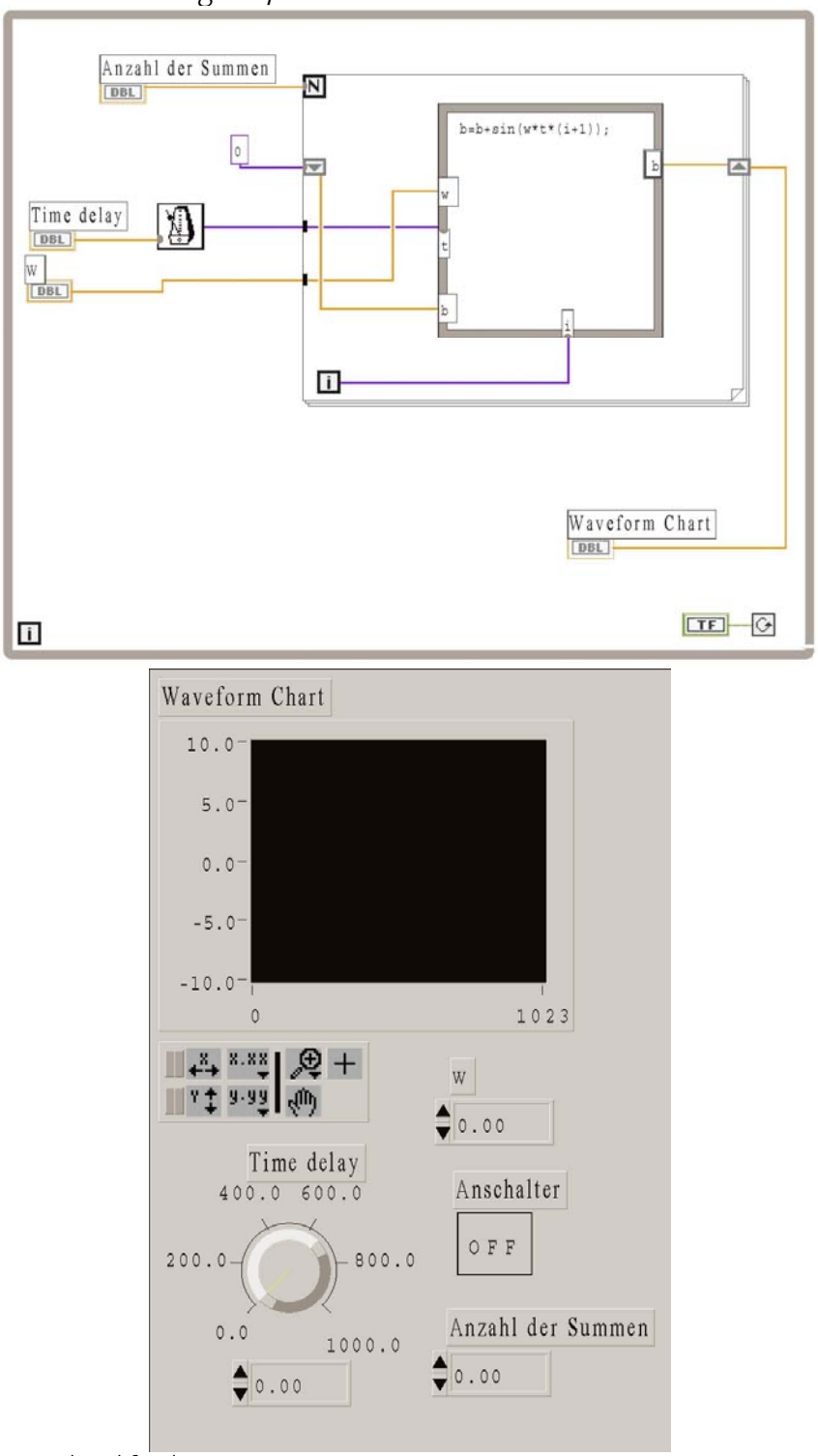


Figure 7: Task 4 – Improved and final program.

The aim was to program that sum in a loop, so the program could be scaled up from then sums to a user defined number of sums. The function $f(t) = \sin(wt) + \sin(2wt) + \sin(3wt) + ... + \sin(10wt)$ has to be adapted, instead of using a fixed

number, there has to be a variable: $f(t)_{n+1} = \sin((i+1)wt) + f(t)_n$. Because i starts at 0, there has to be added one to get the right result. The basis of the program is still the same, a while loop, with an on-off switch, to start and stop the program. The time delay and the angular frequency are now passes into a for-loop. The number of repetition of the for-loop is defined by the number of sum, which the user defined. In the for-loop a formula-node calculates the out of the angular momentum, the time and the variable i, which is the counter for number of times the for-loop is executed. The variable b, defined as 0 in the beginning, is the result of the formula-node is passed through a register, so for the next execution of the for-loop, the value of b is again available. After the maximum of number for the for-loop is reached the value of b is passed through a tunnel in to the while-loop to plot the function. Until the while loop isn't stopped by the user, it will continue to restart the for-loop again to obtain a new value.

3.5 Task 5

In figure 8 the diagram and the overview of task 5 is displayed:

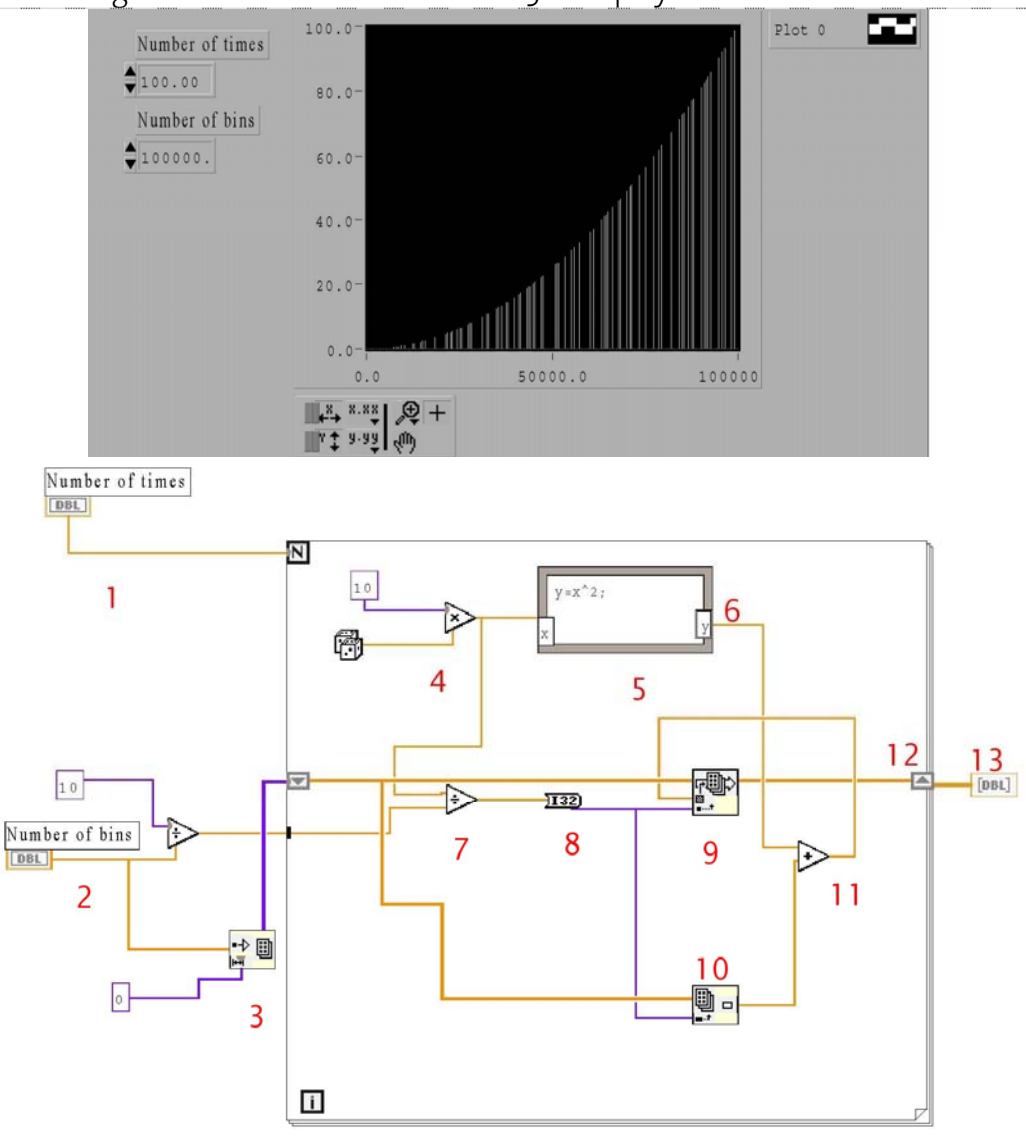


Figure 8: Task 5 –The final program.

In the user overview, the user can choose two settings, the number of bins (number of division

in the domain 0-10) and the number of steps.
The number of times is passed into the for-loop and defines how many times the for-loop will be repeated (1). The number of bins is divided through the size of the domain (in our case 10) (2), this value is passes into the for-loop. The number of bins is as well used to build up an array, which is filled up, in the beginning with zeros (3). A random number in the domain from 0 to 1 is generated and then multiply by the domain 10 (4), the calculated values is passed in to the formula-node, where x^2 is calculated (5) and than passed out (6). The calculated value from the random-number generator is divided by the number of bins (7), so the number of the bin, in which the calculated number of (5) is belonging to can be obtained. The number of (7) is passed through a 32-bit integer-function (8), this integer number stays now for the index in the array (9 and 10). The value which is already saved in the array is picked out, with the help of (8), added (11) up with (6), and the result is put again in the array. The array is passed through the register to get the value again in the next for-loop (12). At the termination of the for-loop the array is sent to the chart, which returns the number versus the bin.
It could be observed that the increase is quadratic, which is clear because of the formula used.

4. Discussion

The first three tasks were very useful to learn the basic principles of LabView. It was necessary to know the basics to solve task 4 and 5. There were no problems devising the first three programs.

4.1 Task 4

Both programs have their advantages. In the first one there is only one while-loop, which allows the program to run faster. 10 calls of a function costs more power than just adding up the function. The disadvantage of this solution is that it isn't scalable, so if the number of sums should be adapted to a new problem, the diagram has to be changed.
The difficulty was to find out how it works with passing a value from a for-loop into the for-loop again using the register. Because of a read over the sub-task to program the frequency into the program instead of the angular frequency was forgotten. But it could be easily integrated by the definition of $\omega = 2 \cdot \pi \cdot f$.

4.2 Task 5

The challenge in this task was to find out how arrays could be created, used and modified. In the used LabView version the function 32-bit integer could not be found in the toolbox, so it was copied from another group into our project.
After setting up the program it was observed that instead of giving back the true x-values, the x-axis is labelled with the number of bins. It was not possible to change it, so it would show the true x-values.

5. References

[1] Meister, E., Grundpraktikum in physikalischer Chemie – Theorie und Experimente, VDF Hochschulverlag, Zürich, 2006.