Oscilloscope GUI Project

After completing the Pair Game project, my internship advisor asked me to design a remote-control interface for my last project, the Rohde&Schwarz RTA4004 Oscilloscope. The oscilloscope I used in this project is shown in the image below.

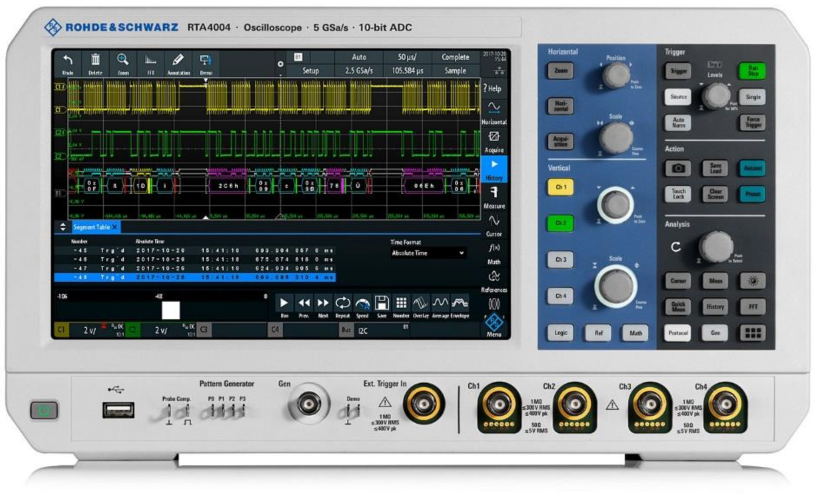
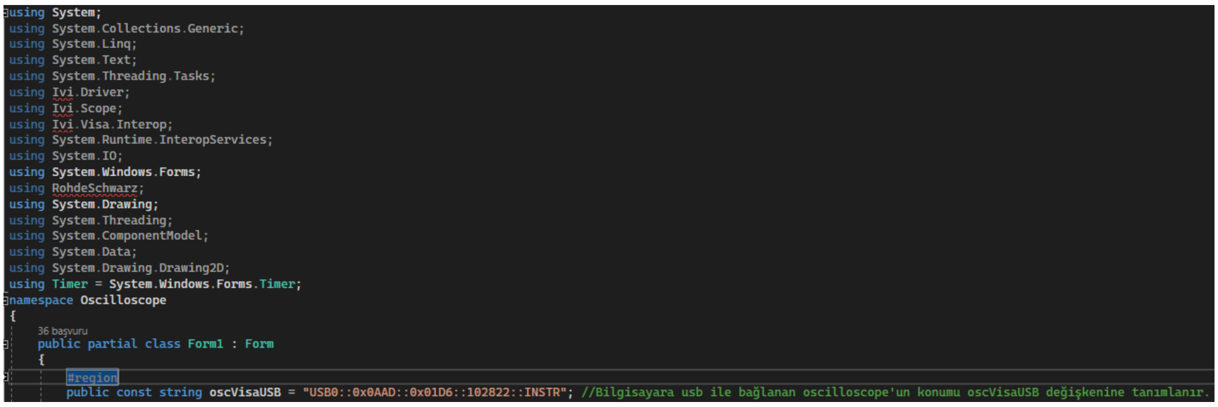


Figure 1 The Picture of Rohde&Schwarz RTA4004

I first started the project by reading the remote-control sections in the manual of the Rohde&Schwarz RTA4004 device. Since I had never remotely controlled any device before, I asked my internship advisor about the parts of the booklet that I did not understand.

First, I researched the methods of connecting the oscilloscope to the computer. I had two different connection options: wireless via ethernet or wired via USB. My internship advisor asked me to connect it to the computer with a USB cable, which is a wired connection method. Then, I installed the driver files on the website of the Rohde&Schwarz RTA4004 device to my computer so that the computer could recognize the oscilloscope. The codes I wrote to connect the oscilloscope and the computer in the Visual Studio 2022 program are shown in the image below.



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Ivi.Driver;
using Ivi.Scope;
using Ivi.Visa.Interop;
using System.Runtime.InteropServices;
using System.IO;
using System.Windows.Forms;
using RohdeSchwarz;
using System.Drawing;
using System.Threading;
using System.ComponentModel;
using System.Data;
using System.Drawing.Drawing2D;
using Timer = System.Windows.Forms.Timer;
namespace Oscilloscope
{
    36 başvuru
    public partial class Form1 : Form
    {
        #region
        public const string oscVisaUSB = "USB0::0x0AAD::0x01D6::102822::INSTR"; //Bilgisayara usb ile bağlanan oscilloscope'un konumu oscVisaUSB değişkenine tanımlanır.
```

Figure 2 The Picture of Oscilloscope GUI Project: Connection of the Oscilloscope

After the device was connected to my computer, I designed the Form 1 interface. The Form 1 interface I designed is shown in the image below.

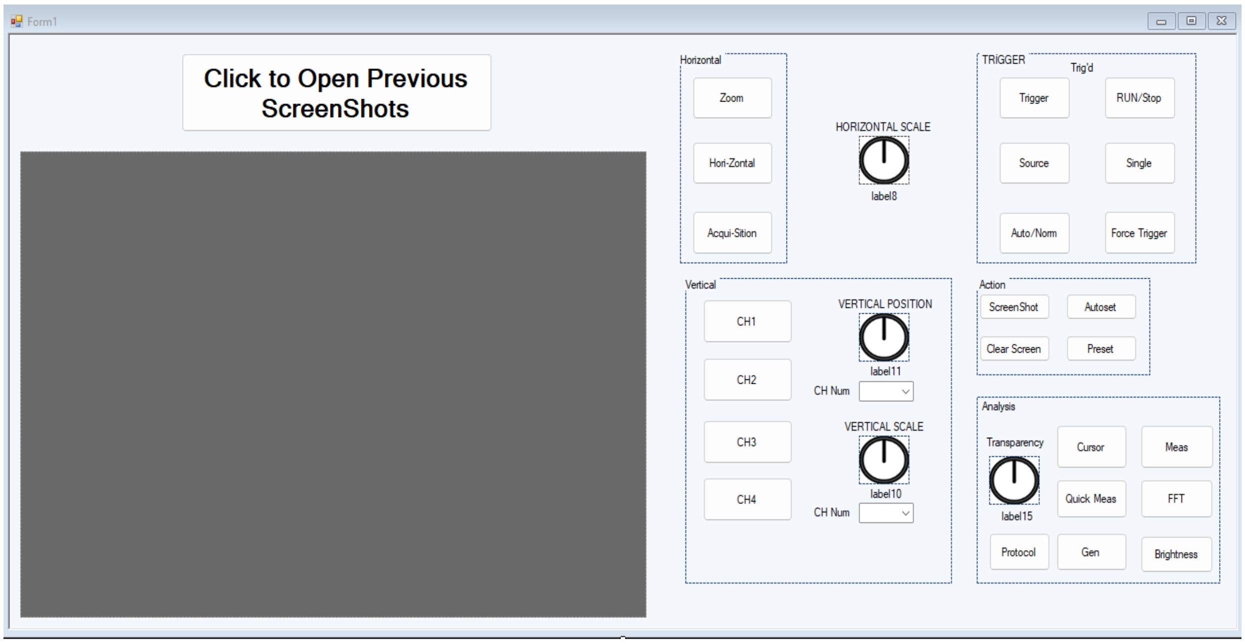


Figure 3 The Picture of Oscilloscope GUI Project: Form 1 Design

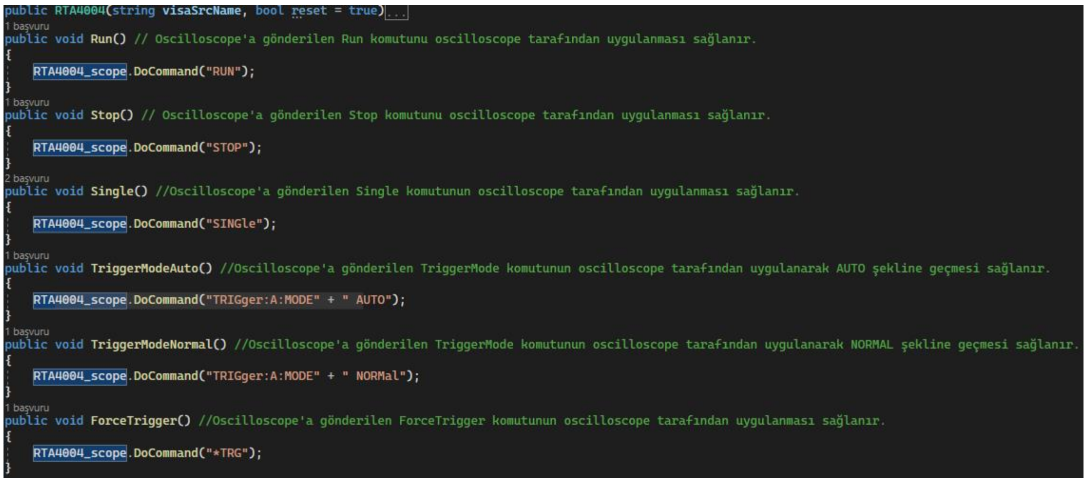
Since the Rohde&Schwarz RTA4004 device is a very complex device, my internship advisor asked me to add the features seen in ***Figure 3***. The main features of the buttons in the interface shown in ***Figure 3*** are given below.

- **Run/Stop:** It is the button that continues/stops the image on the oscilloscope.
- **Trigger:** The level at which the oscilloscope will start measuring the signal is set with this button.
- **Source:** This is the key that determines the source from which the oscilloscope will receive input.
- **Single:** This is the button used for single scanning.
- **Auto/Norm:** Auto or normal is selected for trigger.
- **ForceTrigger:** It is used to read the input at that moment without waiting for the trigger.
- **Screen Shot:** Instant screenshot is taken.
- **Autoset:** When Autoset is pressed, trigger settings are made automatically.
- **Clear Screen:** It ensures that the screen comes to its original state in confusion.
- **Preset:** Reset means that the features return to their original state.
- **Zoom:** It is used to zoom in or out.
- **Horizontal:** It enables the movement of the shape on the screen.
- **Acquisition:** It is decided how the oscilloscope will digitize the signal before displaying it.
- **CH1/CH2/CH3/CH4:** There are 4 different channel types. It is used for multiple analysis.
- **Cursor:** This is the part where graphic options are made.
- **Measure:** It makes calculations of the signals in the oscilloscope.
- **Quick Measure:** It allows quick calculation of current signals.

- **FFT:** The magnitude of the specified frequencies is displayed: power-frequency diagram.
- **Protocol:** It describes how a value is encoded into a waveform of a specific shape.
- **Generator:** Adjusts the signal generator settings of the oscilloscope.
- **Brightness:** This is the button used to adjust the brightness of the oscilloscope screen.

As you can see, since the device has a very complex structure, I wrote the codes in separate Classes in this project to make them appear more understandable. The name of this Class I wrote is "RTA4004". RTA 4004 Class includes codes written in C# to be understood by the oscilloscope. The display formats of the codes in RTA 4004 are written as in the Oscilloscope booklet. In this way, when the codes in other forms are sent to the RTA4004 Class, the relevant commands are made by the oscilloscope.

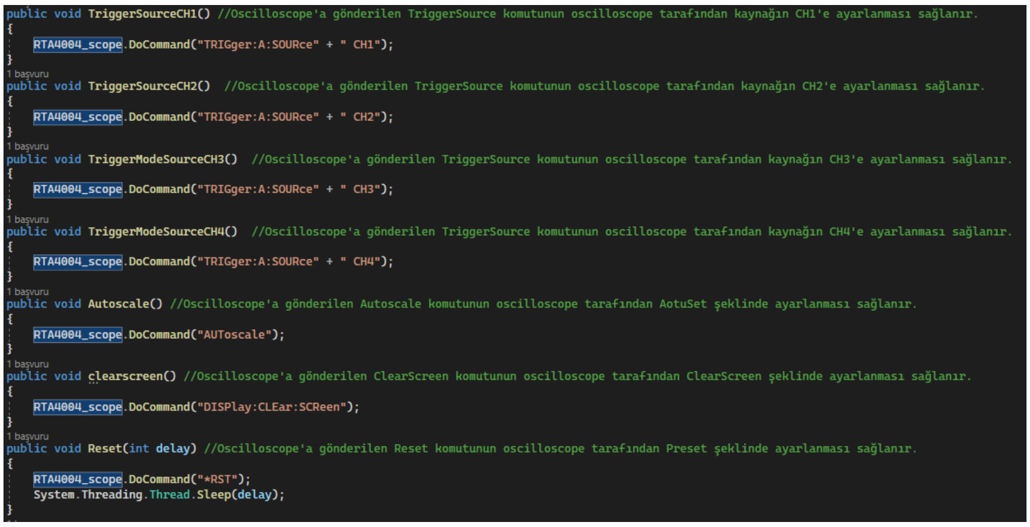
The codes written for the oscilloscope to understand the ***"RUN, STOP, SINGLE, AUTO, TRIGGER, FORCE TRIGGER"*** commands are given in the image below.



```
public RTA4004(string visaSrcName, bool reset = true)...
1 başvuru
public void Run() // Oscilloscope'a gönderilen Run komutunu oscilloscope tarafından uygulanması sağlanır.
{
    RTA4004_scope.DoCommand("RUN");
}
1 başvuru
public void Stop() // Oscilloscope'a gönderilen Stop komutunu oscilloscope tarafından uygulanması sağlanır.
{
    RTA4004_scope.DoCommand("STOP");
}
2 başvuru
public void Single() //Oscilloscope'a gönderilen Single komutunun oscilloscope tarafından uygulanması sağlanır.
{
    RTA4004_scope.DoCommand("SINGle");
}
1 başvuru
public void TriggerModeAuto() //Oscilloscope'a gönderilen TriggerMode komutunun oscilloscope tarafından uygulanarak AUTO şekline geçmesi sağlanır.
{
    RTA4004_scope.DoCommand("TRIGger:A:MODE" + " AUTO");
}
1 başvuru
public void TriggerModeNormal() //Oscilloscope'a gönderilen TriggerMode komutunun oscilloscope tarafından uygulanarak NORMAL şekline geçmesi sağlanır.
{
    RTA4004_scope.DoCommand("TRIGger:A:MODE" + " NORMal");
}
1 başvuru
public void ForceTrigger() //Oscilloscope'a gönderilen ForceTrigger komutunun oscilloscope tarafından uygulanması sağlanır.
{
    RTA4004_scope.DoCommand("*TRG");
}
```

Figure 4 The Picture of Oscilloscope GUI Project: RTA4004 Codes 1

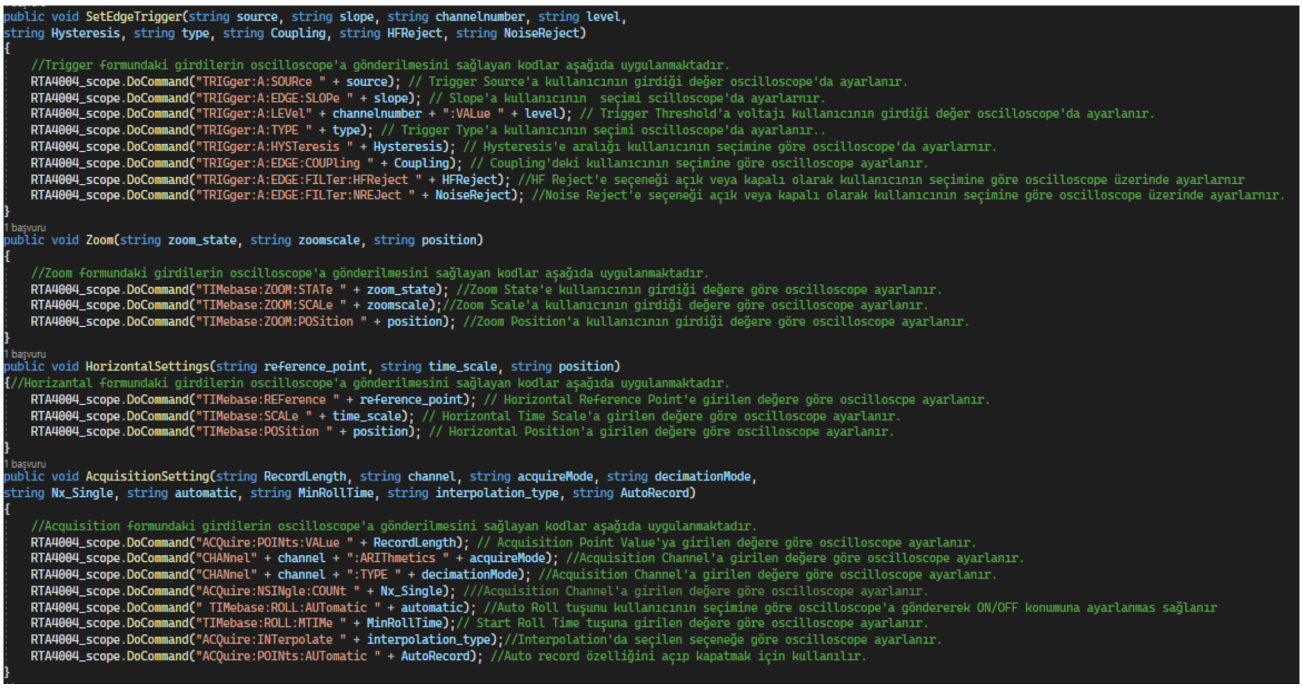
The codes written for the oscilloscope to understand the ***“CH1/CH2/CH3/CH4 SOURCE, AUTOSCALE, CLEARSCREEN, PRESET”*** commands are given in the image below.



```
public void TriggerSourceCH1() //Oscilloscope'a gönderilen TriggerSource komutunun oscilloscope tarafından kaynağın CH1'e ayarlanması sağlanır.
{
    RTA4004_scope.DoCommand("TRIGger:A:SOURce" + " CH1");
}
1 başvuru
public void TriggerSourceCH2()  //Oscilloscope'a gönderilen TriggerSource komutunun oscilloscope tarafından kaynağın CH2'e ayarlanması sağlanır.
{
    RTA4004_scope.DoCommand("TRIGger:A:SOURce" + " CH2");
}
1 başvuru
public void TriggerModeSourceCH3()  //Oscilloscope'a gönderilen TriggerSource komutunun oscilloscope tarafından kaynağın CH3'e ayarlanması sağlanır.
{
    RTA4004_scope.DoCommand("TRIGger:A:SOURce" + " CH3");
}
1 başvuru
public void TriggerModeSourceCH4()  //Oscilloscope'a gönderilen TriggerSource komutunun oscilloscope tarafından kaynağın CH4'e ayarlanması sağlanır.
{
    RTA4004_scope.DoCommand("TRIGger:A:SOURce" + " CH4");
}
1 başvuru
public void Autoscale() //Oscilloscope'a gönderilen Autoscale komutunun oscilloscope tarafından AotuSet şeklinde ayarlanması sağlanır.
{
    RTA4004_scope.DoCommand("AUToscale");
}
1 başvuru
public void clearscreen() //Oscilloscope'a gönderilen ClearScreen komutunun oscilloscope tarafından ClearScreen şeklinde ayarlanması sağlanır.
{
    RTA4004_scope.DoCommand("DISPlay:CLEar:SCReen");
}
1 başvuru
public void Reset(int delay) //Oscilloscope'a gönderilen Reset komutunun oscilloscope tarafından Preset şeklinde ayarlanması sağlanır.
{
    RTA4004_scope.DoCommand("*RST");
    System.Threading.Thread.Sleep(delay);
}
```

Figure 5 The Picture of Oscilloscope GUI Project: RTA4004 Codes 2

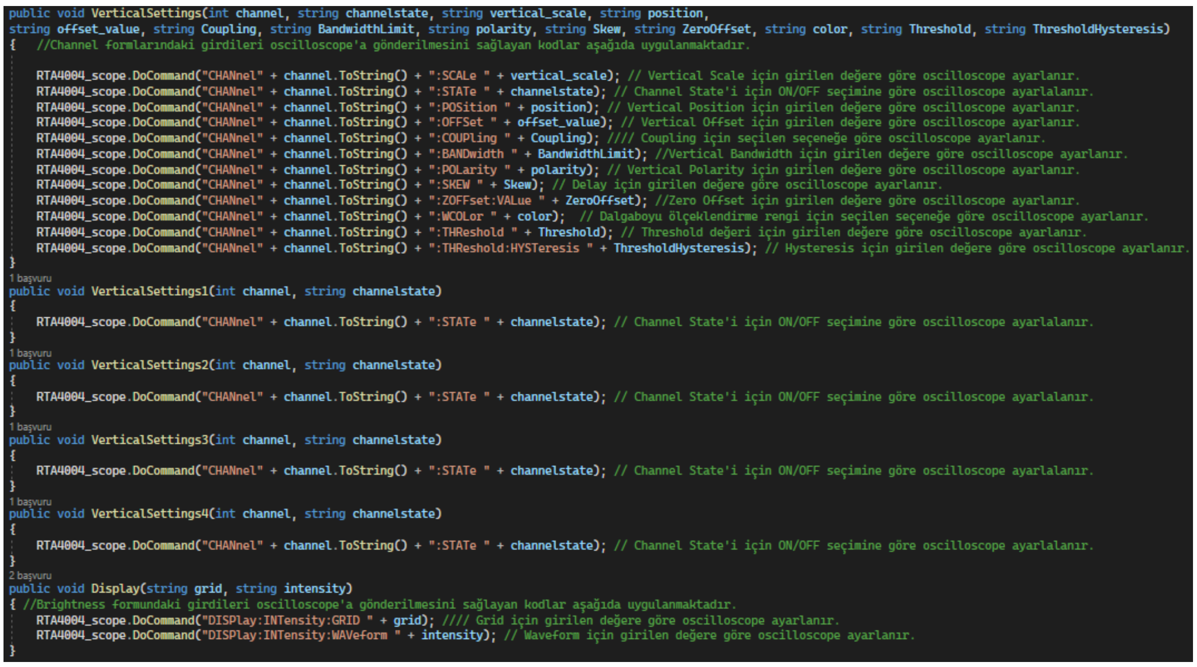
The codes written for the oscilloscope to understand the *"**TRIGGER, ZOOM, HORIZONTAL, ACQUISITION**"* commands are given in the image below.



```
public void SetEdgeTrigger(string source, string slope, string channelnumber, string level,
string Hysteresis, string type, string Coupling, string HFReject, string NoiseReject)
{
    //Trigger formundaki girdilerin oscilloscope'a gönderilmesini sağlayan kodlar aşağıda uygulanmaktadır.
    RTA4004_scope.DoCommand("TRIGger:A:SOURce " + source); // Trigger Source'a kullanıcının girdiği değer oscilloscope'da ayarlanır.
    RTA4004_scope.DoCommand("TRIGger:A:EDGE:SLOPe " + slope); // Slope'a kullanıcının  seçimi scilloscope'da ayarlarnır.
    RTA4004_scope.DoCommand("TRIGger:A:LEVel" + channelnumber + ":VALue " + level); // Trigger Threshold'a voltajı kullanıcının girdiği değer oscilloscope'da ayarlanır.
    RTA4004_scope.DoCommand("TRIGger:A:TYPE " + type); // Trigger Type'a kullanıcının seçimi oscilloscope'da ayarlanır..
    RTA4004_scope.DoCommand("TRIGger:A:HYSTeresis " + Hysteresis); // Hysteresis'e aralığı kullanıcının seçimine göre oscilloscope'da ayarlarnır.
    RTA4004_scope.DoCommand("TRIGger:A:EDGE:COUPling " + Coupling); // Coupling'deki kullanıcının seçimine göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("TRIGger:A:EDGE:FILTer:HFReject " + HFReject); //HF Reject'e seçeneği açık veya kapalı olarak kullanıcının seçimine göre oscilloscope üzerinde ayarlarnır
    RTA4004_scope.DoCommand("TRIGger:A:EDGE:FILTer:NREJect " + NoiseReject); //Noise Reject'e seçeneği açık veya kapalı olarak kullanıcının seçimine göre oscilloscope üzerinde ayarlarnır.
}
1 başvuru
public void Zoom(string zoom_state, string zoomscale, string position)
{
    //Zoom formundaki girdilerin oscilloscope'a gönderilmesini sağlayan kodlar aşağıda uygulanmaktadır.
    RTA4004_scope.DoCommand("TIMebase:ZOOM:STATe " + zoom_state); //Zoom State'e kullanıcının girdiği değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("TIMebase:ZOOM:SCALe " + zoomscale);//Zoom Scale'a kullanıcının girdiği değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("TIMebase:ZOOM:POSition " + position); //Zoom Position'a kullanıcının girdiği değere göre oscilloscope ayarlanır.
}
1 başvuru
public void HorizontalSettings(string reference_point, string time_scale, string position)
{//Horizantal formundaki girdilerin oscilloscope'a gönderilmesini sağlayan kodlar aşağıda uygulanmaktadır.
    RTA4004_scope.DoCommand("TIMebase:REFerence " + reference_point); // Horizontal Reference Point'e girilen değere göre oscilloscpe ayarlanır.
    RTA4004_scope.DoCommand("TIMebase:SCALe " + time_scale); // Horizontal Time Scale'a girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("TIMebase:POSition " + position); // Horizontal Position'a girilen değere göre oscilloscope ayarlanır.
}
1 başvuru
public void AcquisitionSetting(string RecordLength, string channel, string acquireMode, string decimationMode,
string Nx_Single, string automatic, string MinRollTime, string interpolation_type, string AutoRecord)
{
    //Acquisition formundaki girdilerin oscilloscope'a gönderilmesini sağlayan kodlar aşağıda uygulanmaktadır.
    RTA4004_scope.DoCommand("ACQuire:POINts:VALue " + RecordLength); // Acquisition Point Value'ya girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel + ":ARIThmetics " + acquireMode); //Acquisition Channel'a girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel + ":TYPE " + decimationMode); //Acquisition Channel'a girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("ACQuire:NSINgle:COUNt " + Nx_Single); ///Acquisition Channel'a girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand(" TIMebase:ROLL:AUTomatic " + automatic); //Auto Roll tuşunu kullanıcının seçimine göre oscilloscope'a göndererek ON/OFF konumuna ayarlanmas sağlanır
    RTA4004_scope.DoCommand("TIMebase:ROLL:MTIMe " + MinRollTime);// Start Roll Time tuşuna girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("ACQuire:INTerpolate " + interpolation_type);//Interpolation'da seçilen seçeneğe göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("ACQuire:POINts:AUTomatic " + AutoRecord); //Auto record özelliğini açıp kapatmak için kullanılır.
}
```

Figure 6 The Picture of Oscilloscope GUI Project: RTA4004 Codes 3

The codes written for the oscilloscope to understand the *"**VERTICAL, VERTICALSETTINGS CH1/CH2/CH3/CH4, DISPLAY**"* commands are given in the image below.

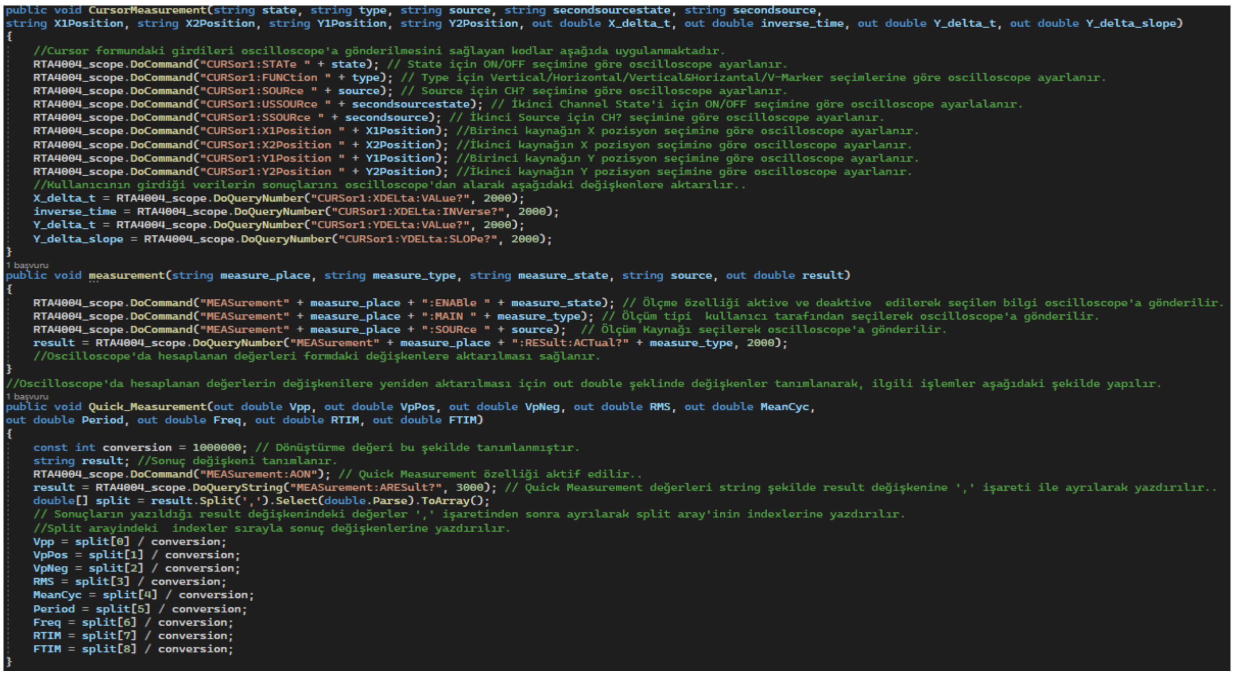


```
public void VerticalSettings(int channel, string channelstate, string vertical_scale, string position,
string offset_value, string Coupling, string BandwidthLimit, string polarity, string Skew, string ZeroOffset, string color, string Threshold, string ThresholdHysteresis)
{   //Channel formlarındaki girdileri oscilloscope'a gönderilmesini sağlayan kodlar aşağıda uygulanmaktadır.

    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":SCALe " + vertical_scale); // Vertical Scale için girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":STATe " + channelstate); // Channel State'i için ON/OFF seçimine göre oscilloscope ayarlalanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":POSition " + position); // Vertical Position için girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":OFFSet " + offset_value); // Vertical Offset için girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":COUPling " + Coupling); //// Coupling için seçilen seçeneğe göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":BANDwidth " + BandwidthLimit); //Vertical Bandwidth için girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":POLarity " + polarity); // Vertical Polarity için girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":SKEW " + Skew); // Delay için girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":ZOFFset:VALue " + ZeroOffset); //Zero Offset için girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":WCOLor " + color);  // Dalgaboyu ölçeklendirme rengi için seçilen seçeneğe göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":THReshold " + Threshold); // Threshold değeri için girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":THReshold:HYSTeresis " + ThresholdHysteresis); // Hysteresis için girilen değere göre oscilloscope ayarlanır.
}
1 başvuru
public void VerticalSettings1(int channel, string channelstate)
{
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":STATe " + channelstate); // Channel State'i için ON/OFF seçimine göre oscilloscope ayarlalanır.
}
1 başvuru
public void VerticalSettings2(int channel, string channelstate)
{
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":STATe " + channelstate); // Channel State'i için ON/OFF seçimine göre oscilloscope ayarlalanır.
}
1 başvuru
public void VerticalSettings3(int channel, string channelstate)
{
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":STATe " + channelstate); // Channel State'i için ON/OFF seçimine göre oscilloscope ayarlalanır.
}
1 başvuru
public void VerticalSettings4(int channel, string channelstate)
{
    RTA4004_scope.DoCommand("CHANnel" + channel.ToString() + ":STATe " + channelstate); // Channel State'i için ON/OFF seçimine göre oscilloscope ayarlalanır.
}
2 başvuru
public void Display(string grid, string intensity)
{ //Brightness formundaki girdileri oscilloscope'a gönderilmesini sağlayan kodlar aşağıda uygulanmaktadır.
    RTA4004_scope.DoCommand("DISPlay:INTensity:GRID " + grid); //// Grid için girilen değere göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("DISPlay:INTensity:WAVeform " + intensity); // Waveform için girilen değere göre oscilloscope ayarlanır.
}
```

Figure 7 The Picture of Oscilloscope GUI Project: RTA4004 Codes 4

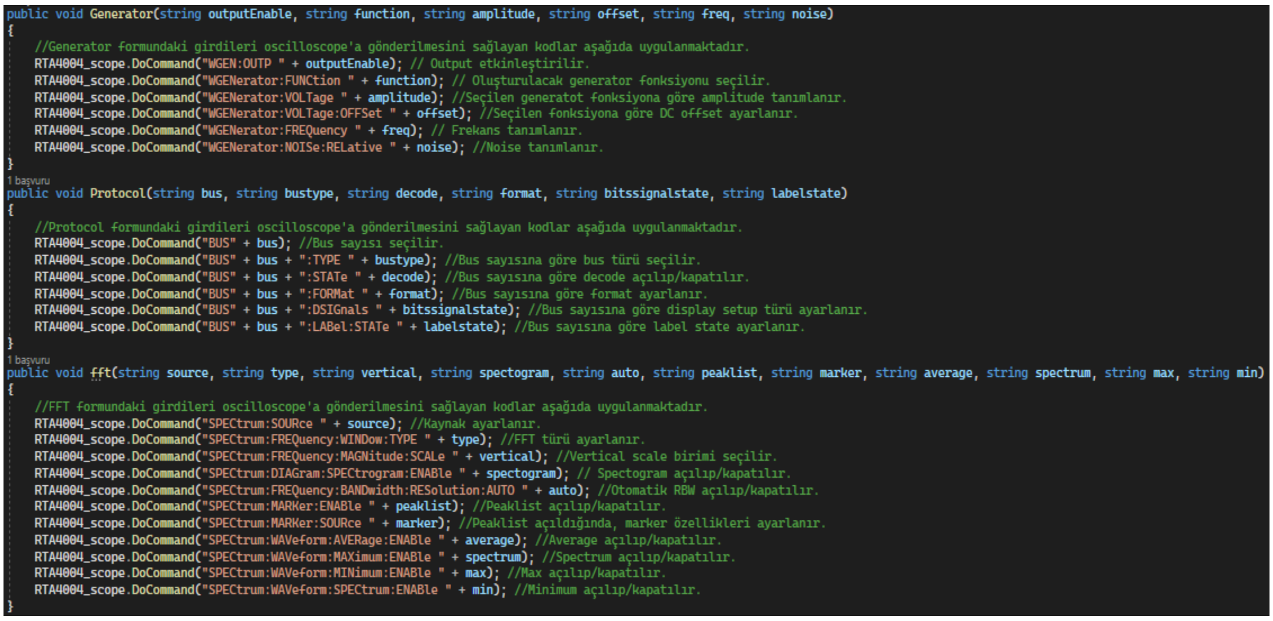
The codes written for the oscilloscope to understand the ***"CURSOR, MEASUREMENT, QUICKMEASUREMENT"*** commands are given in the image below.



```
public void CursorMeasurement(string state, string type, string source, string secondsourcestate, string secondsource,
string X1Position, string X2Position, string Y1Position, string Y2Position, out double X_delta_t, out double inverse_time, out double Y_delta_t, out double Y_delta_slope)
{
    //Cursor formundaki girdileri oscilloscope'a gönderilmesini sağlayan kodlar aşağıda uygulanmaktadır.
    RTA4004_scope.DoCommand("CURSor1:STATe " + state); // State için ON/OFF seçimine göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CURSor1:FUNCtion " + type); // Type için Vertical/Horizontal/Vertical&Horizantal/V-Marker seçimlerine göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CURSor1:SOURce " + source); // Source için CH? seçimine göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CURSor1:USSOURce " + secondsourcestate); // İkinci Channel State'i için ON/OFF seçimine göre oscilloscope ayarlalanır.
    RTA4004_scope.DoCommand("CURSor1:SSOURce " + secondsource); // İkinci Source için CH? seçimine göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CURSor1:X1Position " + X1Position); //Birinci kaynağın X pozisyon seçimine göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CURSor1:X2Position " + X2Position); //İkinci kaynağın X pozisyon seçimine göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CURSor1:Y1Position " + Y1Position); //Birinci kaynağın Y pozisyon seçimine göre oscilloscope ayarlanır.
    RTA4004_scope.DoCommand("CURSor1:Y2Position " + Y2Position); //İkinci kaynağın Y pozisyon seçimine göre oscilloscope ayarlanır.
    //Kullanıcının girdiği verilerin sonuçlarını oscilloscope'dan alarak aşağıdaki değişkenlere aktarılır..
    X_delta_t = RTA4004_scope.DoQueryNumber("CURSor1:XDELta:VALue?", 2000);
    inverse_time = RTA4004_scope.DoQueryNumber("CURSor1:XDELta:INVerse?", 2000);
    Y_delta_t = RTA4004_scope.DoQueryNumber("CURSor1:YDELta:VALue?", 2000);
    Y_delta_slope = RTA4004_scope.DoQueryNumber("CURSor1:YDELta:SLOPe?", 2000);
}
1 başvuru
public void measurement(string measure_place, string measure_type, string measure_state, string source, out double result)
{
    RTA4004_scope.DoCommand("MEASurement" + measure_place + ":ENABle " + measure_state); // Ölçme özelliği aktive ve deaktive  edilerek seçilen bilgi oscilloscope'a gönderilir.
    RTA4004_scope.DoCommand("MEASurement" + measure_place + ":MAIN " + measure_type); // Ölçüm tipi  kullanıcı tarafından seçilerek oscilloscope'a gönderilir.
    RTA4004_scope.DoCommand("MEASurement" + measure_place + ":SOURce " + source);  // Ölçüm Kaynağı seçilerek oscilloscope'a gönderilir.
    result = RTA4004_scope.DoQueryNumber("MEASurement" + measure_place + ":RESult:ACTual?" + measure_type, 2000);
    //Oscilloscope'da hesaplanan değerleri formdaki değişkenlere aktarılması sağlanır.
}
//Oscilloscope'da hesaplanan değerlerin değişkenilere yeniden aktarılması için out double şeklinde değişkenler tanımlanarak, ilgili işlemler aşağıdaki şekilde yapılır.
1 başvuru
public void Quick_Measurement(out double Vpp, out double VpPos, out double VpNeg, out double RMS, out double MeanCyc,
out double Period, out double Freq, out double RTIM, out double FTIM)
{
    const int conversion = 1000000; // Dönüştürme değeri bu şekilde tanımlanmıştır.
    string result; //Sonuç değişkeni tanımlanır.
    RTA4004_scope.DoCommand("MEASurement:AON"); // Quick Measurement özelliği aktif edilir..
    result = RTA4004_scope.DoQueryString("MEASurement:ARESult?", 3000); // Quick Measurement değerleri string şekilde result değişkenine ',' işareti ile ayrılarak yazdırılır..
    double[] split = result.Split(',').Select(double.Parse).ToArray();
    // Sonuçların yazıldığı result değişkenindeki değerler ',' işaretinden sonra ayrılarak split aray'inin indexlerine yazdırılır.
    //Split arayindeki  indexler sırayla sonuç değişkenlerine yazdırılır.
    Vpp = split[0] / conversion;
    VpPos = split[1] / conversion;
    VpNeg = split[2] / conversion;
    RMS = split[3] / conversion;
    MeanCyc = split[4] / conversion;
    Period = split[5] / conversion;
    Freq = split[6] / conversion;
    RTIM = split[7] / conversion;
    FTIM = split[8] / conversion;
}
```

Figure 8 The Picture of Oscilloscope GUI Project: RTA4004 Codes 5

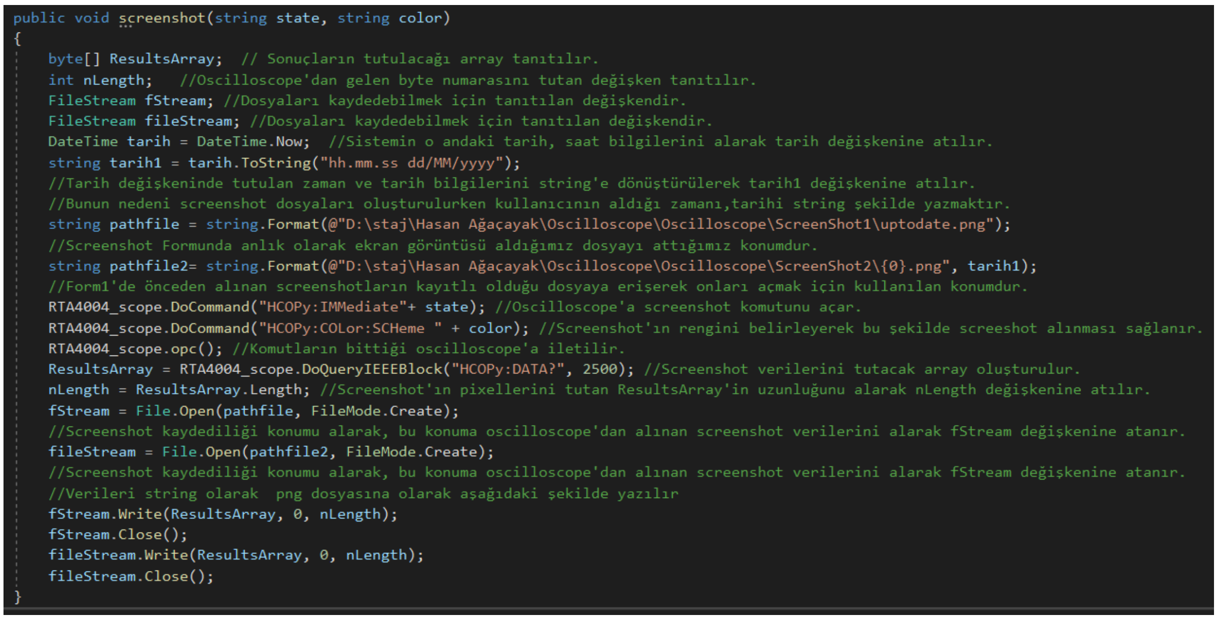
The codes written for the oscilloscope to understand the ***"GENERATOR, PROTOCOL, FFT"*** commands are given in the image below.



```
public void Generator(string outputEnable, string function, string amplitude, string offset, string freq, string noise)
{
    //Generator formundaki girdileri oscilloscope'a gönderilmesini sağlayan kodlar aşağıda uygulanmaktadır.
    RTA4004_scope.DoCommand("WGEN:OUTP " + outputEnable); // Output etkinleştirilir.
    RTA4004_scope.DoCommand("WGENerator:FUNCtion " + function); // Oluşturulacak generator fonksiyonu seçilir.
    RTA4004_scope.DoCommand("WGENerator:VOLTage " + amplitude); //Seçilen generatot fonksiyona göre amplitude tanımlanır.
    RTA4004_scope.DoCommand("WGENerator:VOLTage:OFFSet " + offset); //Seçilen fonksiyona göre DC offset ayarlanır.
    RTA4004_scope.DoCommand("WGENerator:FREQuency " + freq); // Frekans tanımlanır.
    RTA4004_scope.DoCommand("WGENerator:NOISe:RELative " + noise); //Noise tanımlanır.
}
1 başvuru
public void Protocol(string bus, string bustype, string decode, string format, string bitssignalstate, string labelstate)
{
    //Protocol formundaki girdileri oscilloscope'a gönderilmesini sağlayan kodlar aşağıda uygulanmaktadır.
    RTA4004_scope.DoCommand("BUS" + bus); //Bus sayısı seçilir.
    RTA4004_scope.DoCommand("BUS" + bus + ":TYPE " + bustype); //Bus sayısına göre bus türü seçilir.
    RTA4004_scope.DoCommand("BUS" + bus + ":STATe " + decode); //Bus sayısına göre decode açılıp/kapatılır.
    RTA4004_scope.DoCommand("BUS" + bus + ":FORMat " + format); //Bus sayısına göre format ayarlanır.
    RTA4004_scope.DoCommand("BUS" + bus + ":DSIGnals " + bitssignalstate); //Bus sayısına göre display setup türü ayarlanır.
    RTA4004_scope.DoCommand("BUS" + bus + ":LABel:STATe " + labelstate); //Bus sayısına göre label state ayarlanır.
}
1 başvuru
public void fft(string source, string type, string vertical, string spectogram, string auto, string peaklist, string marker, string average, string spectrum, string max, string min)
{
    //FFT formundaki girdileri oscilloscope'a gönderilmesini sağlayan kodlar aşağıda uygulanmaktadır.
    RTA4004_scope.DoCommand("SPECtrum:SOURce " + source); //Kaynak ayarlanır.
    RTA4004_scope.DoCommand("SPECtrum:FREQuency:WINDow:TYPE " + type); //FFT türü ayarlanır.
    RTA4004_scope.DoCommand("SPECtrum:FREQuency:MAGNitude:SCALe " + vertical); //Vertical scale birimi seçilir.
    RTA4004_scope.DoCommand("SPECtrum:DIAGram:SPECtrogram:ENABle " + spectogram); // Spectogram açılıp/kapatılır.
    RTA4004_scope.DoCommand("SPECtrum:FREQuency:BANDwidth:RESolution:AUTO " + auto); //Otomatik RBW açılıp/kapatılır.
    RTA4004_scope.DoCommand("SPECtrum:MARKer:ENABle " + peaklist); //Peaklist açılıp/kapatılır.
    RTA4004_scope.DoCommand("SPECtrum:MARKer:SOURce " + marker); //Peaklist açıldığında, marker özellikleri ayarlanır.
    RTA4004_scope.DoCommand("SPECtrum:WAVeform:AVERage:ENABle " + average); //Average açılıp/kapatılır.
    RTA4004_scope.DoCommand("SPECtrum:WAVeform:MAXimum:ENABle " + spectrum); //Spectrum açılıp/kapatılır.
    RTA4004_scope.DoCommand("SPECtrum:WAVeform:MINimum:ENABle " + max); //Max açılıp/kapatılır.
    RTA4004_scope.DoCommand("SPECtrum:WAVeform:SPECtrum:ENABle " + min); //Minimum açılıp/kapatılır.
}
```

Figure 9 The Picture of Oscilloscope GUI Project: RTA4004 Codes 6

The codes written for the oscilloscope to understand the ***“SCREENSHOT”*** command are given in the image below.



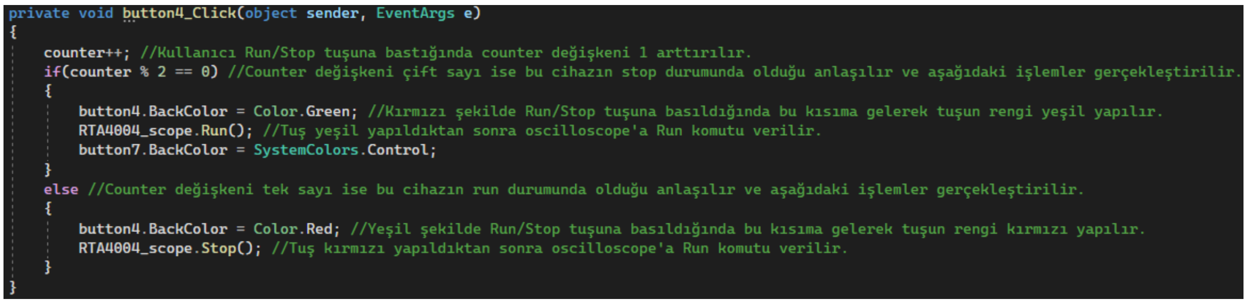
```
public void screenshot(string state, string color)
{
    byte[] ResultsArray;  // Sonuçların tutulacağı array tanıtılır.
    int nLength;   //Oscilloscope'dan gelen byte numarasını tutan değişken tanıtılır.
    FileStream fStream; //Dosyaları kaydedebilmek için tanıtılan değişkendir.
    FileStream fileStream; //Dosyaları kaydedebilmek için tanıtılan değişkendir.
    DateTime tarih = DateTime.Now;  //Sistemin o andaki tarih, saat bilgilerini alarak tarih değişkenine atılır.
    string tarih1 = tarih.ToString("hh.mm.ss dd/MM/yyyy");
    //Tarih değişkeninde tutulan zaman ve tarih bilgilerini string'e dönüştürülerek tarih1 değişkenine atılır.
    //Bunun nedeni screenshot dosyaları oluşturulurken kullanıcının aldığı zamanı,tarihi string şekilde yazmaktır.
    string pathfile = string.Format(@"D:\staj\Hasan Ağaçayak\Oscilloscope\Oscilloscope\ScreenShot1\uptodate.png");
    //Screenshot Formunda anlık olarak ekran görüntüsü aldığımız dosyayı attığımız konumdur.
    string pathfile2= string.Format(@"D:\staj\Hasan Ağaçayak\Oscilloscope\Oscilloscope\ScreenShot2\{0}.png", tarih1);
    //Form1'de önceden alınan screenshotların kayıtlı olduğu dosyaya erişerek onları açmak için kullanılan konumdur.
    RTA4004_scope.DoCommand("HCOPy:IMMediate"+ state); //Oscilloscope'a screenshot komutunu açar.
    RTA4004_scope.DoCommand("HCOPy:COLor:SCHeme " + color); //Screenshot'ın rengini belirleyerek bu şekilde screeshot alınması sağlanır.
    RTA4004_scope.opc(); //Komutların bittiği oscilloscope'a iletilir.
    ResultsArray = RTA4004_scope.DoQueryIEEEBlock("HCOPy:DATA?", 2500); //Screenshot verilerini tutacak array oluşturulur.
    nLength = ResultsArray.Length; //Screenshot'ın pixellerini tutan ResultsArray'in uzunluğunu alarak nLength değişkenine atılır.
    fStream = File.Open(pathfile, FileMode.Create);
    //Screenshot kaydediliği konumu alarak, bu konuma oscilloscope'dan alınan screenshot verilerini alarak fStream değişkenine atanır.
    fileStream = File.Open(pathfile2, FileMode.Create);
    //Screenshot kaydediliği konumu alarak, bu konuma oscilloscope'dan alınan screenshot verilerini alarak fStream değişkenine atanır.
    //Verileri string olarak  png dosyasına olarak aşağıdaki şekilde yazılır
    fStream.Write(ResultsArray, 0, nLength);
    fStream.Close();
    fileStream.Write(ResultsArray, 0, nLength);
    fileStream.Close();
}
```

Figure 10 The Picture of Oscilloscope GUI Project: RTA4004 Codes 7

The codes of the RTA4004 Class given in the images above are the codes that provide the connection between the oscilloscope and the computer. The codes given after this section are codes given for calculations, design of forms and similar features in C#.

When I look at the image in ***Figure 3***, the codes of the keys that do not open a new form are written in the Form1.cs file. These keys are mainly *"**RUN/STOP, Single, AUTO/NORM, SOURCE, AUTOSET/PRESET/CLEARSCREEN, Click to Open Previous ScreenShots**"*. The images of these buttons are shown below.

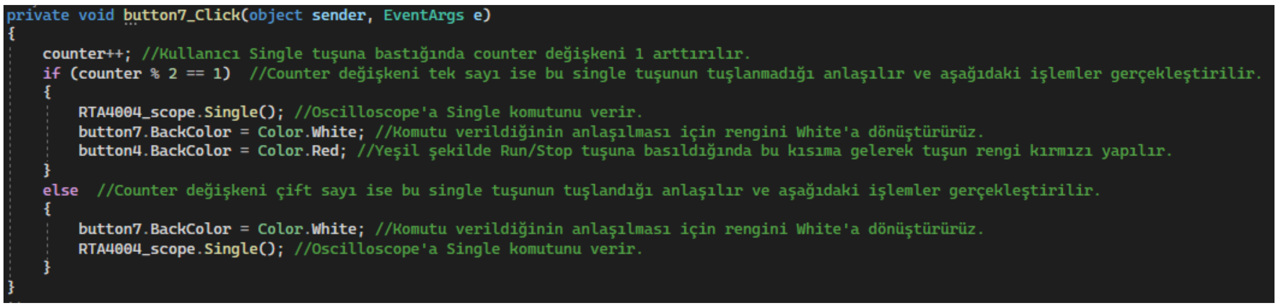
Images of the codes of the ***“RUN/STOP”*** button are given below.



```
private void button4_Click(object sender, EventArgs e)
{
    counter++; //Kullanıcı Run/Stop tuşuna bastığında counter değişkeni 1 arttırılır.
    if(counter % 2 == 0) //Counter değişkeni çift sayı ise bu cihazın stop durumunda olduğu anlaşılır ve aşağıdaki işlemler gerçekleştirilir.
    {
        button4.BackColor = Color.Green; //Kırmızı şekilde Run/Stop tuşuna basıldığında bu kısıma gelerek tuşun rengi yeşil yapılır.
        RTA4004_scope.Run(); //Tuş yeşil yapıldıktan sonra oscilloscope'a Run komutu verilir.
        button7.BackColor = SystemColors.Control;
    }
    else //Counter değişkeni tek sayı ise bu cihazın run durumunda olduğu anlaşılır ve aşağıdaki işlemler gerçekleştirilir.
    {
        button4.BackColor = Color.Red; //Yeşil şekilde Run/Stop tuşuna basıldığında bu kısıma gelerek tuşun rengi kırmızı yapılır.
        RTA4004_scope.Stop(); //Tuş kırmızı yapıldıktan sonra oscilloscope'a Run komutu verilir.
    }
}
```

Figure 11 The Picture of Oscilloscope GUI Project: Form 1 RUN/STOP Codes

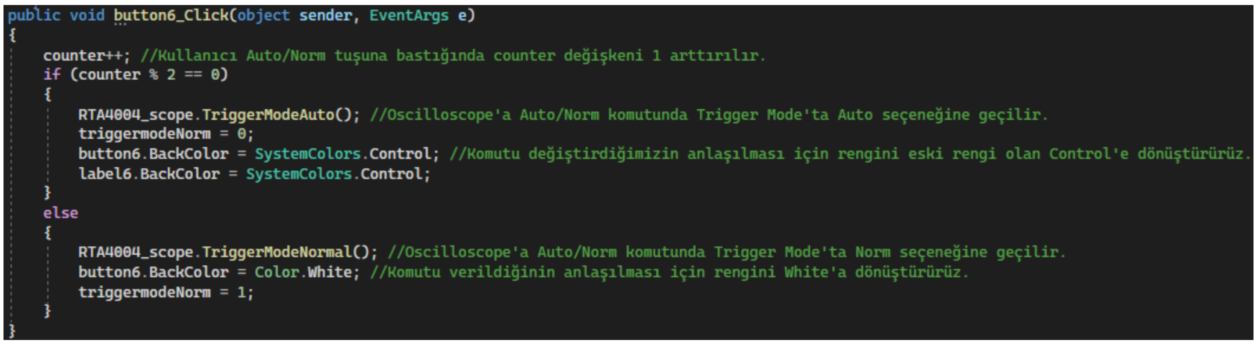
Images of the codes of the ***"SINGLE"*** button are given below.



```
private void button7_Click(object sender, EventArgs e)
{
    counter++; //Kullanıcı Single tuşuna bastığında counter değişkeni 1 arttırılır.
    if (counter % 2 == 1)  //Counter değişkeni tek sayı ise bu single tuşunun tuşlanmadığı anlaşılır ve aşağıdaki işlemler gerçekleştirilir.
    {
        RTA4004_scope.Single(); //Oscilloscope'a Single komutunu verir.
        button7.BackColor = Color.White; //Komutu verildiğinin anlaşılması için rengini White'a dönüştürürüz.
        button4.BackColor = Color.Red; //Yeşil şekilde Run/Stop tuşuna basıldığında bu kısıma gelerek tuşun rengi kırmızı yapılır.
    }
    else  //Counter değişkeni çift sayı ise bu single tuşunun tuşlandığı anlaşılır ve aşağıdaki işlemler gerçekleştirilir.
    {
        button7.BackColor = Color.White; //Komutu verildiğinin anlaşılması için rengini White'a dönüştürürüz.
        RTA4004_scope.Single(); //Oscilloscope'a Single komutunu verir.
    }
}
```

Figure 12 The Picture of Oscilloscope GUI Project: Form 1 SINGLE Codes

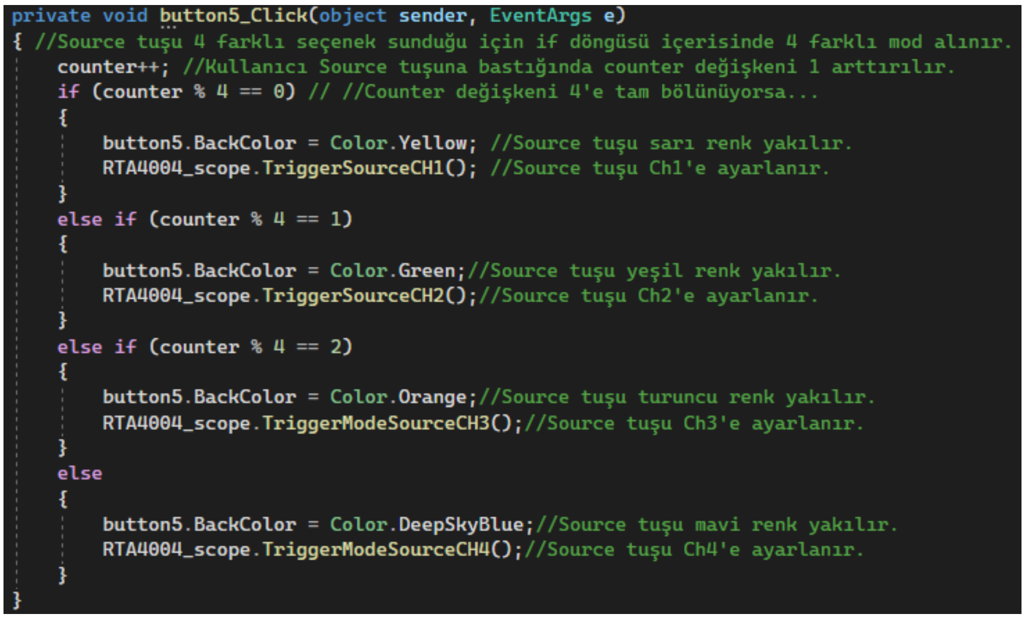
Images of the codes of the ***"AUTO/NORM"*** button are given below.



```
public void button6_Click(object sender, EventArgs e)
{
    counter++; //Kullanıcı Auto/Norm tuşuna bastığında counter değişkeni 1 arttırılır.
    if (counter % 2 == 0)
    {
        RTA4004_scope.TriggerModeAuto(); //Oscilloscope'a Auto/Norm komutunda Trigger Mode'ta Auto seçeneğine geçilir.
        triggermodeNorm = 0;
        button6.BackColor = SystemColors.Control; //Komutu değiştirdiğimizin anlaşılması için rengini eski rengi olan Control'e dönüştürürüz.
        label6.BackColor = SystemColors.Control;
    }
    else
    {
        RTA4004_scope.TriggerModeNormal(); //Oscilloscope'a Auto/Norm komutunda Trigger Mode'ta Norm seçeneğine geçilir.
        button6.BackColor = Color.White; //Komutu verildiğinin anlaşılması için rengini White'a dönüştürürüz.
        triggermodeNorm = 1;
    }
}
```

Figure 13 The Picture of Oscilloscope GUI Project: Form 1 AUTO/NORM Codes

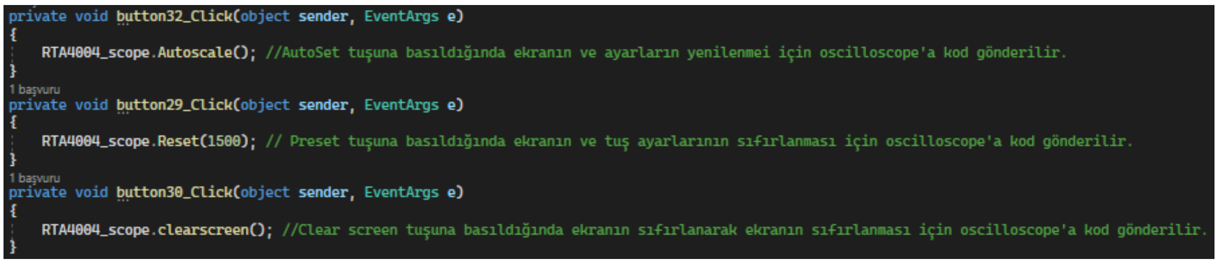
Images of the codes of the ***"*****SOURCE*****"*** button are given below.



```
private void button5_Click(object sender, EventArgs e)
{ //Source tuşu 4 farklı seçenek sunduğu için if döngüsü içerisinde 4 farklı mod alınır.
    counter++; //Kullanıcı Source tuşuna bastığında counter değişkeni 1 arttırılır.
    if (counter % 4 == 0) // //Counter değişkeni 4'e tam bölünüyorsa...
    {
        button5.BackColor = Color.Yellow; //Source tuşu sarı renk yakılır.
        RTA4004_scope.TriggerSourceCH1(); //Source tuşu Ch1'e ayarlanır.
    }
    else if (counter % 4 == 1)
    {
        button5.BackColor = Color.Green;//Source tuşu yeşil renk yakılır.
        RTA4004_scope.TriggerSourceCH2();//Source tuşu Ch2'e ayarlanır.
    }
    else if (counter % 4 == 2)
    {
        button5.BackColor = Color.Orange;//Source tuşu turuncu renk yakılır.
        RTA4004_scope.TriggerModeSourceCH3();//Source tuşu Ch3'e ayarlanır.
    }
    else
    {
        button5.BackColor = Color.DeepSkyBlue;//Source tuşu mavi renk yakılır.
        RTA4004_scope.TriggerModeSourceCH4();//Source tuşu Ch4'e ayarlanır.
    }
}
```

Figure 14 The Picture of Oscilloscope GUI Project: Form 1 SOURCE Codes

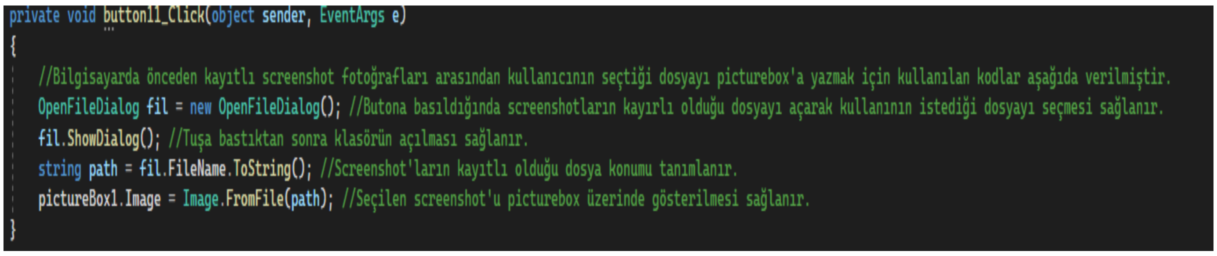
Images of the codes of the ***“AUTOSET/PRESET/CLEARSCREEN”*** button are given below.



```
private void button32_Click(object sender, EventArgs e)
{
    RTA4004_scope.Autoscale(); //AutoSet tuşuna basıldığında ekranın ve ayarların yenilenmei için oscilloscope'a kod gönderilir.
}
1 başvuru
private void button29_Click(object sender, EventArgs e)
{
    RTA4004_scope.Reset(1500); // Preset tuşuna basıldığında ekranın ve tuş ayarlarının sıfırlanması için oscilloscope'a kod gönderilir.
}
1 başvuru
private void button30_Click(object sender, EventArgs e)
{
    RTA4004_scope.clearscreen(); //Clear screen tuşuna basıldığında ekranın sıfırlanarak ekranın sıfırlanması için oscilloscope'a kod gönderilir.
}
```

Figure 15 The Picture of Oscilloscope GUI Project: Form 1 AUTOSET/PRESET/CLEARSCREEN Codes

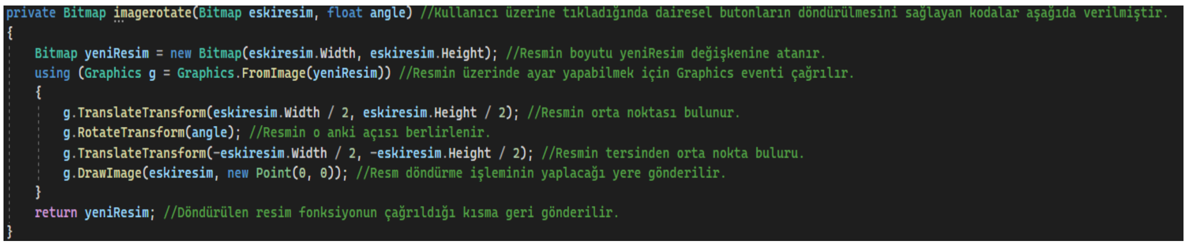
The “Click to Open Previous ScreenShots” button allows previously saved Screenshot images to be selected by the user and displayed on the Picturebox in the Form 1 interface. Images of the codes of the ***“Click to Open Previous ScreenShots”*** button are given below.



```
private void button11_Click(object sender, EventArgs e)
{
    //Bilgisayarda önceden kayıtlı screenshot fotoğrafları arasından kullanıcının seçtiği dosyayı picturebox'a yazmak için kullanılan kodlar aşağıda verilmiştir.
    OpenFileDialog fil = new OpenFileDialog(); //Butona basıldığında screenshotların kayırlı olduğu dosyayı açarak kullanının istediği dosyayı seçmesi sağlanır.
    fil.ShowDialog(); //Tuşa bastıktan sonra klasörün açılması sağlanır.
    string path = fil.FileName.ToString(); //Screenshot'ların kayıtlı olduğu dosya konumu tanımlanır.
    pictureBox1.Image = Image.FromFile(path); //Seçilen screenshot'u picturebox üzerinde gösterilmesi sağlanır.
}
```

Figure 16 The Picture of Oscilloscope GUI Project: Form Click to Open Previous ScreenShots Button Codes

The codes that enable the user to rotate the ***"Horizontal Scale, Vertical Position, Vertical Scale, Transparency"*** buttons in ***Form 1*** with the mouse are given below.



```
private Bitmap imagerotate(Bitmap eskiresim, float angle) //Kullanıcı üzerine tıkladığında dairesel butonların döndürülmesini sağlayan kodalar aşağıda verilmiştir.
{
    Bitmap yeniResim = new Bitmap(eskiresim.Width, eskiresim.Height); //Resmin boyutu yeniResim değişkenine atanır.
    using (Graphics g = Graphics.FromImage(yeniResim)) //Resmin üzerinde ayar yapabilmek için Graphics eventi çağrılır.
    {
        g.TranslateTransform(eskiresim.Width / 2, eskiresim.Height / 2); //Resmin orta noktası bulunur.
        g.RotateTransform(angle); //Resmin o anki açısı berlirlenir.
        g.TranslateTransform(-eskiresim.Width / 2, -eskiresim.Height / 2); //Resmin tersinden orta nokta buluru.
        g.DrawImage(eskiresim, new Point(0, 0)); //Resm döndürme işleminin yaplacağı yere gönderilir.
    }
    return yeniResim; //Döndürülen resim fonksiyonun çağrıldığı kısma geri gönderilir.
}
```

Figure 17 The Picture of Oscilloscope GUI Project: Form 1 Rotating Images for Control Buttons Codes

The Acqui-Sition Form that appears before the user when ***the Acqui-Sition*** button in the image in ***Figure 3*** is pressed is given in the image below.

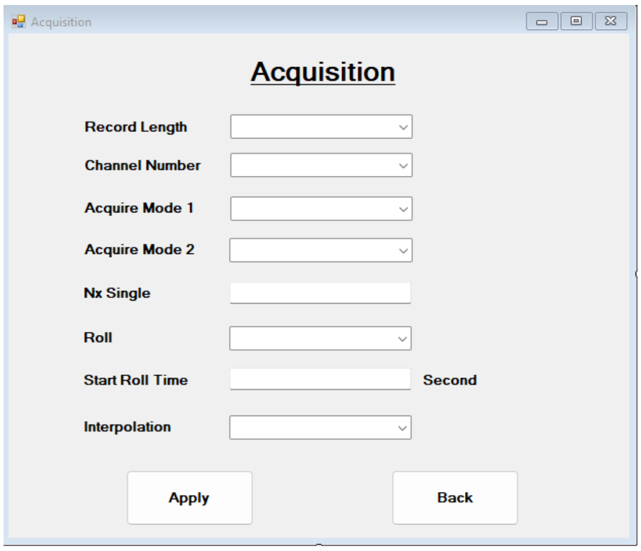
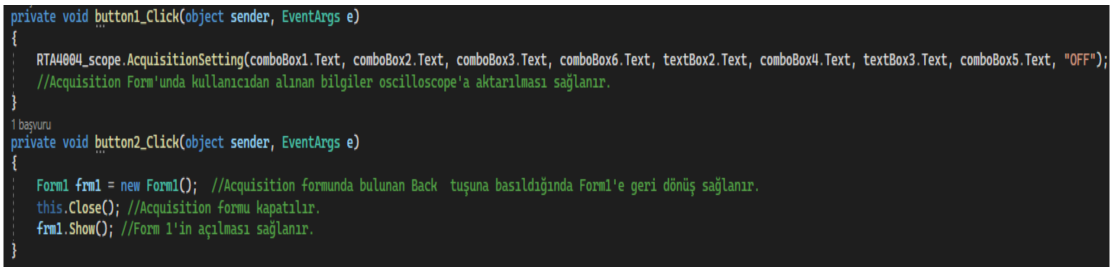


Figure 18 The Picture of Oscilloscope GUI Project: Acqui-Sition Form Design

"Record Length, Channel Number, Acquire Mode 1, Acquire Mode 2, Nx Single, Roll, Start Roll Time, Interpolation" values are set by the user using ***the Acqui-Sition Form*** seen in ***Figure 18***. The values requested by the user are transmitted to the Oscilloscope through ***the AcquisitionSetting*** function in the RTA 4004 Class shown in ***Figure 6***. The codes for ***the Acqui-Sition*** Form are given in the image below.



```
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.AcquisitionSetting(comboBox1.Text, comboBox2.Text, comboBox3.Text, comboBox6.Text, textBox2.Text, comboBox4.Text, textBox3.Text, comboBox5.Text, "OFF");
    //Acquisition Form'unda kullanıcıdan alınan bilgiler oscilloscope'a aktarılması sağlanır.
}
1 başvuru
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Acquisition formunda bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Acquisition formu kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
```

Figure 19 The Picture of Oscilloscope GUI Project: Acqui-Sition Form Codes

The Brightness Form that appears before the user when ***the Brightness*** button in the image in ***Figure 3*** is pressed is given in the image below.

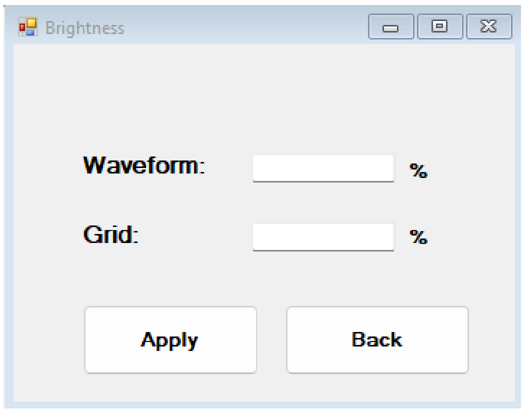
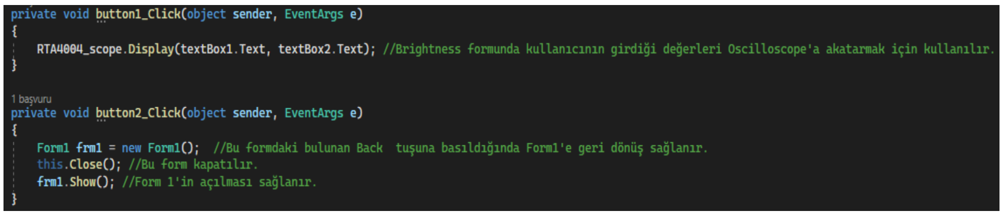


Figure 20 The Picture of Oscilloscope GUI Project: Brightness Form Design

"Waveform, Grid" values are adjusted by the user using ***the Brightness Form*** seen in ***Figure 20***. The values requested by the user are transmitted to the Oscilloscope through the ***Display*** function in the RTA 4004 Class shown in ***Figure 7***. The codes for ***the Brightness Form*** are given in the image below.



```
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.Display(textBox1.Text, textBox2.Text); //Brightness formunda kullanıcının girdiği değerleri Oscilloscope'a akatarmak için kullanılır.
}

1 başvuru
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
```

Figure 21The Picture of Oscilloscope GUI Project: Brightness Form Codes

The CH1 Form that appears before the user when ***the CH1*** button in the image in ***Figure 3*** is pressed is given in the image below.

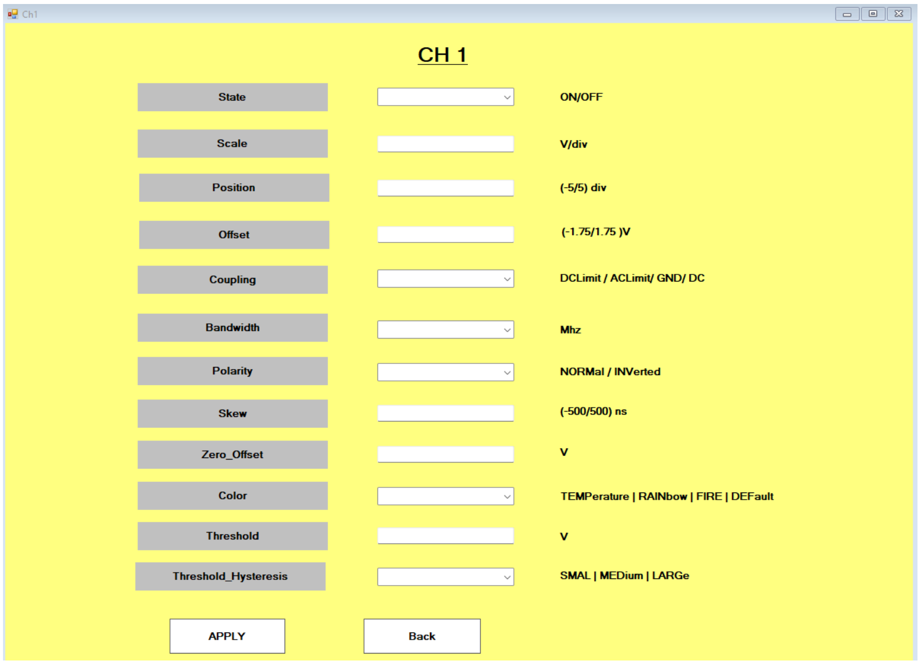
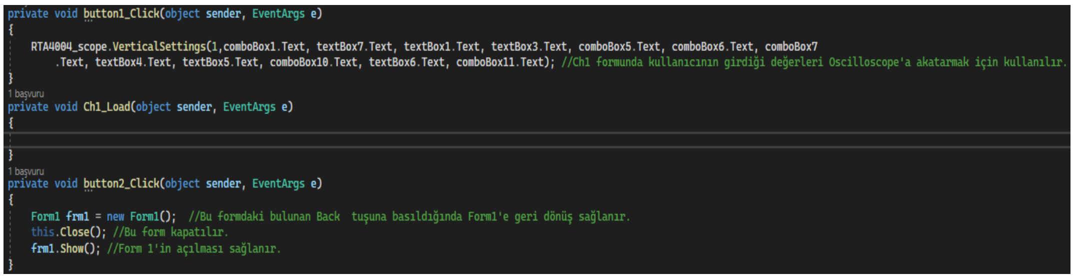


Figure 22 The Picture of Oscilloscope GUI Project: CH1 Form Design

"State, Scale, Position, Offset, Coupling, Bandwidth, Polarity, Skey, Zero_Offset, Color, Threshold, Threshold_Hysteresis" values are set by the user using ***the Ch1 Form*** seen in ***Figure 22***. The values requested by the user are transmitted to the oscilloscope through the ***VerticalSettings*** function in the RTA 4004 Class shown in ***Figure 7***. The codes for ***Ch1 Form*** are given in the image below.



```
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.VerticalSettings(1,comboBox1.Text, textBox7.Text, textBox1.Text, textBox3.Text, comboBox5.Text, comboBox6.Text, comboBox7
        .Text, textBox4.Text, textBox5.Text, comboBox10.Text, textBox6.Text, comboBox11.Text); //Ch1 formunda kullanıcının girdiği değerleri Oscilloscope'a akatarmak için kullanılır.
}
1 başvuru
private void Ch1_Load(object sender, EventArgs e)
{

}
1 başvuru
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
```

Figure 23 The Picture of Oscilloscope GUI Project: Ch1 Form Codes

The CH2 Form that appears before the user when ***the CH2*** button in the image in ***Figure 3*** is pressed is given in the image below.

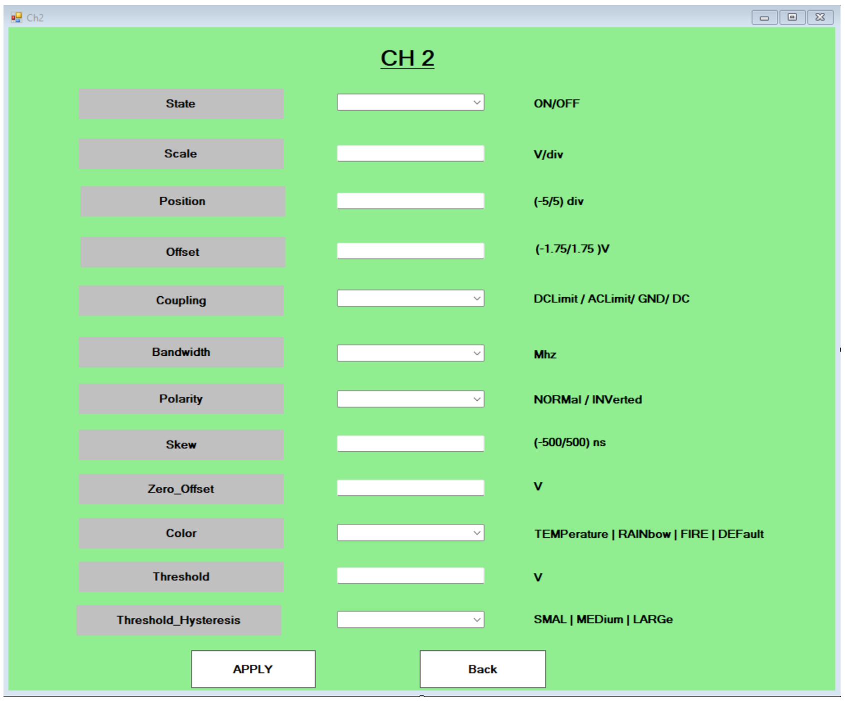
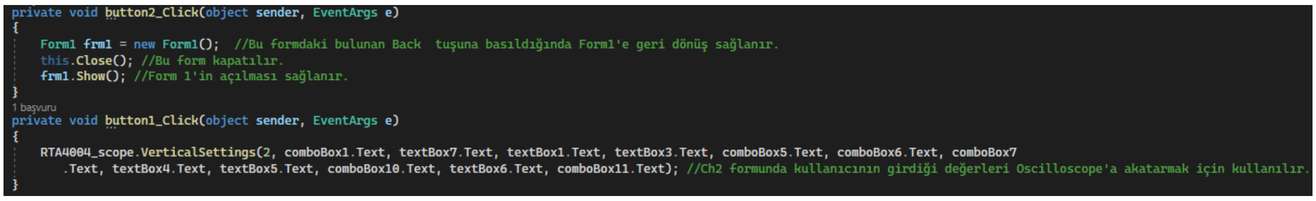


Figure 24 The Picture of Oscilloscope GUI Project: CH2 Form Design

"State, Scale, Position, Offset, Coupling, Bandwidth, Polarity, Skey, Zero_Offset, Color, Threshold, Threshold_Hysteresis" values are set by the user using ***the Ch2 Form*** seen in ***Figure 24***. The values requested by the user are transmitted to the oscilloscope through the ***VerticalSettings*** function in the RTA 4004 Class shown in ***Figure 7***. The codes for ***the Ch2 Form*** are given in the image below.



```
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
1 başvuru
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.VerticalSettings(2, comboBox1.Text, textBox7.Text, textBox1.Text, textBox3.Text, comboBox5.Text, comboBox6.Text, comboBox7
       .Text, textBox4.Text, textBox5.Text, comboBox10.Text, textBox6.Text, comboBox11.Text); //Ch2 formunda kullanıcının girdiği değerleri Oscilloscope'a akatarmak için kullanılır.
}
```

Figure 25 The Picture of Oscilloscope GUI Project: Ch2 Form Codes

The CH3 Form that appears before the user when ***the CH3*** key in the image in ***Figure 3*** is pressed is given in the image below.

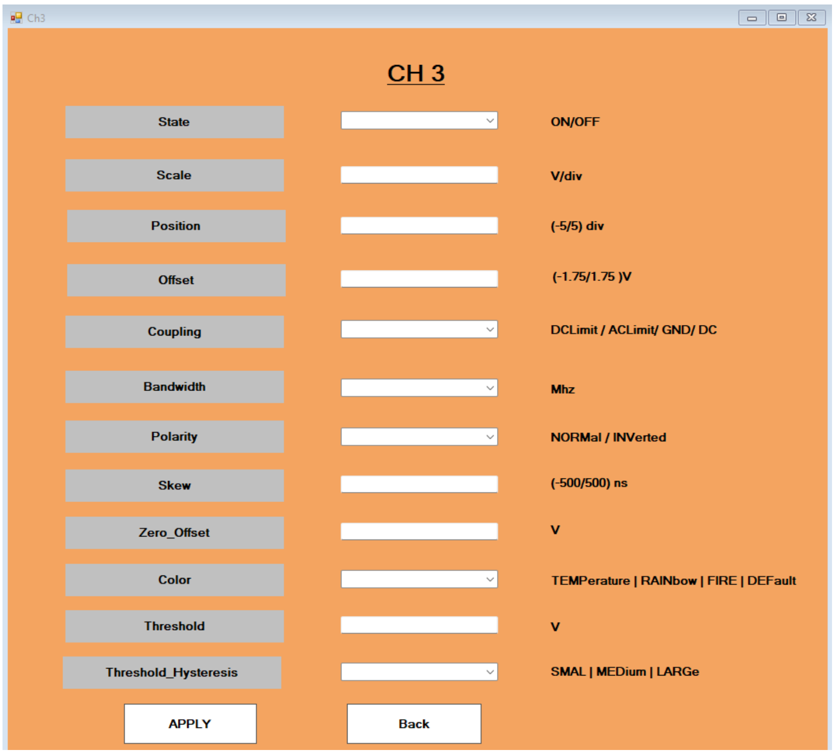
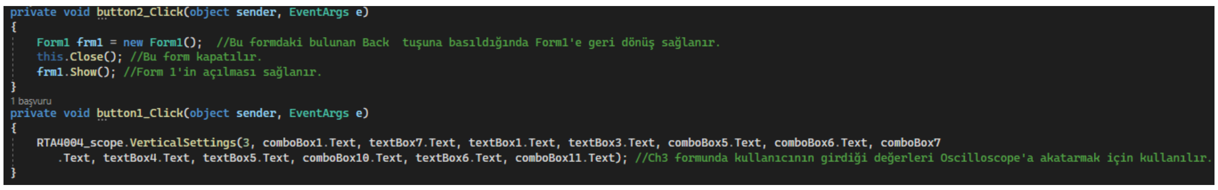


Figure 26 The Picture of Oscilloscope GUI Project: CH3 Form Design

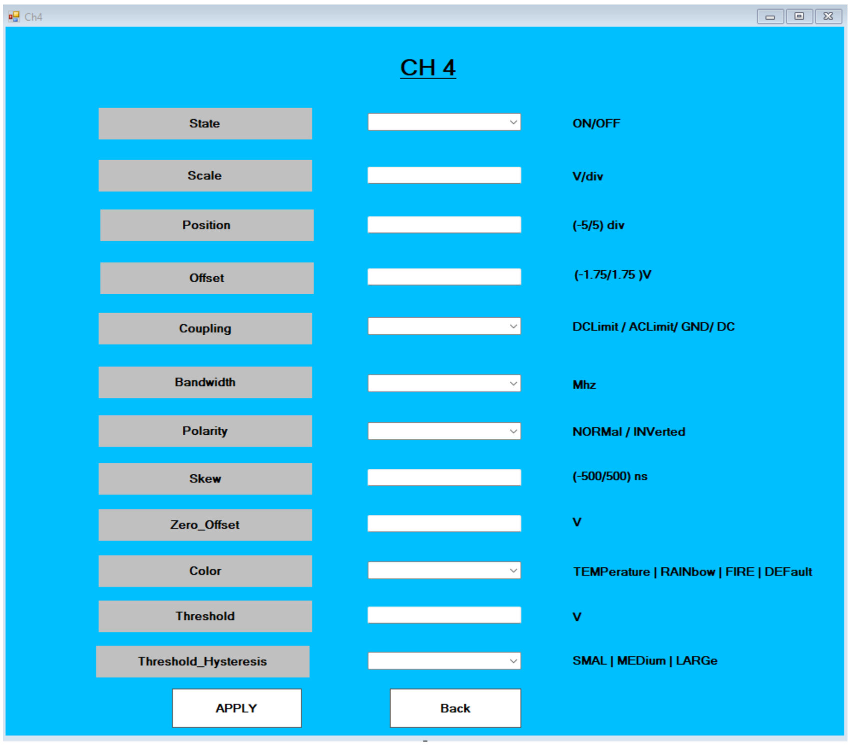
"State, Scale, Position, Offset, Coupling, Bandwidth, Polarity, Skey, Zero_Offset, Color, Threshold, Threshold_Hysteresis" values are set by the user using ***the Ch3 Form*** seen in ***Figure 26***. The values requested by the user are transmitted to Oscilloscope through the ***VerticalSettings*** function in the RTA 4004 Class shown in ***Figure 7***. The codes for ***the Ch3 Form*** are given in the image below.



```
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
1 başvuru
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.VerticalSettings(3, comboBox1.Text, textBox7.Text, textBox1.Text, textBox3.Text, comboBox5.Text, comboBox6.Text, comboBox7
       .Text, textBox4.Text, textBox5.Text, comboBox10.Text, textBox6.Text, comboBox11.Text); //Ch3 formunda kullanıcının girdiği değerleri Oscilloscope'a akatarmak için kullanılır.
}
```

Figure 27 The Picture of Oscilloscope GUI Project: Ch3 Form Codes

The CH4 Form that appears before the user when ***the CH4*** button in the image in ***Figure 3*** is pressed is given in the image below.



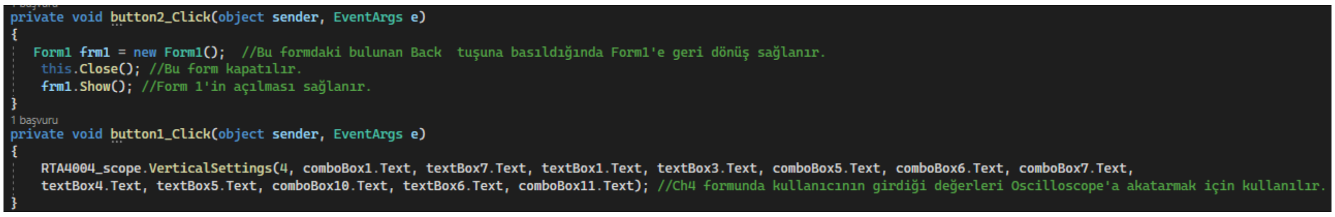
Ch4

CH 4

State		ON/OFF
Scale		V/div
Position		(-5/5) div
Offset		(-1.75/1.75)V
Coupling		DCLimit / ACLimit/ GND/ DC
Bandwidth		Mhz
Polarity		NORMal / INVerted
Skew		(-500/500) ns
Zero_Offset		V
Color		TEMPerature | RAINbow | FIRE | DEFault
Threshold		V
Threshold_Hysteresis		SMAL | MEDium | LARGe

Figure 28 The Picture of Oscilloscope GUI Project: CH4 Form Design

"State, Scale, Position, Offset, Coupling, Bandwidth, Polarity, Skey, Zero_Offset, Color, Threshold, Threshold_Hysteresis" values are set by the user using ***the Ch4 Form*** seen in ***Figure 28***. The values requested by the user are transmitted to the oscilloscope through ***the VerticalSettings*** function in the RTA 4004 Class shown in ***Figure 7***. The codes for ***the Ch4 Form*** are given in the image below.



```
private void button2_Click(object sender, EventArgs e)
{
   Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
1 başvuru
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.VerticalSettings(4, comboBox1.Text, textBox7.Text, textBox1.Text, textBox3.Text, comboBox5.Text, comboBox6.Text, comboBox7.Text,
    textBox4.Text, textBox5.Text, comboBox10.Text, textBox6.Text, comboBox11.Text); //Ch4 formunda kullanıcının girdiği değerleri Oscilloscope'a akatarmak için kullanılır.
}
```

Figure 29 The Picture of Oscilloscope GUI Project: Ch4 Form Codes

The Cursor Form that appears before the user when ***the Cursor*** button in the image in ***Figure 3*** is pressed is given in the image below.

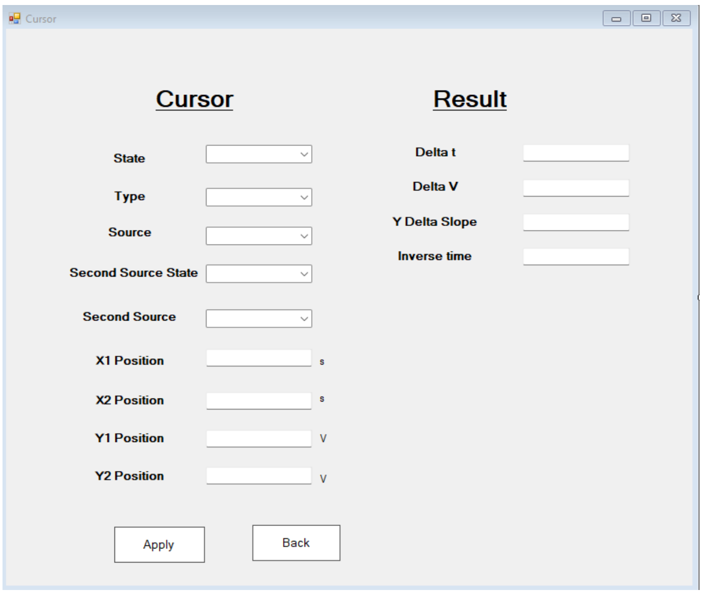
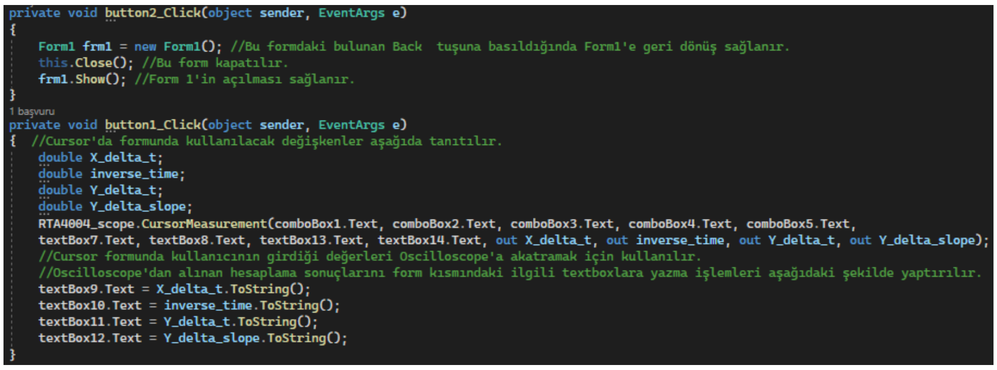


Figure 30 The Picture of Oscilloscope GUI Project: Cursor Form Design

"State, Type, Source, Second Source State, Second Source, X1 Position, X2 Position, Y1 Position, Y2 Position" values are set by the user using ***the Cursor Form*** shown in ***Figure 30***. The values requested by the user are transmitted to the oscilloscope through the ***CursorMeasurement*** function in the RTA 4004 Class shown in ***Figure 8***. The calculation results made using the values sent to the oscilloscope are printed in the result section. The codes for ***the Cursor Form*** are given in the image below.



```
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1(); //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
1 başvuru
private void button1_Click(object sender, EventArgs e)
{  //Cursor'da formunda kullanılacak değişkenler aşağıda tanıtılır.
    double X_delta_t;
    double inverse_time;
    double Y_delta_t;
    double Y_delta_slope;
    RTA4004_scope.CursorMeasurement(comboBox1.Text, comboBox2.Text, comboBox3.Text, comboBox4.Text, comboBox5.Text,
    textBox7.Text, textBox8.Text, textBox13.Text, textBox14.Text, out X_delta_t, out inverse_time, out Y_delta_t, out Y_delta_slope);
    //Cursor formunda kullanıcının girdiği değerleri Oscilloscope'a akatramak için kullanılır.
    //Oscilloscope'dan alınan hesaplama sonuçlarını form kısmındaki ilgili textboxlara yazma işlemleri aşağıdaki şekilde yaptırılır.
    textBox9.Text = X_delta_t.ToString();
    textBox10.Text = inverse_time.ToString();
    textBox11.Text = Y_delta_t.ToString();
    textBox12.Text = Y_delta_slope.ToString();
}
```

Figure 31 The Picture of Oscilloscope GUI Project: Cursor Form Codes

The FFT Form that appears before the user when ***the FFT*** button in the image in ***Figure 3*** is pressed is given in the image below.

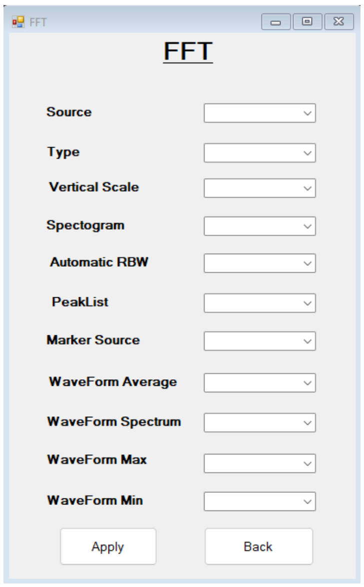
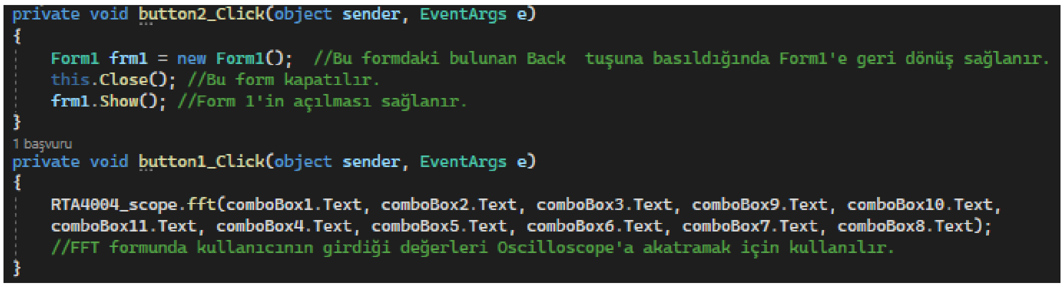


Figure 32 The Picture of Oscilloscope GUI Project: FFT Form Design

"Source, Type, Vertical Scale, Spectogram, Automatic RBW, PeakList, Marker Source, WaveForm Average WaveForm Spectrum, WaveForm Max, WaveForm Min" values are set by the user using ***the FFT Form*** seen in ***Figure 32***. The values requested by the user are transmitted to the Oscilloscope through ***the FFT*** function in the RTA 4004 Class shown in ***Figure 9***. The codes for ***the FFT Form*** are given in the image below.



```
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
1 başvuru
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.fft(comboBox1.Text, comboBox2.Text, comboBox3.Text, comboBox9.Text, comboBox10.Text,
    comboBox11.Text, comboBox4.Text, comboBox5.Text, comboBox6.Text, comboBox7.Text, comboBox8.Text);
    //FFT formunda kullanıcının girdiği değerleri Oscilloscope'a akatramak için kullanılır.
}
```

Figure 33 The Picture of Oscilloscope GUI Project: FFT Form Codes

The Generator Form that appears before the user when ***the Generator*** button in the image in ***Figure 3*** is pressed is given in the image below.

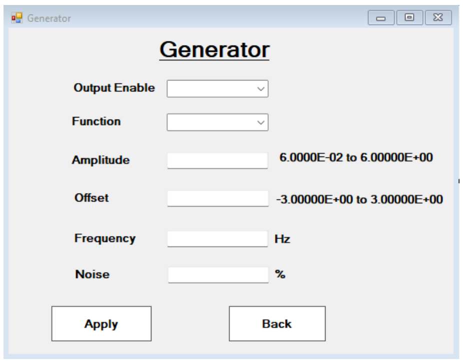
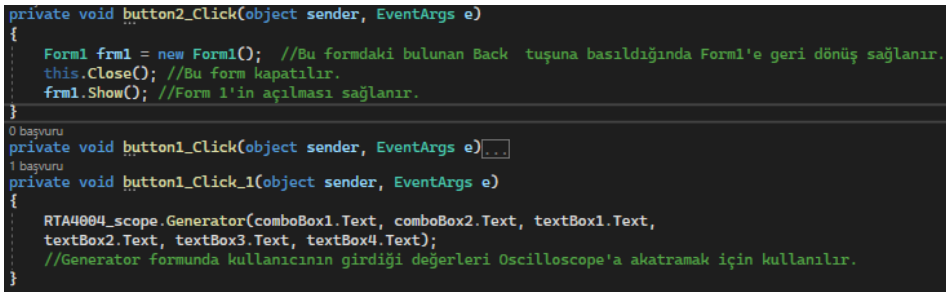


Figure 34 The Picture of Oscilloscope GUI Project: Generator Form Design

"Output Enable, Function, Amplitude, Offset, Frequency, Noise" values are adjusted by the user using ***the Generator Form*** seen in ***Figure 34***. The values requested by the user are transmitted to the oscilloscope through ***the Generator*** function in the RTA 4004 Class shown in ***Figure 9***. The codes for ***the Generator Form*** are given in the image below.



```
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
0 başvuru
private void button1_Click(object sender, EventArgs e)[...]
1 başvuru
private void button1_Click_1(object sender, EventArgs e)
{
    RTA4004_scope.Generator(comboBox1.Text, comboBox2.Text, textBox1.Text,
    textBox2.Text, textBox3.Text, textBox4.Text);
    //Generator formunda kullanıcının girdiği değerleri Oscilloscope'a akatramak için kullanılır.
}
```

Figure 35 The Picture of Oscilloscope GUI Project: Generator Form Codes

The Horizontal Form that appears before the user when ***the Horizontal*** button in the image in ***Figure 3*** is pressed is given in the image below.

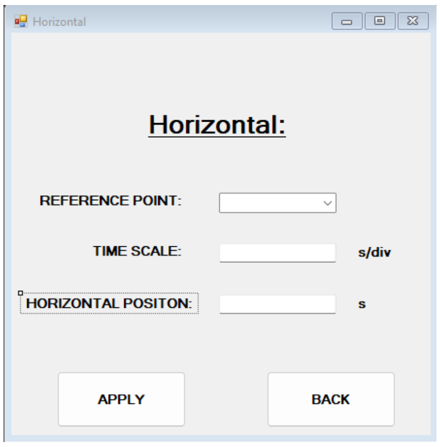
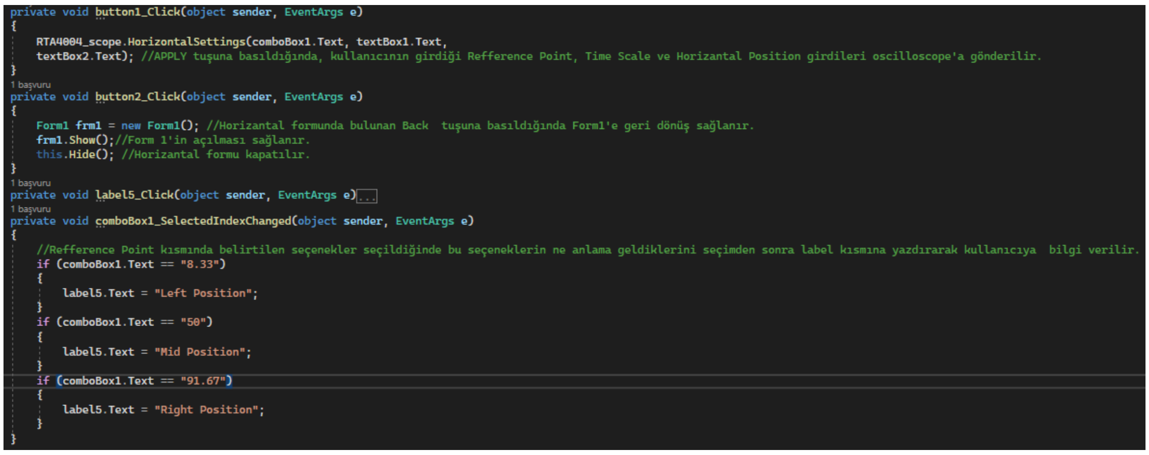


Figure 36 The Picture of Oscilloscope GUI Project: Horizontal Form Design

"Reference Point, Time Scale, Horizontal Position" values are set by the user using ***the Horizontal Form*** seen in ***Figure 36***. The values requested by the user are transmitted to the oscilloscope through ***the Horizontalsettings*** function in the RTA 4004 Class shown in ***Figure6***. The codes for ***the Horizontal Form*** are given in the image below.



```
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.HorizontalSettings(comboBox1.Text, textBox1.Text,
    textBox2.Text); //APPLY tuşuna basıldığında, kullanıcının girdiği Refference Point, Time Scale ve Horizantal Position girdileri oscilloscope'a gönderilir.
}
1 başvuru
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1(); //Horizantal formunda bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    frm1.Show();//Form 1'in açılması sağlanır.
    this.Hide(); //Horizantal formu kapatılır.
}
1 başvuru
private void label5_Click(object sender, EventArgs e)[...]
1 başvuru
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    //Refference Point kısmında belirtilen seçenekler seçildiğinde bu seçeneklerin ne anlama geldiklerini seçimden sonra label kısmına yazdırarak kullanıcıya  bilgi verilir.
    if (comboBox1.Text == "8.33")
    {
        label5.Text = "Left Position";
    }
    if (comboBox1.Text == "50")
    {
        label5.Text = "Mid Position";
    }
    if (comboBox1.Text == "91.67")
    {
        label5.Text = "Right Position";
    }
}
```

Figure 37 The Picture of Oscilloscope GUI Project: Horizontal Form Codes

The Measurement Form that appears before the user when ***the Measurement*** button in the image in ***Figure 3*** is pressed is given in the image below.

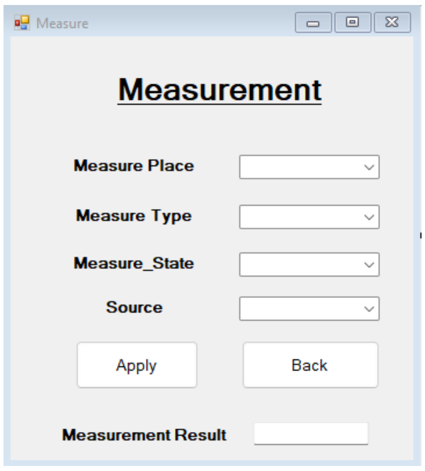
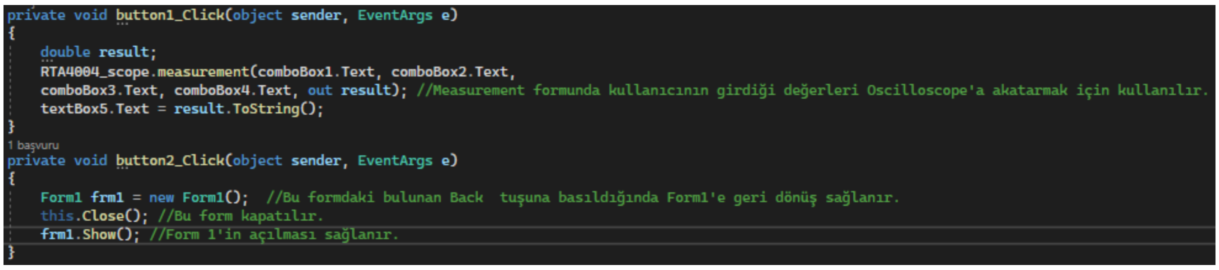


Figure 38 The Picture of Oscilloscope GUI Project: Measurement Form Design

"Measure Place, Measure Type, Measure State, Source" values are set by the user using ***the Measurement Form*** seen in ***Figure 38***. The values requested by the user are transmitted to the oscilloscope through ***the measurement*** function in the RTA 4004 Class shown in ***Figure 8***. The calculation results made using the values sent to the Oscilloscope are printed in ***the Measurement Result*** section. The codes for ***the Measurement Form*** are given in the image below.



```
private void button1_Click(object sender, EventArgs e)
{
    double result;
    RTA4004_scope.measurement(comboBox1.Text, comboBox2.Text,
    comboBox3.Text, comboBox4.Text, out result); //Measurement formunda kullanıcının girdiği değerleri Oscilloscope'a akatarmak için kullanılır.
    textBox5.Text = result.ToString();
}
1 başvuru
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
```

Figure 39 The Picture of Oscilloscope GUI Project: Measurement Form Codes

The Protocol Form that appears before the user when ***the Protocol*** button in the image in ***Figure 3*** is pressed is given in the image below.

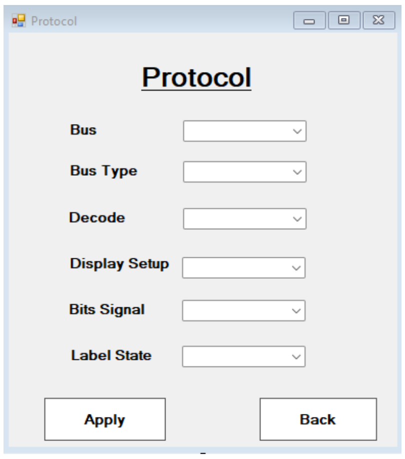
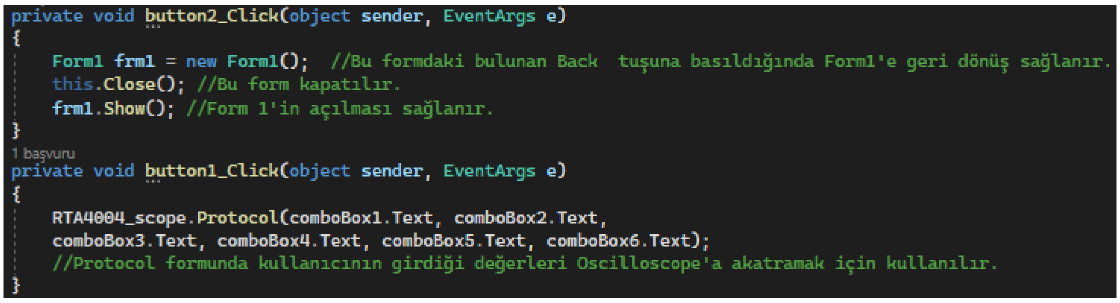


Figure 40 The Picture of Oscilloscope GUI Project: Protocol Form Design

"Bus, Bus Type, Decode, Display Setup, Bits Signal, Label State" values are set by the user using ***the Protocol Form*** seen in ***Figure 40***. The values requested by the user are transmitted to the oscilloscope through ***the Protocol*** function in the RTA 4004 Class shown in ***Figure 9***. The calculation results made using the values sent to the oscilloscope are printed in the result section. The codes for ***the Protocol Form*** are given in the image below.



```
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
1 başvuru
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.Protocol(comboBox1.Text, comboBox2.Text,
    comboBox3.Text, comboBox4.Text, comboBox5.Text, comboBox6.Text);
    //Protocol formunda kullanıcının girdiği değerleri Oscilloscope'a akatramak için kullanılır.
}
```

Figure 41 The Picture of Oscilloscope GUI Project: Protocol Form Codes

The Quick Measure Form that appears before the user when ***the Quick Measure*** button in the image in ***Figure 3*** is pressed is given in the image below.

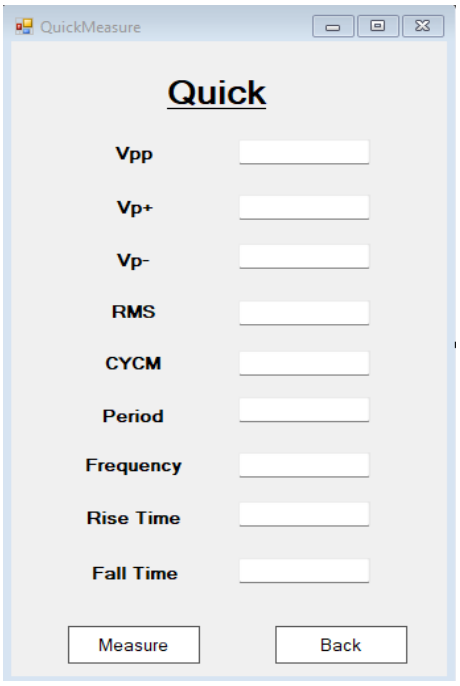
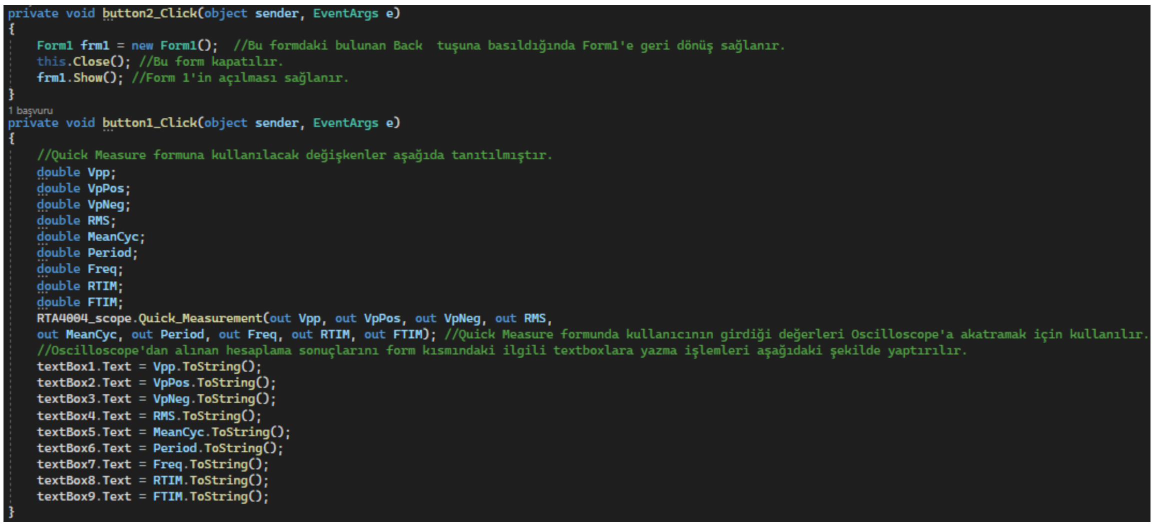


Figure 42 The Picture of Oscilloscope GUI Project: Quick Measure Form Design

When the user presses the measure button in ***the QuickMeasure Form*** shown in ***Figure 42***, the values calculated on the oscilloscope are displayed on the screen. The values requested by the user are transmitted to the oscilloscope form via ***the QuickMeasurement*** function in the RTA 4004 Class shown in ***Figure 8***. The codes for ***the QuickMeasure Form*** are given in the image below.



```
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
1 başvuru
private void button1_Click(object sender, EventArgs e)
{
    //Quick Measure formuna kullanılacak değişkenler aşağıda tanıtılmıştır.
    double Vpp;
    double VpPos;
    double VpNeg;
    double RMS;
    double MeanCyc;
    double Period;
    double Freq;
    double RTIM;
    double FTIM;
    RTA4004_scope.Quick_Measurement(out Vpp, out VpPos, out VpNeg, out RMS,
    out MeanCyc, out Period, out Freq, out RTIM, out FTIM); //Quick Measure formunda kullanıcının girdiği değerleri Oscilloscope'a akatramak için kullanılır.
    //Oscilloscope'dan alınan hesaplama sonuçlarını form kısmındaki ilgili textboxlara yazma işlemleri aşağıdaki şekilde yaptırılır.
    textBox1.Text = Vpp.ToString();
    textBox2.Text = VpPos.ToString();
    textBox3.Text = VpNeg.ToString();
    textBox4.Text = RMS.ToString();
    textBox5.Text = MeanCyc.ToString();
    textBox6.Text = Period.ToString();
    textBox7.Text = Freq.ToString();
    textBox8.Text = RTIM.ToString();
    textBox9.Text = FTIM.ToString();
}
```

Figure 43 The Picture of Oscilloscope GUI Project: Quick Measure Form Codes

The Screen Shot Form that appears before the user when ***the Screen Shot*** button in the image in ***Figure 3*** is pressed is given in the image below.

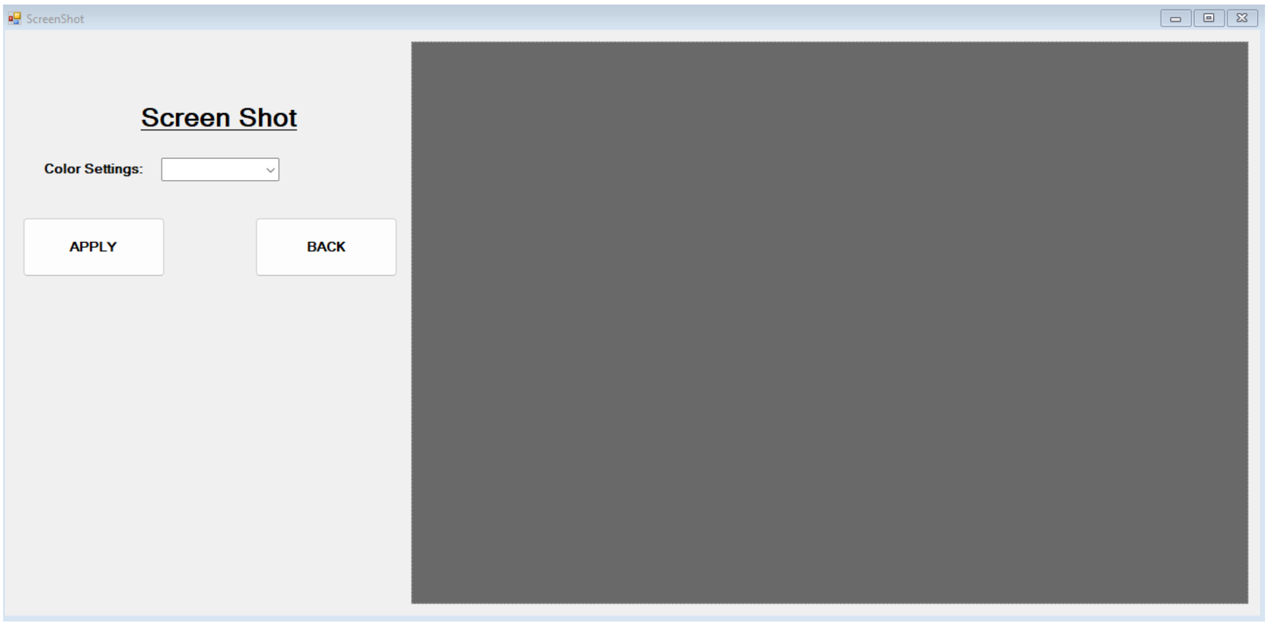
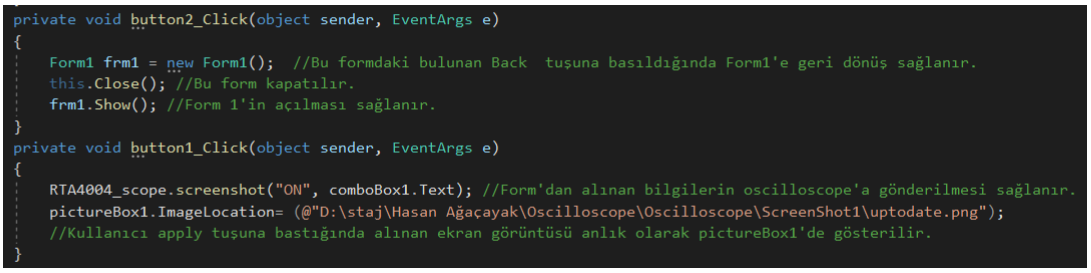


Figure 44 The Picture of Oscilloscope GUI Project: Screen Shot Form Design

"Color Settings" value is set by the user using ***the ScreenShot Form*** seen in ***Figure 44***. The values requested by the user are transmitted to the oscilloscope through ***the screenshot*** function in the RTA 4004 Class, shown in ***Figure 10***. A screenshot is taken using the commands sent to the Oscilloscope and shown in ***the picturebox*** on the right side of the form. The codes for ***the ScreenShot Form*** are given in the image below.



```
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Bu formdaki bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Bu form kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.screenshot("ON", comboBox1.Text); //Form'dan alınan bilgilerin oscilloscope'a gönderilmesi sağlanır.
    pictureBox1.ImageLocation= (@"D:\staj\Hasan Ağaçayak\Oscilloscope\Oscilloscope\ScreenShot1\uptodate.png");
    //Kullanıcı apply tuşuna bastığında alınan ekran görüntüsü anlık olarak pictureBox1'de gösterilir.
}
```

Figure 45 The Picture of Oscilloscope GUI Project: Screen Shot Form Codes

The Trigger Form that appears before the user when ***the Trigger*** button in the image in ***Figure 3*** is pressed is given in the image below.

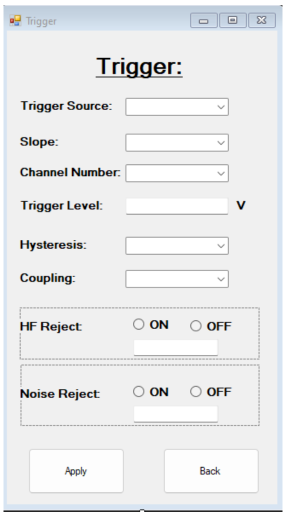
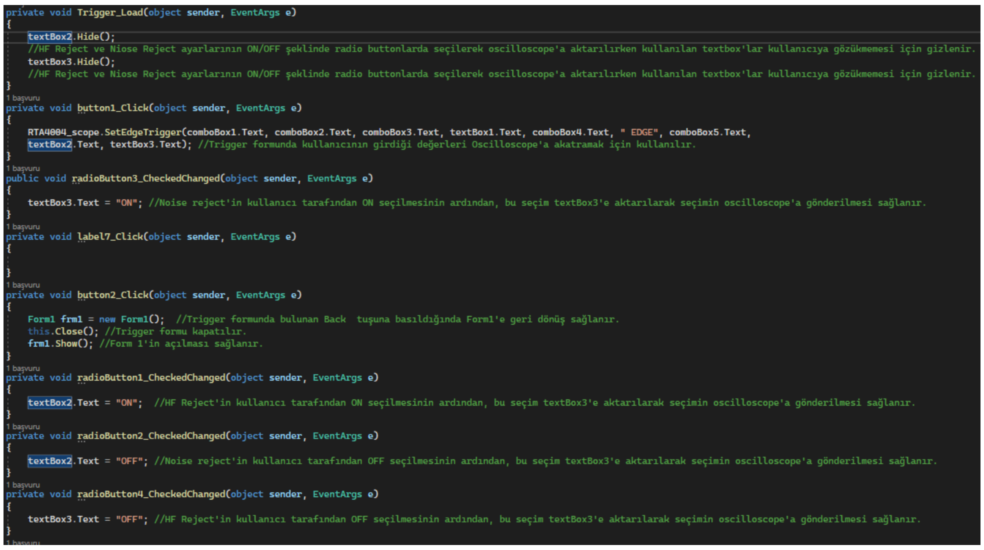


Figure 46 The Picture of Oscilloscope GUI Project: Trigger Form Design

"Trigger Source, Slope, Channel Number, Trigger Level, Hysteresis, Coupling, HF Reject, Noise Reject" values are set by the user using ***the Trigger Form*** shown in ***Figure 46***. The values requested by the user are transmitted to the oscilloscope through ***the SetEdgeTrigger*** function in the RTA 4004 Class shown in ***Figure 6***. The codes for ***the Triger Form*** are given in the image below.



```
private void Trigger_Load(object sender, EventArgs e)
{
    textBox2.Hide();
    //HF Reject ve Niose Reject ayarlarının ON/OFF şeklinde radio buttonlarda seçilerek oscilloscope'a aktarılırken kullanılan textbox'lar kullanıcıya gözükmemesi için gizlenir.
    textBox3.Hide();
    //HF Reject ve Niose Reject ayarlarının ON/OFF şeklinde radio buttonlarda seçilerek oscilloscope'a aktarılırken kullanılan textbox'lar kullanıcıya gözükmemesi için gizlenir.
}
1 başvuru
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.SetEdgeTrigger(comboBox1.Text, comboBox2.Text, comboBox3.Text, textBox1.Text, comboBox4.Text, " EDGE", comboBox5.Text,
    textBox2.Text, textBox3.Text); //Trigger formunda kullanıcının girdiği değerleri Oscilloscope'a akatramak için kullanılır.
}
1 başvuru
public void radioButton3_CheckedChanged(object sender, EventArgs e)
{
    textBox3.Text = "ON"; //Noise reject'in kullanıcı tarafından ON seçilmesinin ardından, bu seçim textBox3'e aktarılarak seçimin oscilloscope'a gönderilmesi sağlanır.
}
1 başvuru
private void label7_Click(object sender, EventArgs e)
{


}
1 başvuru
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Trigger formunda bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Trigger formu kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
1 başvuru
private void radioButton1_CheckedChanged(object sender, EventArgs e)
{
    textBox2.Text = "ON";  //HF Reject'in kullanıcı tarafından ON seçilmesinin ardından, bu seçim textBox3'e aktarılarak seçimin oscilloscope'a gönderilmesi sağlanır.
}
1 başvuru
private void radioButton2_CheckedChanged(object sender, EventArgs e)
{
    textBox2.Text = "OFF"; //Noise reject'in kullanıcı tarafından OFF seçilmesinin ardından, bu seçim textBox3'e aktarılarak seçimin oscilloscope'a gönderilmesi sağlanır.
}
1 başvuru
private void radioButton4_CheckedChanged(object sender, EventArgs e)
{
    textBox3.Text = "OFF"; //HF Reject'in kullanıcı tarafından OFF seçilmesinin ardından, bu seçim textBox3'e aktarılarak seçimin oscilloscope'a gönderilmesi sağlanır.
}
1 başvuru
```

Figure 47 The Picture of Oscilloscope GUI Project: Trigger Form Codes

The Zoom Form that appears before the user when ***the Zoom*** button in the image in ***Figure 3*** is pressed is given in the image below.

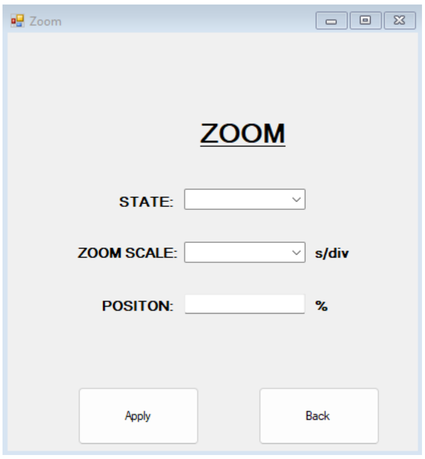
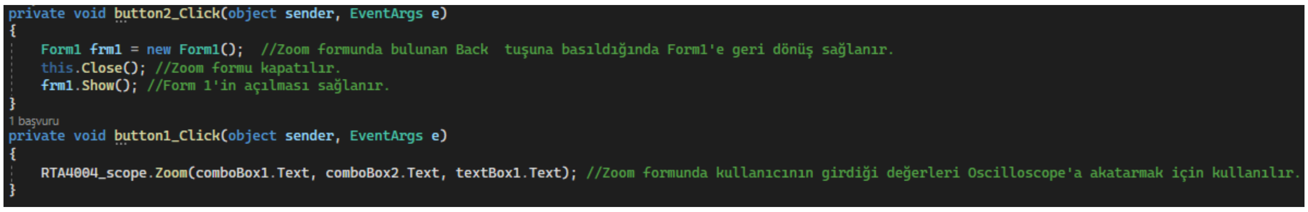


Figure 48 The Picture of Oscilloscope GUI Project: Zoom Form Design

"State, Zoom Scale, Position" values are set by the user using ***the Zoom Form*** seen in ***Figure 48***. The values requested by the user are transmitted to the oscilloscope through ***the Zoom*** function in the RTA 4004 Class, shown in ***Figure 6***. The codes for ***the Zoom Form*** are given in the image below.



```
private void button2_Click(object sender, EventArgs e)
{
    Form1 frm1 = new Form1();  //Zoom formunda bulunan Back  tuşuna basıldığında Form1'e geri dönüş sağlanır.
    this.Close(); //Zoom formu kapatılır.
    frm1.Show(); //Form 1'in açılması sağlanır.
}
1 başvuru
private void button1_Click(object sender, EventArgs e)
{
    RTA4004_scope.Zoom(comboBox1.Text, comboBox2.Text, textBox1.Text); //Zoom formunda kullanıcının girdiği değerleri Oscilloscope'a akatarmak için kullanılır.
}
```

Figure 49 The Picture of Oscilloscope GUI Project: Zoom Form Codes