

**FSST 3. Jahrgang**Qt Projekt 2014

***Signalanalyse***

**Signalanalyse**

ausgeführt von

**Hofstätter**  Alexander 3BHEL

am 6. Juni 2014

# Aufgabenstellung

Einlesen von Messdaten von einer Datei,

mit grafischer Darstellung der Messdaten und

Berechnung von Mittelwert, Uss und RMS.

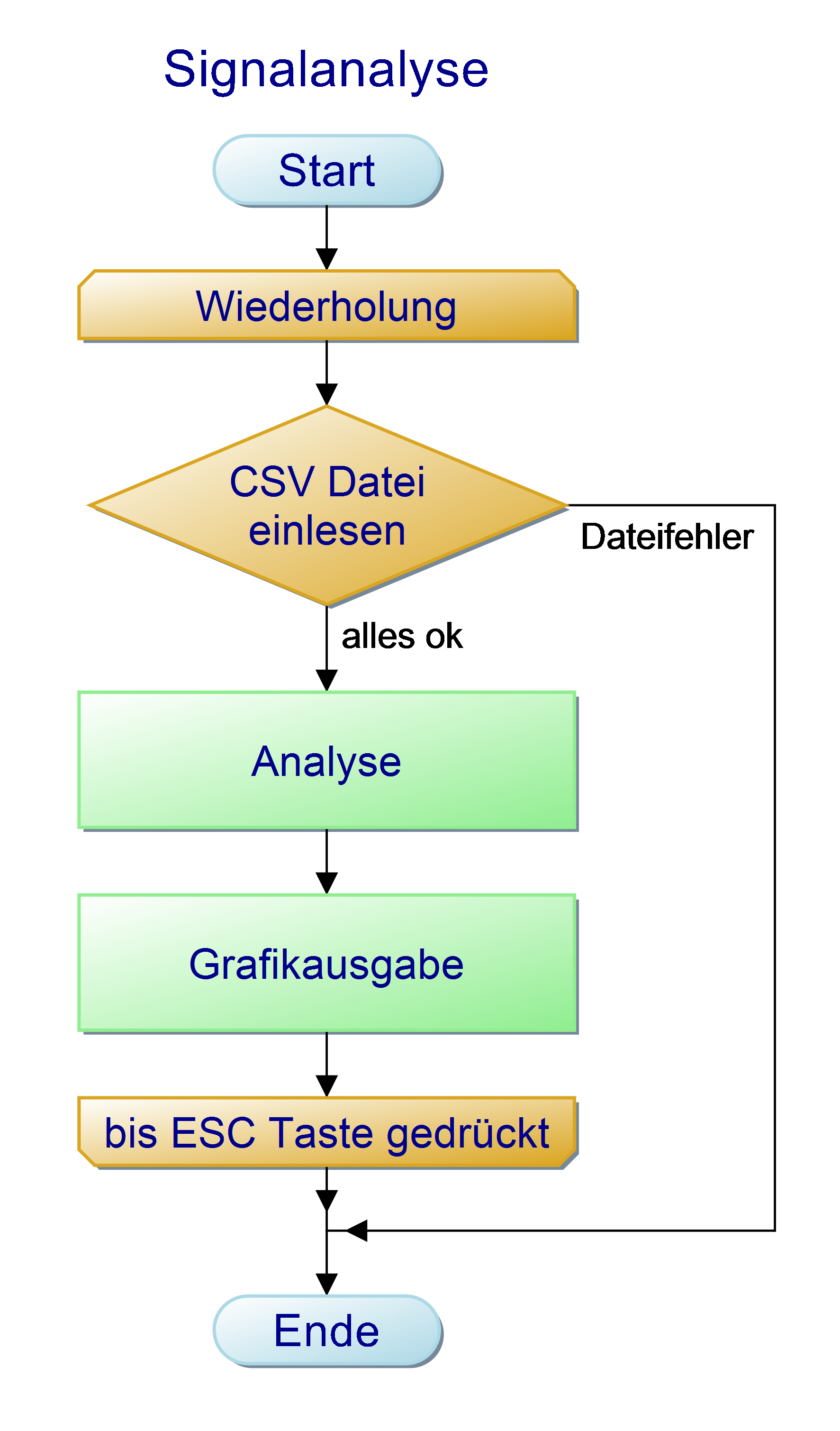
# Programmbeschreibung

Beim Programmstart wird die Einggabedatei eingelesen (CSV-Datei mit allen Messdaten direkt vom Oszilloskop)

Anschließend werden die Daten analysiert und visualisiert.

Auf der rechten Fensterseite sind diverse Analysewerte zu finden (RMS, Mittel und Spitze-Spitze)

# Programmübersicht

Alle verwendeten Methoden und Klassen findet man in grafik.h sowie mainwindow.h und den dazugehörigen Source-files.

Die CSV-Datei folgt folgenden Format Regeln:

Zeit,Wert-Kanal-1,Wert-Kanal-2

x-axis,1,2

second,Volt,Volt

-2.800000E-03,+2.226130597E+00,+1.594057798E+00

-2.797200E-03,+2.226130597E+00,+1.594057798E+00

-2.794400E-03,+2.301507481E+00,+1.594057798E+00

-2.791600E-03,+2.306532606E+00,+1.594057798E+00

-2.788800E-03,+2.306532606E+00,+1.594057798E+00

-2.786000E-03,+2.306532606E+00,+1.678479910E+00

...

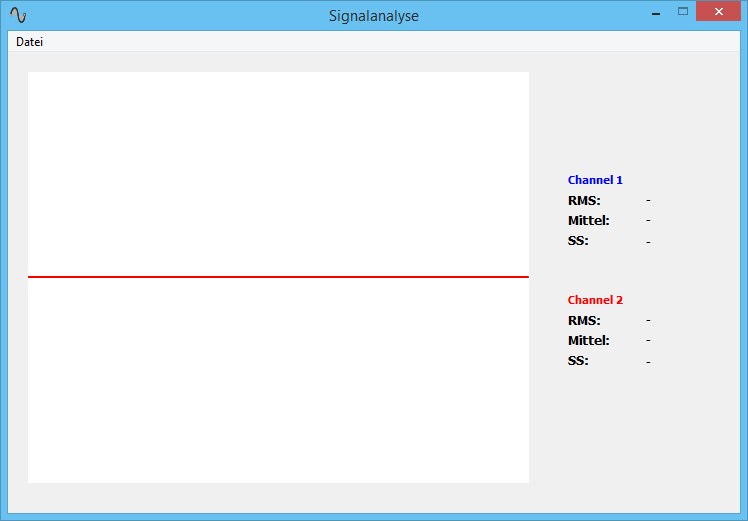
...

...

Das Oszilloskope liefert CSV-Dateien mit ca. 2000 Werten. Es können aber beliebig viele Werte übermittelt werden, da das Array dynamisch alloziert wird.

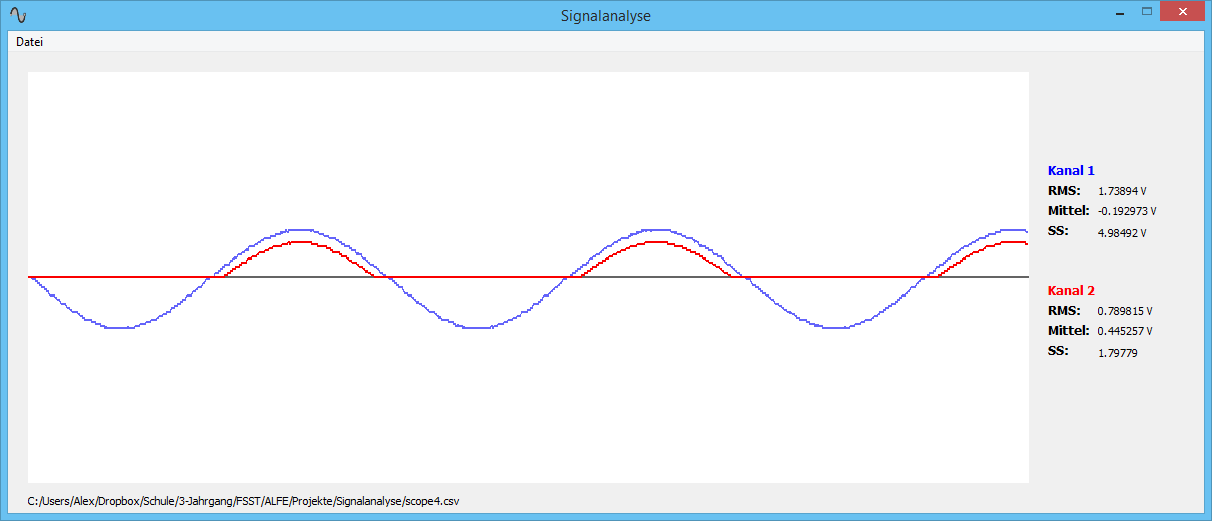
# Benutzerhandbuch

Direkt nach dem Programmstart erscheint ein leeres Grafikfenster. Nach laden der CSV Datei über den Menüpunkt: „Datei -> Datei laden“ erscheint ein visualisiertes Signal Array inlusive aller Analysewerte (RMS, Mittel und Spitze-Spitze)



Je nach CSV File sind 1-2 Kanäle vorhanden. Kanal 1 wird in blau und Kanal 2 in rot dargestellt.

Rechts außen werden Mittelwerte, RMS und Spitze-Spitze dargestellt. Unten wird der absolute Pfad zur CSV Datei eingeblendet.



# Klassenübersicht

## Klasse „grafik“

class grafik : public QWidget

{

protected:

static const int N=1000;

float samples[N];

float samples2[N];

public:

grafik(QWidget \*w);

virtual void *paintEvent*( QPaintEvent \* );

int getN()

{ return N;

}

float\* getsamples()

{ return samples;

}

float\* getsamples2()

{ return samples2;

}

};

## Klasse „MainWindow“

class MainWindow : public QMainWindow

{

Q\_OBJECT

private:

Ui::MainWindow \*ui;

int NMAX;

void getAm();

void getFre();

void getPhi();

void getSu();

public:

explicit MainWindow(QWidget \*parent = 0);

~*MainWindow*();

QString filename;

float am,fr,phi,su;

float \*samples;

float \*samples2;

float array\_max ( float a[], int n1 , int n2 );

float array\_min ( float a[], int n1 , int n2 );

float array\_mittel(float a[], int n1, int n2);

float array\_ss ( float a[], int n1 , int n2 );

float array\_rms ( float a[], int n1, int n2 );

private slots:

void on\_actionDatei\_laden\_triggered();

};