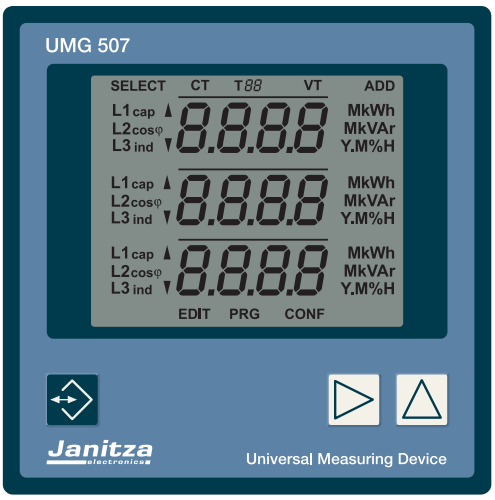
Janitza electronics

Universal Measuring Device

UMG507

Funktionsbeschreibung

Webserver



Dok. Nr. 10322100.pmd

Janitza electronics GmbH
Vor dem Polstück 1
35633 Lahnau
Support Tel. (06441) 9642-22
email: info@janitza.de
internet: http://www.janitza.de

Allgemein

Das UMG507 unterstützt je nach Ausführungsvariante die Protokolle Modbus RTU, Modbus TCP/IP, Modbus over TCP/IP (Modbus Gateway) oder Profibus DP V0. Diese Funktionsbeschreibung ist eine Ergänzung zum Handbuch und beschreibt schrittweise die notwendigen Einrichtungsschritte der jeweiligen Funktion
Weitere Funktionsbeschreibungen finden Sie auf der CD ROM PSWbasic/professional. Derzeit sind folgende Funktionsbeschreibungen erhältlich:

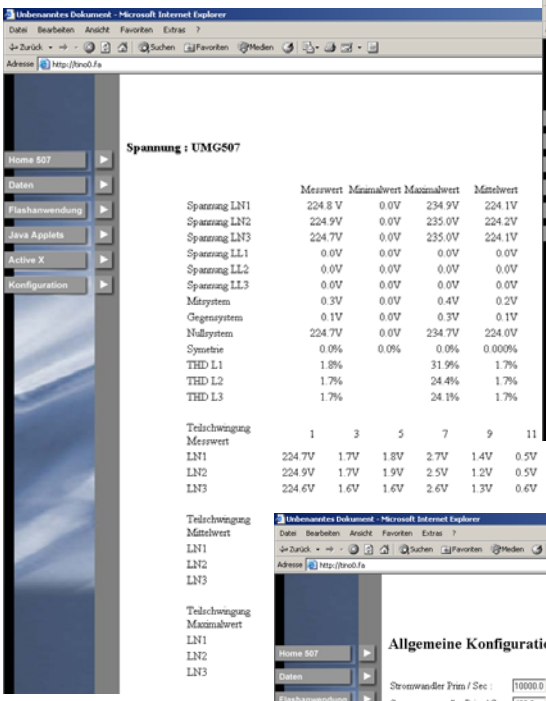
- UMG507 als Datenanzeige für externe Modbus Slaves
- OPC Server Port 502
- OPC Server Port 8000 (Modbus Gateway Funktion)
- Der Webserver des UMG507
- Speicheraufbau des UMG507
- Profibus Beschreibung mit Beispielen
- Maximumüberwachung
- Applikationen

Ausgabevermerk:

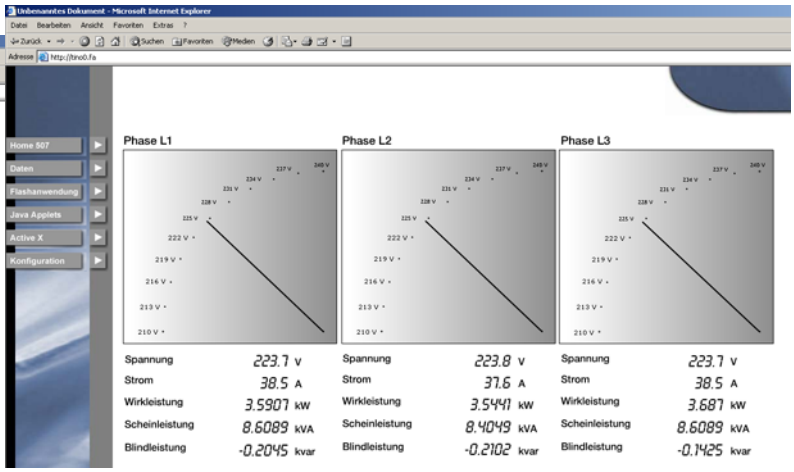
18.11.2004	Erstausgabe / Wagner
22.11.2004	Korrekturlesung erfolgt

Der Webserver des UMG507E

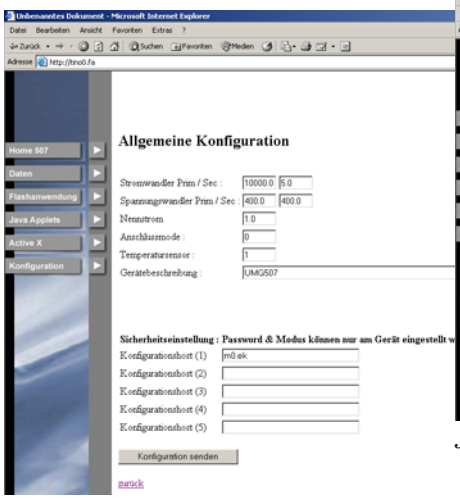
Der Anwender kann nach eigenen Vorstellungen HTML Seiten, Java-Applets, Active X-Komponeten und Macromedia Flash Files auf dem internen Webserver des UMG507E/EP ablegen. Über das Intranet oder Internet sind die Seiten über jeden Internetbrowser abrufbar. Die komplette Parametrierung des Gerätes kann ebenfalls über die Webseite des Gerätes erfolgen. Zur Übertragung von eigenen Websites finden Sie auf der CD ROM PSWbasic/professional ein Upload Programm sowie Beispiel HTML Seiten. Diese Beispielseiten dienen als Muster und können für eigene Applikationen geändert werden.



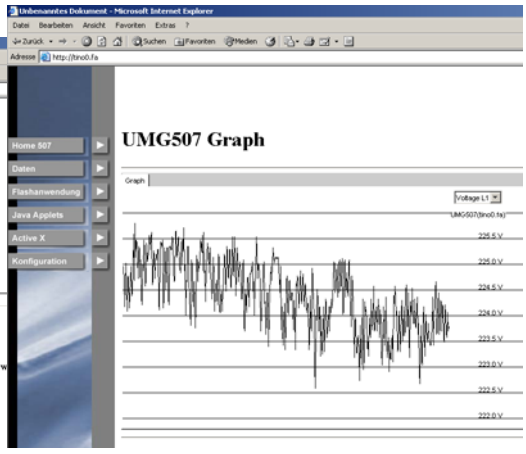
Aktuelle Messwerte



Macromedia Flash Applikation



Konfiguration



JAVA Applikation

Über einen symbolischen Namen werden die Messgrößen in die jeweilige Website eingebunden. Der Parameter „rep“ gefolgt vom symbolischen Namen z.B. „ul1“ wird beim Aufruf der Website durch den aktuellen Messwert ersetzt. Eine Adressenliste mit allen symbolischen Namen finden Sie auf der CD PSWbasic/professional.

Auszug aus der Adressenliste:

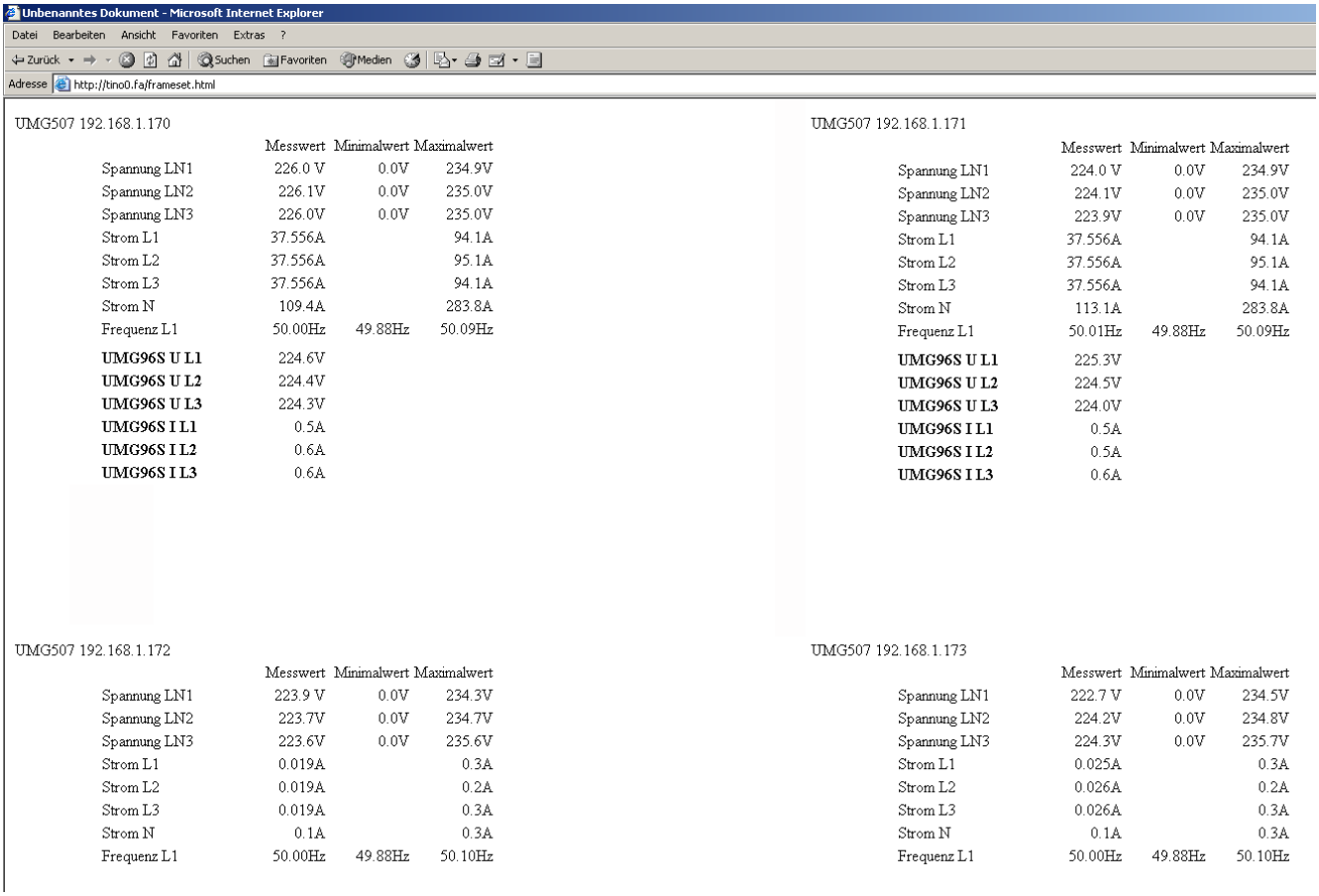
Bezeichnung	**Symbolischer Name**	**HTML Befehl**
Spannung Phase L1	ul1	<rep ul1>
Spannung Phase L1	ul2	<rep ul2>
Spannung Phase L1	ul3	<rep ul3>
Strom Phase L1	il1	<rep il1>
Strom Phase L1	il2	<rep il2>
Strom Phase L1	il3	<rep il3>
Leistung Phase L1	pl1	<rep pl1>
Leistung Phase L1	pl2	<rep pl2>
Leistung Phase L1	pl3	<rep pl3>

Beispiele zum Webserver

Im folgendem Kapitel finden Sie diverse Beispiele zur Nutzung des internen Webservers.

Beispiel HTML

Darstellung von mehreren UMG507E und UMG96S auf einer Webseite mit automatischer Aktualisierung.



UMG507 192.168.1.170

	Messwert	Minimalwert	Maximalwert
Spannung LN1	226.0 V	0.0V	234.9V
Spannung LN2	226.1V	0.0V	235.0V
Spannung LN3	226.0V	0.0V	235.0V
Strom L1	37.556A		94.1A
Strom L2	37.556A		95.1A
Strom L3	37.556A		94.1A
Strom N	109.4A		283.8A
Frequenz L1	50.00Hz	49.88Hz	50.09Hz
UMG96S U L1	224.6V		
UMG96S U L2	224.4V		
UMG96S U L3	224.3V		
UMG96S I L1	0.5A		
UMG96S I L2	0.6A		
UMG96S I L3	0.6A		

UMG507 192.168.1.171

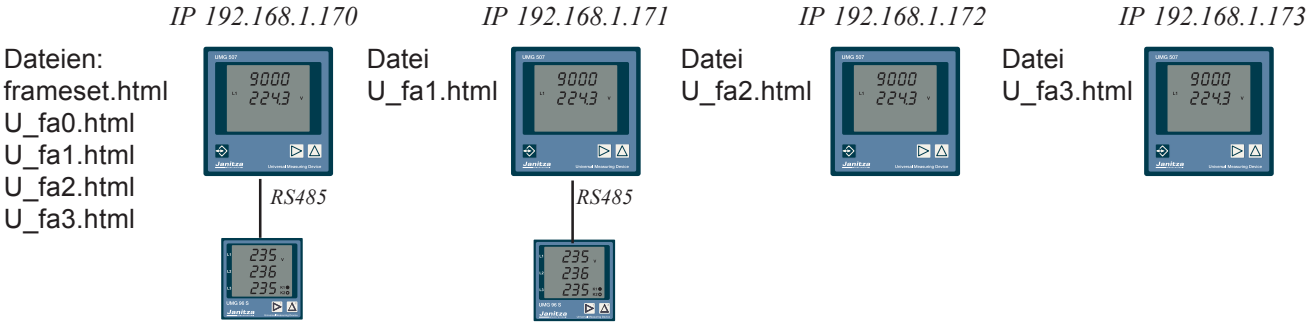
	Messwert	Minimalwert	Maximalwert
Spannung LN1	224.0 V	0.0V	234.9V
Spannung LN2	224.1V	0.0V	235.0V
Spannung LN3	223.9V	0.0V	235.0V
Strom L1	37.556A		94.1A
Strom L2	37.556A		95.1A
Strom L3	37.556A		94.1A
Strom N	113.1A		283.8A
Frequenz L1	50.01Hz	49.88Hz	50.09Hz
UMG96S U L1	225.3V		
UMG96S U L2	224.5V		
UMG96S U L3	224.0V		
UMG96S I L1	0.5A		
UMG96S I L2	0.5A		
UMG96S I L3	0.6A		

UMG507 192.168.1.172

	Messwert	Minimalwert	Maximalwert
Spannung LN1	223.9 V	0.0V	234.3V
Spannung LN2	223.7V	0.0V	234.7V
Spannung LN3	223.6V	0.0V	235.6V
Strom L1	0.019A		0.3A
Strom L2	0.019A		0.2A
Strom L3	0.019A		0.3A
Strom N	0.1A		0.3A
Frequenz L1	50.00Hz	49.88Hz	50.10Hz

UMG507 192.168.1.173

	Messwert	Minimalwert	Maximalwert
Spannung LN1	222.7 V	0.0V	234.5V
Spannung LN2	224.2V	0.0V	234.8V
Spannung LN3	224.3V	0.0V	235.7V
Strom L1	0.025A		0.3A
Strom L2	0.026A		0.2A
Strom L3	0.026A		0.3A
Strom N	0.1A		0.3A
Frequenz L1	50.00Hz	49.88Hz	50.10Hz



Aufgabenstellung:
Beim Aufruf der Webseite des UMG507 mit der IP Adresse 192.168.1.170 sollen die Messwerte von vier UMG507 und jeweils zwei UMG96S auf einer HTML Seite dargestellt werden. Die Aktualisierung der Messwerte soll jede Sekunde erfolgen.

Lösung:
Dies ist nur über ein entsprechendes Frameset zu lösen. Legen Sie hierfür ein Framset mit vier unterschiedlichen Frames auf dem UMG507 mit der IP Adresse 192.168.1.170 ab. Die Umleitung zu den anderen Geräten erfolgt über den META TAG Befehl **„meta http-equiv="refresh" content="1; url=http://192.168.1.171/U_fa1.html“**

Programmierung des Frameset für Gerät 192.168.1.170
Dateiname: Frameset.html

```
<html>
<head>
<title>UMG507E  IP 192.168.1.170</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<frameset rows="50%,50%" cols="*" framespacing="0"" frameborder="no" border="0">
  <frameset rows="*" cols="50%,50%">
    <frame src="U_fa0.html" name="topFrame" scrolling="auto" >
    <frame src="U_fa1.html" name="topFrame" >
  </frameset>
  <frameset rows="*" cols="50%,50%">
    <frame src="U_fa2.html" name="topFrame" >
    <frame src="U_fa3.html" name="topFrame" >
  </frameset>
</frameset>
<noframes>
<body>
Ihr Browser unterstützt keine Frames
</body>
</noframes>
</html>
```

Auszug aus der Frameseite für Gerät IP 192.168.1.170
Dateiname: U_fa0.html

```
<html>
<head>
<title>UMG507E  IP 192.168.1.170</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<meta http-equiv="refresh" content="1; url=U_fa0.html">
</head>
<body bgcolor="#FFFFFF">
<table width="332" border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td width="332" height="19" valign="top">UMG507 192.168.1.170</td>
  </tr>
</table>
<table border="0">
<colgroup> <col width="80"> <col width="140"> <col width="80" align=right><col width="80" align=right><col
width="80" align=right> </colgroup>
<tr><td></td><td></td><td> Messwert </td><td> Minimalwert</td><td>Maximalwert</td></tr>
<tr><td></td><td>Spannung LN1</td><td> <rep ul1>V</td><td> <rep ul1min>V</td><td><rep ul1max>V</td></
tr>
.....
<tr><td></td><td> </td><td> </td></tr>
<tr><td></td><td><b>UMG96S U L1</b></td><td><rep darray_1>V</td></tr>
<tr><td></td><td><b>UMG96S U L2</b></td><td><rep darray_2>V</td></tr>
.....
```

Über den META Refresh Befehl wird die Seite im Sekundenintervall automatisch neu geladen. Die Messwerte des UMG96S werden über die symbolischen Namen der Datenarrays (Adresse 9000 - 9126) (siehe Adressenliste) eingebunden.

Auszug aus der Frameseite für Gerät IP 192.168.1.171
Dateiname: U_fa1.html
(die Dateien U_fa2.html, U_fa3.html sind identisch aufgebaut)

```
<html>
<head>
<title>UMG507E  IP 192.168.1.171</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<meta http-equiv="refresh" content="1; url=http://192.168.1.171/U_fa1.html">
</head>
<body bgcolor="#FFFFFF">
<table width="332" border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td width="332" height="19" valign="top">UMG507 192.168.1.171</td>
  </tr>
  <tr>
    <td height="2"></td>
  </tr>
</table>
<table border="0">
<colgroup> <col width="80"> <col width="140"> <col width="80" align=right><col width="80" align=right><col width="80" align=right> </colgroup>

<tr><td></td><td></td><td> Messwert </td><td> Minimalwert</td><td>Maximalwert</td></tr>

<tr><td></td><td>Spannung LN1</td><td> <rep ul1>     V</td><td> <rep ul1min>V</td><td><rep ul1max>V</td></tr>
<tr><td></td><td>Spannung LN2</td><td><rep ul2>V</td><td><rep ul2min>V</td><td><rep ul2max>V</td></tr>
<tr><td></td><td>Spannung LN3</td><td><rep ul3>V</td><td><rep ul3min>V</td><td><rep ul3max>V</td></tr>
<tr><td></td><td>Strom L1</td><td><rep il1,3>A</td>   <td></td>          <td><rep il1max>A</td></tr>
<tr><td></td><td>Strom L2</td><td><rep il2,3>A</td>   <td></td>          <td><rep il2max>A</td></tr>
<tr><td></td><td>Strom L3</td><td><rep il3,3>A</td>   <td></td>          <td><rep il3max>A</td></tr>
<tr><td></td><td>Strom N</td><td><rep is>A</td>        <td></td>          <td><rep ismax>A</td></tr>
<tr><td></td><td>Frequenz L1</td><td><rep fl1,2>Hz</td><td><rep fl1min,2>Hz</td><td><rep fl1max,2>Hz</td></tr>
<tr><td></td><td> </td><td> </td></tr>
<tr><td></td><td> </td><td> </td></tr>
<tr><td></td><td><b>UMG96S U L1</b></td><td><rep darray_1>V</td></tr>
<tr><td></td><td><b>UMG96S U L2</b></td><td><rep darray_2>V</td></tr>
<tr><td></td><td><b>UMG96S U L3</b></td><td><rep darray_3>V</td></tr>
<tr><td></td><td><b>UMG96S I L1</b></td><td><rep darray_4>A</td></tr>
<tr><td></td><td><b>UMG96S I L2</b></td><td><rep darray_5>A</td></tr>
<tr><td></td><td><b>UMG96S I L3</b></td><td><rep darray_6>A</td></tr>
</table>
</body>
</html>
```

Beim Aufruf des Framesets wird zunächst die Seite „U_fa1.html“ vom UMG507 mit der IP Adresse 192.168.1.170 geladen. Durch die automaticheAktualisierung mit Angabe der **„url“** wird die Seite beim nächsten Aufruf automatisch vom Gerät mit der IP Adresse 192.168.1.171 geladen und jede Sekunde aktualisiert. Die Seiten U_fa2.html und U_fa3.html sind, mit Ausnahme der URL, identisch aufgebaut.

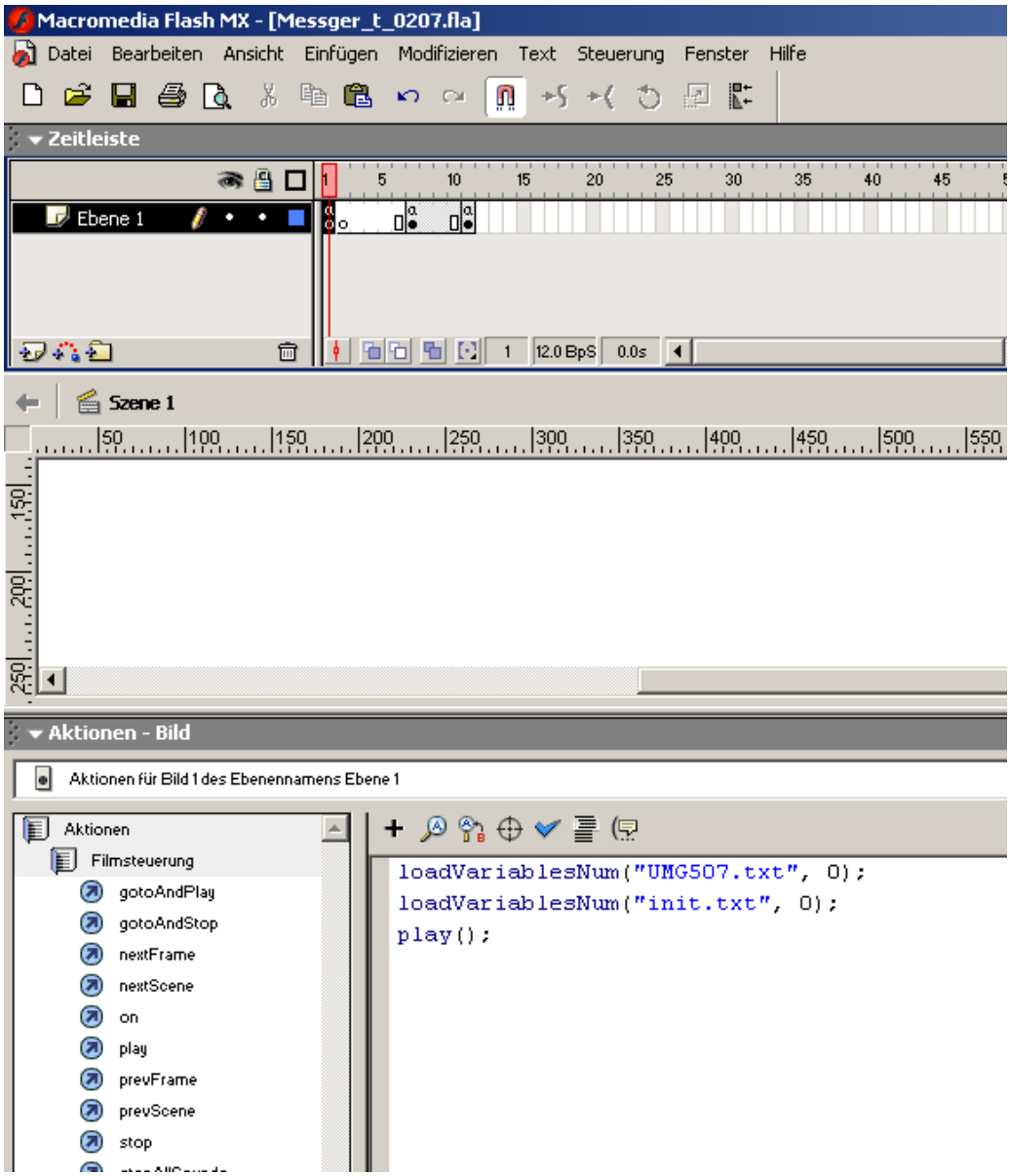
Wichtiger Hinweis: Damit der Parser des UMG507 die symbolischen Namen durch die aktuelle Messgröße ersetzen kann, dürfen diese nur mit der Dateiendung „html“ abgespeichert werden. Die veraltete Dateiendung „htm“ wird nicht mehr unterstützt. Ferner dürfen die Dateinamen maximal 8 Zeichen haben.

Beispiel Macromedia Flash

Über Macromedia Flash können sehr anspruchsvolle Anwenderoberflächen gestaltet werden. Dabei sind der grafischen Gestaltung fast keine Grenzen gesetzt. Das UMG507E/EP versteht Flashfiles der Versionen MX und 2004. Ältere Versionen werden nicht mehr unterstützt.
Flash bietet die Möglichkeit während der Laufzeit auf externe Dateien zuzugreifen und den Inhalt dieser Dateien im aktuellen Flash Player Film anzuzeigen. Das UMG507E/EP nutzt diese Möglichkeit und ersetzt ähnlich wie bei den HTML Seiten Parameter Befehle in einer „ini“ Datei durch die aktuellen Messwerte und schreibt diese in eine TxT Datei. Diese Txt Datei kann von Flash während der Laufzeit gelesen werden.
Die INI Datei zur Festlegung der Messwerte muss den folgenden Aufbau haben:

&Var_0=<rep ul1>
&Var_2=<rep ul2>
&Var_4=<rep ul3>
&Var_6=<rep il1>
.....

&Var_0 ist hierbei die interne Flashvariable. Dieser Variable wird der aktuelle Spannungswert durch den Parameter **<rep ul1>** zugewiesen.



Über den Befehl **„loadVariablesNum("UMG507.txt", 0)“** werden die Variablen beim Filmstart eingelesen und können innerhalb des Flash Films weiterverarbeitet werden. Die Txt Datei wird jede Sekunde vom UMG507 aktualisiert und zyklisch von Flash im Film aktualisiert.
Wichtig ist hierbei, dass der Internetbrowser nicht auf temporäre Seiten zugreift. Diese Einstellung kann meistens im Menü „Temporäre Internetseiten“ Ihres Internetbrowser überprüft werden.

Beispiel JAVA

Im folgendem Beispiel finden Sie den Quelltext inklusive Beschreibung eines JAVA MODBUS TCP/IP Clients. Der Client wurde für das UMG507E/EP geschrieben und getestet. Auf der CD-ROM PSWbasic/professional finden Sie die zum Beispiel passende CLASS Datei.

```
/*
 * MOD_TCPClient.java
 *
 * Created on 30. Juni 2003, 08:30
 */

package de.janitza.mod_tcp;

import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.InputStream;
import java.io.IOException;
import java.io.OutputStream;

import java.net.InetSocketAddress;
import java.net.Socket;

class TransactionIDWrong extends Exception{};

/** This class handles the communication with an device over MOD/TCP.
 * It only provides access to registers. So we are roughly Class 0 conform.
 * Typical usage:
 *    MOD_TCPClient modClient = new MOD_TCPClient();
 *    modClient.setUnitID(128);
 *    modClient.connect("192.168.2.214", 502);
 *    modClient.getRegisterAsFloat(0); // Voltage L1 on an UMG507
 *    // Possibly more calls here...
 *    modClient.close();
 * </code>
 */
public class MOD_TCPClient {

    /** Creates a new instance of MOD_TCPClient */
    public MOD_TCPClient() throws IOException {
        m_Socket = new Socket();
        m_Address = null;
        m_UnitID = 0;
        m_TransactionID = 0;
    }

    /** Connects to a MOD/TCP device.
     * @param host The host to connect to.
     * @param port The port to connect to. The specification says its always 502.
     * @throws IOException if an connection could not be estabilshed.
     */
    public void connect(String host, int port) throws IOException{
        m_Address = new InetSocketAddress(host, port);
        connect();
    }
```

```
/** Reconnects to a previosly connected device
     * @see #connect(String, int)
     * @throws IOException if an connection could not be estabilshed.
     */
    public void connect() throws IOException{
        if(m_Socket.isConnected()) {
            close();
        }
        m_Socket = new Socket();
        m_Socket.connect(m_Address);
        OutputStream os = m_Socket.getOutputStream();
        m_BufOutStream = new BufferedOutputStream(os);

        InputStream is = m_Socket.getInputStream();
        m_BufInStream = new BufferedInputStream(is);
    }

    /** Closes the socket and frees used resources.
     * @throws IOException if an connection could not be closed.
     */
    public void close() throws IOException{
        m_Socket.close();
        m_Socket = null;
    }

    /** Sets the unit identifier.
     * @param id The unit identifier.
     */
    public void setUnitID(byte id) {
        m_UnitID = id;
    }

    /** Gets the unit identifier.
     * @return The unit identifier previously set by setUnitID(int).
     */
    public byte getUnitID() {
        return m_UnitID;
    }

    /** Returns a given number of words of the specified register.
     * The words are returnd as bytes in a byte array.
     * @param refNr The register number.
     * @param wordCount the number of words(2 bytes) that should be requested.
     * @return An array of bytes with the size specified in the answer of the device.
     * @throws IOException if the connection is not open.
     */
    public byte[] getRegisterAsBytes(short refNr, short wordCount) throws IOException{
        byte[] sendBuf = new byte[5];
        byte[] result;
        byte[] recvBuf;
```

```
sendBuf[0] = 0x03;
        sendBuf[1] = (byte)((refNr & 0xff00) >> 8);
        sendBuf[2] = (byte)(refNr & 0xff);
        sendBuf[3] = (byte)((wordCount & 0xff00) >> 8);
        sendBuf[4] = (byte)(wordCount & 0xff);
       sendBytes(sendBuf);
        recvBuf = recvBytes();
        result = new byte[recvBuf[1]];
        for(int i = 0; i < recvBuf[1]; ++i) {
            result[i] = recvBuf[i + 2];
       }
       return result;
   }

   /** Returns a register as float.
    * Two words are requested. Since MOD/TCP is Bigendian we convert the value
    * into littleendian floats.
    * @param refNr The register to read from.
    * @return The float representing the register.
    * @throws IOException if the connection is not open.
    * @see #getRegisterAsBytes(short, short)
    */
   public float getRegisterAsFloat(short refNr) throws IOException {
       return bitsToFloat(getRegisterAsBytes(refNr, (short)2));
   }

         /** Converts Bigendian bytes to a float.
          * @param bytes The bigendian bytes.
          * @return The littleendian float.
          */
   private static float bitsToFloat(byte[] bytes) {
       float result = 0;
                int floatBits = bytes[3] & 0xff;
                floatBits += (bytes[2] & 0xff) << 8;
                floatBits += (bytes[1] & 0xff) << 16;
                floatBits += (bytes[0] & 0xff) << 24;
                result = Float.intBitsToFloat(floatBits);
       if((result < 1E-5) && (result > -1E-5))
           result = 0;
       return result;
   }


         /** Converts Bigendian bytes to a short.
          * @param bytes The bigendian bytes.
          * @return The littleendian short.
          */
   public static short bitsToShort(byte[] bytes) {
       short result = (short)(bytes[1] & 0xff);
       result += (short)((bytes[0] & 0xff) << 8);
       return result;
   }
   /** For internal use. Sends one datagram to a MOD/TCP device.
    */
   private void sendBytes(byte[] buf) throws IOException{
       byte[] sendBuf = new byte[7];
```

```
        sendBuf[0] = (byte)((m_TransactionID & 0xff00) >> 8);
        sendBuf[1] = (byte)(m_TransactionID & 0xff);
        sendBuf[2] = 0;
        sendBuf[3] = 0;
        sendBuf[4] = 0;
        sendBuf[5] = (byte)(buf.length + 1 & 0xff);
        sendBuf[6] = (byte)(m_UnitID & 0xff);
        m_BufOutStream.write(sendBuf);
        m_BufOutStream.write(buf);
        m_BufOutStream.flush();
    }

    /** For internal use. Receives one datagram.
     */
    private byte[] recvBytes() throws IOException {
        byte[] result;
        byte[] recvBuf = new byte[7];
        m_BufInStream.read(recvBuf);
        if( ! (m_TransactionID == bitsToShort(recvBuf))) {
//            throw new TransactionIDWrong();
        }
        ++m_TransactionID;
        result = new byte[recvBuf[5] -1];
        m_BufInStream.read(result);
        return result;
    }

    /** The socket used to connect to the device */
    private Socket m_Socket;
    /** The streams connected to the socket */
    private BufferedInputStream m_BufInStream;
    /** The streams connected to the socket */
    private BufferedOutputStream m_BufOutStream;
    /** The address to connect to */
    private InetSocketAddress m_Address;
    /** The unit identifier */
    private byte m_UnitID;
    /** The actual transaction ID */
    private short m_TransactionID;

        public static void main(String[] args) {
                try {
                        if(args.length < 1) {
                                System.err.println("Usage: java de.janitza.mod_tcp.MOD_TCPClient <host>");
                                System.exit(1);
                        }
                        MOD_TCPClient modClient = new MOD_TCPClient();
                        modClient.setUnitID((byte)128);
                        modClient.connect(args[0], 502);
                        System.err.println(modClient.getRegisterAsFloat((short)0)); // Voltage L1 on an UMG507
                        // Possibly more calls here...
                        modClient.close();
                } catch(Exception e) {
                        e.printStackTrace();
                }
        }
}
```