PLC SERVER Data output

Rev. 3

I. The form of the data output is determined in the PLC.properties file.

Please see more details in the PLC.properties description.

Parameter "data_output_type " can have following values: analog, tcp,udp

- "analog" means data is send to the FPGA board-> DAC and to the PLC received in the analog form (current of voltage)
- "tcp" means that the data is sent, using TCP data client, to an external TCP data server
- "udp" means that the data is sent, using UDP data client, to an external UDP data server

II. The format/content of the data sent is determined in PLC.properties file.

Following parameters are used only in the case of the TCP/UDP output (the are ignored in the case of the analog output)

The two parameters that determined the data output are "include_mse_data" (function 81) and "include_mse_exc_data" (function 82). Both of these parameters are false by default and output data is sent using function 80

a. function 80

Only measured values are sent using this function. If the data filter or MSE filter are enabled and the data rejected – no data is sent.

/80[message length][parameter 1][parameter 2]…[parameter n] [ 0xD,0xA]

where parameter 1,.. parameter n are measurement results (one measurement) – 4 bytes float each. Length is 1 byte

b. function 81

If value "include_mse_data"=true in the PLC.properties filer – function 81 output is used.

MSE data is included in the response when function #81 is used
Function #81 format:

/81[message length] [parameter 1][parameter 2]…[parameter n] [MSE] [ 0xD,0xA]

where parameter 1,.. parameter n are measurement results (one measurement) – 4 bytes float each and MSE is a 4 byte float, length 1 byte

If the data filter and/or MSE filter are enabled and data is rejected – no data is sent

c. function 82

If value "include_mse_exc_data"=true in the PLC.properties file, function 82 output is used.

Function 82 output includes warning and MSE data. The data is always sent even if results are rejected by the filter.

Function #82 is used with following format:

/82[message length- 1 byte] [warning 2 bytes][parameter 1][parameter 2]…[parameter n] [MSE] [ 0xD,0xA]

where parameter 1,.. parameter n are measurement results (one measurement) – 4

bytes float each and MSE is a 4 byte float.

warning – includes 2 bytes.

warning[0] is general system warning

NO_WARNING = 0;

LOW_SIGNAL = 1;

HIGH_SIGNAL = 2;

LOW_MEMORY = 3;

warning[1] is rejection warning

NO_WARNING = 0;

OUTLIER_DATA=50;

HIGH_MSE=51

Example of the reading and parsing data message by the data server.

```
package DataServer;


import java.io.BufferedInputStream;
import java.io.DataInputStream;
import java.io.EOFException;
import java.io.IOException;
import java.net.Socket;
import net.wimpi.modbus.io.BytesInputStream;
import org.apache.log4j.Category;
import tcp.net.ConstantsTCP;
```

```
/**
 *
 * @author leo
 */
public final class TCPServerTransport {

    Category m_cat = Category.getInstance(DataServer.TCPServerTransport.class);
    private DataInputStream m_Input;
    private BytesInputStream m_ByteIn;
    private Socket m_socket;
    public static final int MAX_MESSAGE_LENGTH = 1600;
    public static final byte MESSAGE_START = 0x2F; // back slash '/'
   // public static final byte STAR = 0X2A;
    public static final byte[] IN_MESSAGE_END = {0xD, 0xA};
    private static long st_time;
    /* \n\r i.e. CR+LF */

 public TCPServerTransport(Socket soc) throws IOException {

        setSocket(soc);
    }
    public void setSocket(Socket socket) throws IOException {
        m_socket = socket;
        prepareStreams();
    }//setSocket

    private void prepareStreams() throws IOException {
        m_Input = new DataInputStream(
                new BufferedInputStream(m_socket.getInputStream(), MAX_MESSAGE_LENGTH));
        m_ByteIn = new BytesInputStream(MAX_MESSAGE_LENGTH);
    }//prepareStreams
```

```
public boolean readData() throws IOException {
    boolean ret = true;
    synchronized (m_ByteIn) {
        byte[] buffer = m_ByteIn.getBuffer();
        if (m_Input.read(buffer, 0, 2) == -1) {
            throw new EOFException("Premature end of stream (Header truncated).");
        }
        // check if message is valid
        if (buffer[0] == MESSAGE_START) {
            byte funcCode = buffer[1];
            m_Input.read(buffer, 2, 1);
            int length = buffer[2];
            //m_cat.info("Message length="+String.valueOf(length));
            if (m_Input.read(buffer, 3, length) == -1) {
                throw new IOException("TCPTransport:readRequest:Premature end of the
               message: length=" + String.valueOf(length));
            }
            m_ByteIn.reset();
            if(st_time==0){
                st_time=System.currentTimeMillis();
            }else{
              long  en_time=System.currentTimeMillis();
                m_cat.info("time interval="+(en_time-st_time));
                st_time=en_time;
            }
            parseData(funcCode, length);
        }

    }
    return ret;
}
```

```
private void parseData(byte func, int length) throws IOException {
    m_ByteIn.skip(3);
    switch (func) {
        case ConstantsTCP.SEND_DATA_TCP: {  //function 80
            // func=80
            /*
            check the number of measured parameters
            */
            int numParam = length / 4;
            if (numParam == 1) {
                float param = m_ByteIn.readFloat();
                m_cat.info("function=" + func + ": param=" + param);
            } else {
                float param;
                for (int i = 0; i < numParam; i++) {
                    param = m_ByteIn.readFloat();
                    m_cat.info("function=" + func + ": param" + i + "=" + param);
                }
            }

            break;
        }
        case ConstantsTCP.SEND_MSE_TCP: { //function 81
            //func=81
           int numParam=(length-4)/4;
           if(numParam==1){
               float param = m_ByteIn.readFloat();
               float mse=m_ByteIn.readFloat();
               m_cat.info("function=" + func + ": param=" + param+" mse="+mse);
           }else{
               float param;
               for(int i=0;i<numParam;i++){
                   param = m_ByteIn.readFloat();
```

```
            m_cat.info("function=" + func + ": param"+i+"=" + param);
        }
            float mse=m_ByteIn.readFloat();
            m_cat.info("mse="+mse);
    }

    break;
}
case ConstantsTCP.SEND_MSE_EXC_TCP: { // function 82
    // func=82
    byte warn_sys=m_ByteIn.readByte();
    byte warn_status=m_ByteIn.readByte();
    byte warn_1= m_ByteIn.readByte();
    byte warn_2= m_ByteIn.readByte();
     m_cat.info("function=" + func);
     m_cat.info("Warning system="+warn_sys);
     m_cat.info("Warning status="+warn_status);
     m_cat.info("Warning High_MSE="+warn_1);
     m_cat.info("Warning OUTLIER="+warn_2);
     int numParam=(length-8)/4;
     if(numParam==1){
         float param=m_ByteIn.readFloat();
         float mse=m_ByteIn.readFloat();
         m_cat.info( "param=" + param+" mse="+mse);
     }else{
         float param;
         for(int i=0;i<numParam;i++){
             param=m_ByteIn.readFloat();
             m_cat.info( "param"+i+"=" + param);
         }
         float mse=m_ByteIn.readFloat();
         m_cat.info( "MAE"+"=" + mse);
     }
```

```
                break;
            }
            case ConstantsTCP.SEND_HELLO_DATA_SERVER: { // function 83
                // func=83
                byte[] message = new byte[length];
                m_ByteIn.readFully(message);
                String messageStr = new String(message);
                m_cat.info(messageStr);
                break;
            }
            default: {
                  m_cat.warn("Invalid Function="+func);
            }
        }
   }
    public void close() {
        try {
            if (m_socket != null && !m_socket.isClosed()) {
                m_socket.close();
            }
        } catch (IOException ex) {
        }
    }
}
```

Output of of the data server for function 82

```
2024-04-07 16:04:01,349 [Thread-1] INFO  DataServer.TCPServerTransport - function=82
2024-04-07 16:04:01,352 [Thread-1] INFO  DataServer.TCPServerTransport - Warning system=0
2024-04-07 16:04:01,352 [Thread-1] INFO  DataServer.TCPServerTransport - Warning status=0
2024-04-07 16:04:01,353 [Thread-1] INFO  DataServer.TCPServerTransport - Warning High_MSE=0
2024-04-07 16:04:01,353 [Thread-1] INFO  DataServer.TCPServerTransport - Warning OUTLIER=0
2024-04-07 16:04:01,354 [Thread-1] INFO  DataServer.TCPServerTransport - param=194.61903
mse=1.3502303
2024-04-07 16:04:02,359 [Thread-1] INFO  DataServer.TCPServerTransport - time interval=1012
```