

**Dokumentation IPA**  
 EOL Kapazitiver Tankfüllstandsgeber

Auszubildender Informatiker Fachrichtung Applikationsentwicklung Hallauer Joel

# Inhaltsverzeichnis

[1 Inhaltsverzeichnis 1](#_Toc36620707)

[2 Glossar 3](#_Toc36620708)

[3 Vorwort 4](#_Toc36620709)

[3.1 Zweck des Dokuments 4](#_Toc36620710)

[3.2 Allgemeines 4](#_Toc36620711)

[3.2.1 Verwendete Tools 4](#_Toc36620712)

[3.2.2 Vorgehensmodell 4](#_Toc36620713)

[4 Informieren 5](#_Toc36620714)

[4.1 Ausgangslage 5](#_Toc36620715)

[4.1.1 Ist-Stand 5](#_Toc36620716)

[4.1.2 Soll-Stand 5](#_Toc36620717)

[4.1.3 Anforderungen 6](#_Toc36620718)

[4.2 Ziele 6](#_Toc36620719)

[4.2.1 Muss-Ziele 6](#_Toc36620720)

[4.2.2 Kann-Ziele 6](#_Toc36620721)

[5 Planen 7](#_Toc36620722)

[5.1 Terminplan 7](#_Toc36620723)

[5.1.1 Terminplan Grob 7](#_Toc36620724)

[5.1.2 Terminplan Detailiert 7](#_Toc36620725)

[5.2 Meilensteine 9](#_Toc36620726)

[5.3 Ablauf EOL 10](#_Toc36620727)

[5.4 Vordefinierte Testfälle 11](#_Toc36620728)

[5.4.1 Kommunikation 11](#_Toc36620729)

[5.4.2 EOL 11](#_Toc36620730)

[5.5 Prüfplan 12](#_Toc36620731)

[5.5.1 Testing aller Tests 12](#_Toc36620732)

[5.5.2 Einzeltest 12](#_Toc36620733)

[6 Entscheiden 13](#_Toc36620734)

[7 Realisieren 14](#_Toc36620735)

[7.1 Kommunikation 14](#_Toc36620736)

[7.1.1 Implementierung 14](#_Toc36620737)

[7.2 EOL 16](#_Toc36620738)

[8 Kontrollieren 17](#_Toc36620739)

[8.1 Prüffälle 17](#_Toc36620740)

[8.1.1 T001 - Auf Prüfdatei warten 17](#_Toc36620741)

[9 Auswerten 18](#_Toc36620742)

[9.1 Reflexion 18](#_Toc36620743)

[9.1.1 Allgemein 18](#_Toc36620744)

[10 Abbildungsverzeichnis 19](#_Toc36620745)

# Glossar

|  |  |
| --- | --- |
| Fachwort | Erklärung |
| DLL | Dynamic Link Library:  Dynamische Programmbibliothek die Code und Daten enthält. |
| DUT | Device Under Test: Prüfling |
| EOL | „End of Line“ Endkontroll Software, welche ein Gerät auf seine Korrektheit in jeder Hinsicht prüft. |
| \*.gui Datei | Userinterface Datei welche von der verwendeten Entwicklungsumgebung CVI verwendet wird. |
| NMEA2000 | NMEA2000 ist ein Netzwerktyp, welcher von der veratron AG zur Datenübertragung in verschiedenen Boots-komponenten eingesetzt wird. |
| PTS | Datenblatt |
| State-maschine | Eine State Maschine hat eine bestimmte Anzahl von definierten Zuständen. Durch das setzen eines Zustandes führ die Applikation den für diesen Zustand vorgesehen Code aus. |

# Vorwort

## Zweck des Dokuments

Dieses Dokument dient dazu, die Schritte in der praktischen Arbeit von Joel Hallauer nachvollziehbar darzulegen. Es werden alle Arbeitsschritte sowie Gedankengänge zur Arbeit «EOL für Kapazitiver Tankfüllstandsgeber» dokumentiert und beschrieben.

## Allgemeines

### Verwendete Tools

|  |  |
| --- | --- |
| Tool | Verwendung |
| SlickEdit Pro (v 19.0.11) | Codeing, Codegestaltung |
| CVI Labwindows (v 15.0.1) | Kompilieren des Codes, GUI Gestaltung |
| Notepad++ (v 7.5.9) | Temporäre Notizen |
| GanttProject (v 2.8.5) | Terminplan |
| Microsoft Word | Dokumentation und Aufgabenstellung |
| Github |  |

### Vorgehensmodell

Für die Arbeit zur EOL für einen Kapazitiven Tankfüllstandsgeber wird das Vorgehensmodell IPERKA verwendet.

**Informieren:**Was für Ziele wurden definiert?  
Wie sieht die Ausgangslage aus?  
Welche Mittel stehen zur Verfügung?

**Planen:**Wann sollen die Arbeiten erledigt werden (Terminplan/Meilensteine)?  
Was muss getestet werden?

**Entscheiden:**Ablauf der EOL bestimmen

**Realisieren:**Programmierung der KommunikationProgrammierung der EOL

**Kontrollieren:**Testen der Kommunikation  
Testen der EOL

**Auswerten:**Reflexion

# Informieren

## Ausgangslage

### Ist-Stand

Aktuell besteht bereits eine EOL-Vorrichtung für analoge Tankfüllstandgeber. Jedoch ist diese Vorrichtung Steuerungs- und Softwaremässig veraltet. Für die Steuerung sind bereits keine Ersatzteile mehr erhältlich. Auf der EOL-Vorrichtung können lediglich zwei analoge Tankfüllstandsgeber getestet werden:

* **Tankfüllstandsgeber Source-Mode:**  
  liefert je nach Füllstand des Tankes, eine Spannung, welche vom Anzeigeinstrument ausgewertet wird und dann entsprechend anzeigt
* A close up of a blue wall

  Description automatically generatedA picture containing indoor, table, desk, sitting

  Description automatically generated**Tankfüllstandsgeber Sink-Mode:**  
  verbraucht je nach Füllstand des Tankes, einen Strom. Dieser Stromverbrauch wird vom Anzeigeinstrument ausgewertet und entsprechend angezeigt

Abbildung 1 kapazitiver Tankfüllstandsgeber

Abbildung 2 Vorrichtung EOL kapazitiver Tankfüllstandsgeber

### Soll-Stand

Die bestehende EOL-Vorrichung soll Hard- und Softwaremässig auf einen aktuellen Stand gebracht werden. Neu soll neben den analogen Tankfüllstandgebern auch ein digitaler Tankfüllstandgeber getestet werden können. Dieser Kommuniziert neu über NMEA2000 wodurch diese neu implementiert werden muss. Die Überarbeitung der Hardware ist nicht Bestandteil dieser IPA.

### Anforderungen

* Die Steuersoftware für die Endkontrolle „kapazitiver Tankfüllstandsgeber“ muss neu mit aktuellerer Technik (Windows, C, Ethernet, ...) erstellt werden.
* Der Ablauf der neuen Software muss gemäss der Prüfspezifikation (Dokument 10371354.pdf) erstellt werden.
* Die NMEA2000 Kommunikation muss für das Produkt angepasst werden (TPL\_PTS\_Z10006660\_AA.pdf).

## Ziele

### Muss-Ziele

|  |  |
| --- | --- |
| Ziel | Kriterium |
| Vorrichtung über GUI steuerbar | Ablauf muss über GUI gestartet werden können.  Programmzustand muss über GUI ersichtlich sein. |
| Die Artikeldefinitionen aus Ini-Datei laden | Alle Artikel aus bestehender Ini-Datei fehlerfrei einlesen |
| Zu prüfender Artikel aus Liste auswählbar | Alle Artikel aus Ini-Datei werden zur Auswahl aufgelistet |
| Analoge kapazitive Tankfüllstandsgeber prüfbar | Analoge kapazitive Tankfüllstandsgeber müssen geprüft werden können, gemäss Pruefvorschr\_98601240\_.pdf (Ablauf) Kapitel 4.3 |

### Kann-Ziele

|  |  |
| --- | --- |
| Ziel | Kriterium |
| Diagnosepanel Inputs und Outputs der Vorrichtung | Alle Ein-/Ausgänge beobachtbar/steuerbar |
| Digitale kapazitive Tankfüllstandsgeber prüfbar | Digitale kapazitive Tankfüllstandsgeber müssen geprüft werden können, gemäss Pruefvorschr\_98601240\_.pdf (Ablauf) Kapitel 4.3 und TPL\_PTS\_Z10006660\_AA.pdf (Kommunikation) |

# Planen

## Terminplan

### Terminplan Grob

Im Groben Terminplan ist zu erkennen, an welchen Tagen vorgesehen wird, welche Arbeit zu verrichten.



Abbildung 3 Terminplan Text

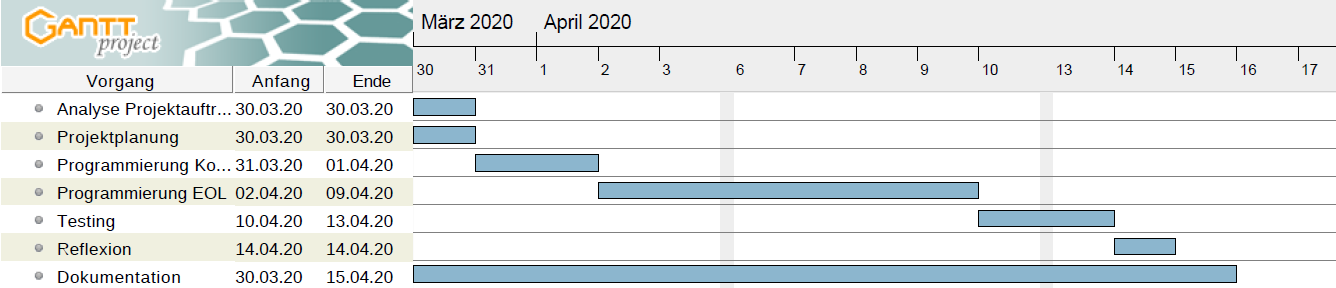


Abbildung 4 Terminplan Grafisch

### Terminplan Detailiert

Der Detaillierte Terminplan dient dazu, die Stunden, welche für eine Arbeit benötigt werden zu planen.

|  |  |
| --- | --- |
| Arbeit | Geplante Zeit [h] |
| Informieren |  |
| * Analyse des Projektauftrages | 0.5 |
| * Ist Stand Beschreiben | 0.5 |
| * Soll Stand beschreiben | 0.25 |
| * Muss- und Kann-Ziele beschreiben und definieren | 0.25 |
| Planen | 1.5 |
| * Terminplan Grob | 0.25 |
| * Terminplan detailliert | 1 |
| * Meilensteinplan | 0.5 |
| * Ablauf der EOL | 1.5 |
| * Definition de Testfälle | 1 |
| * Definition Prüfplan | 0.5 |
| Entscheiden |  |
| * Allfällige Entscheidungen | 1 |
| Realisieren |  |
| * Implementation der Kommunikation   + Analoge Inputs   + Digitale Inputs   + Digitale Outputs | 10 |
| * Implementation der EOL   + Kommunikation mit EA-Modul   + Kommunikation mit DUT   + Prüfungsablauf | 38 |
| Kontrollieren |  |
| * Prüfen + Massnahmen Kommunikation | 4 |
| * Prüfen + Massnahmen EOL | 4 |
| * Soll- Ist Stand überprüfen (Zeit nach Terminplan) | 2 |
| Auswerten |  |
| * Reflexion der Arbeit | 4 |
| Dokumentation | 20 |
| Total: | 89.25 |

## Meilensteine

|  |  |  |
| --- | --- | --- |
| Datum | Meilenstein | Zuweisung |
| 30.03.2013 | Projektstart | HaJ |
| 30.03.2013 | Projektplan | HaJ |
| 02.04.2013 | Kommunikation implementiert | HaJ |
| 10.04.2013 | EOL Implementiert | HaJ |
| 14.04.2013 | Getestet | HaJ |
| 15.04.2013 | Reflektiert und kontrolliert | HaJ |
| 15.04.2013 | Projektende | HaJ |

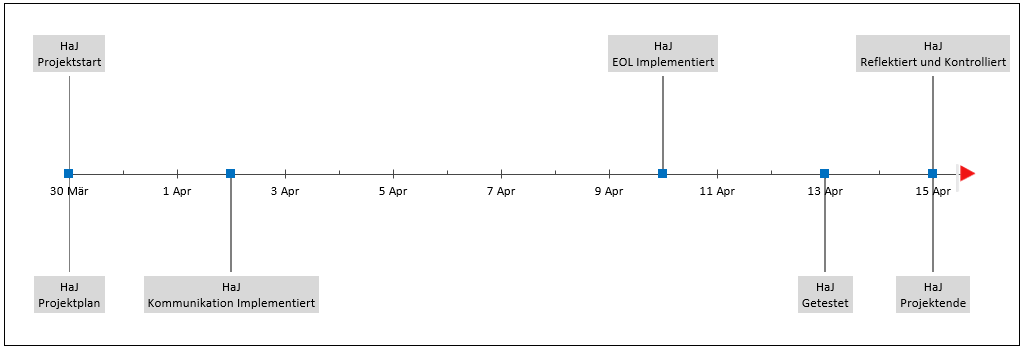


Abbildung 5 Meilenstein Planung

## Ablauf EOL

Die Prüfung eines Tankfüllstandgebers durch die EOL soll wie folgt ablaufen:

1. Nachdem der DUT an die Vorrichtung angeschlossen wurde, sendet die EOL einen Login Command an den Prüfling, um mit ihm kommunizieren zu können.
2. Nun wird der Wasserstand in den Tanks so lange erhöht, bis er laut dem Referenzgerät 100 % erreicht hat.
3. Somit kann nun der Wert vom Prüfling abgelesen werden.
4. Die Ausgelesenen Werte können nun miteinander verglichen werden. Stimmen die Werte mit der Beachtung der Toleranz überein, so kann der Test weitergeführt werden. Andernfalls wird eine Fehlermeldung angezeigt und die Prüfung abgebrochen, da der Prüfling fehlerhaft ist.
5. Nun senkt man den Wasserstand wieder, bis das Referenzgerät 50 % anzeigt.
6. Ist der Wasserstand nun bei 50 % kann der Wert beim Prüfling ausgelesen werden.
7. Wenn dir Wert des DUT’s und des Referenzgerätes übereinstimmen, so wird der Wasserstand ein letztes Mal auf 0% gesenkt.
8. Die Werte aus dem Referenzgerät und dem Prüfling werden erneut verglichen.
9. Wenn die Werte übereinstimmen, so wird der geprüfte Tankfüllstandsgeber als «in Ordnung» angesehen und die Prüfung wird abgeschlossen. Stimmen die Werte jedoch nicht überein, wird eine Ausgabe erfolgen, welche über den Fehler berichtet und die Prüfung wird abgebrochen.

Auf der Vorrichtung ist genug Platz, um bis zu Zehn Prüflinge zur selben Zeit anzuschliessen. Jedoch kann nur das Signal eines Prüflings zur gleichen Zeit ausgelesen werden.

Mit einem Potentiometer an der Vorrichtung ist es jedoch möglich, den DUT auszuwählen, von welchem man aktuell die Daten auslesen möchte.

Um Zeit zu sparen, wird nun nachdem man den Wasserspiegel auf den richtigen Wert erhöht oder gesenkt hat, alle Zehn Prüflinge prüfen indem man mit dem Potentiometer das Signal umschaltet. Erst wenn von allen Prüflingen der Wert eingelesen und überprüft wurde, fährt die EOL im Test weiter.

## Vordefinierte Testfälle

Die verschiedenen Testfälle werden im Vorhinein definiert. Die einzelnen Tests werden nach dem ebenfalls im Vorhinein ausgearbeiteten Prüfplan durchgeführt. Sobald alle Tests durchlaufen wurden, werden wenn nötig Massnahmen definiert und ergriffen.

### Kommunikation

|  |  |  |
| --- | --- | --- |
| ID | Test | Erwartetes Resultat |
| TK01 | Lesen der Analogen Inputs | Die Analogen Inputs können korrekt vom Tankfüllstandsgeber eingelesen werden. |
| TK02 | Lesen der Digitalen Inputs | Die Digitalen Inputs können korrekt vom Tankfüllstandsgeber eingelesen werden. |
| TK03 | Schreibend er Digitalen Outpus | Die Digitalen Outputs können korrekt am Tankfüllstandsgeber gesetzt werden. |
| TK04 | Lesen Software Version | Die Software Version soll korrekt eingelesen werden können. |
| TK05 | Schreiben der B-Nummer | Die B-Nummer soll korrekt geschrieben werden können. |

### EOL

|  |  |  |
| --- | --- | --- |
| ID | Test | Erwartetes Resultat |
| TE01 | Wasser füllen | Durch das Schalten eines Ventiles über da EA-Modul wird Luft in den Wassertank gepumpt. Durch den Steigenden Druck im Tank steigt das Wasser die dafür vorgesehenen Rohre hinauf. |
| TE02 | Wasser halten | Durch das Schalten eines weiteren Ventils über das EA-Moduls wird das Wasser in den Rohren gehalten, selbst wenn der Luftdruck im Wassertank nachlässt. |
| TE03 | Zu wenig / Zu viel Wasser | Ist zu wenig oder zu viel Wasser in die Rohre gepumpt worden, so soll dies vom Referenzgeber erkannt werden. Entsprechen soll auch eine Benachrichtigung an den Benutzer stattfinden. |
| TE04 | Entfernen des DUT | Entfernt man einen DUT bevor die Prüfung abgeschlossen ist, so soll dies von der EOL erkannt und die Prüfung gestoppt werden. |
| TE05 | Korrektes Darstellen der ausgelesenen Werte | Die von den Tankfüllstandsgebern ausgelesenen Werte sollen korrekt dargestellt werden. Es soll auf einen Blick erkennbar sein, ob sich der gelesene Wert im Toleranzbereich befindet. |

## Prüfplan

### Testing aller Tests

Um sicherzustellen, dass alle Tests erfolgreich sind, wurde ein Testkonzept entwickelt, damit am Ende die komplette Applikation funktioniert. Die Folgende Grafik soll das Testkonzept grafisch darstellen:



Abbildung 6 Prüfplan gesamt

##### Beschreibung

Begonnen wird mit dem ersten Testfall. Falls dieser ohne Probleme und Bugs erfüllt ist, geht man zum nächsten Testfall und testet diesen. Dies macht man so lange, bis man einen Fehler findet. In diesem Fall löst man das Problem sofort und fokussiert sich darauf, den Bug zu fixen. Sobald auch dieser Testfall erfolgreich ist, fängt man wieder mit dem ersten Testfall an und geht nochmal alle genau durch. Dies hat den Vorteil, dass falls die Änderung im Programm einen Einfluss auf früheren Test hat, man diese findet und ebenfalls beheben kann. Man macht dieses Spiel nun so lange, bis alle Bugs behoben wurden und man sagen kann das alle Tests erfolgreich erfüllt wurden.

### Einzeltest



Abbildung 7 Prüfplan einzeln

Die einzelnen Anforderungen werden nach dem Blackbox Konzept getestet. Ein Black Box Verfahren ist ein spezifikationsorientiertes Testentwurfsverfahren. Dabei werden die Testfälle ausschließlich aus der Spezifikation des zu testenden Objekts abgeleitet, ohne dabei dessen innere Struktur, den Code, zu berücksichtigen. Es wird also nur das von aussen sichtbaren Verhalten des Testobjektes beobachtet.

Bei jedem Test werden die erwarteten Resultate mit dem tatsächlichen Resultat vergleichen, Unterschiede festgehalten und gegebenenfalls Massnahmen definiert.

# Entscheiden

# Realisieren

## Kommunikation DLL

Für die Implementierung der Kommunikation wurde die PTS «TPL\_PTS\_Z10006660\_AA.pdf» herangezogen. In diesem Datenblatt sind alle Telegramme beschrieben, welche zum Anfordern eines Wertes benötigt werden. Darin enthalten sind alle digitalen und analogen Ein- und Ausgänge, sowie Geräteinformationen wie die Software Version.

Die Kommunikation zum Tankfüllstandsgeber wird nach erfolgreicher Implementierung als dll kompiliert, welche dann für die EOL verwendet werden kann.

### Implementierung

Zur Implementierung der Kommunikation über NMEA2000 mit dem Kapazitivem Tankgeber, wurde eine Vorlage herangezogen.

#### Analoge Inputs

Aus dem Kapazitivem Tankfüllstandsgeber lassen sich drei Analoge Inputs auslesen. Diese Inputs können alle aus einem Telegramm ausgelesen werden, wobei jedoch nur zwei davon für die EOL relevant sind:

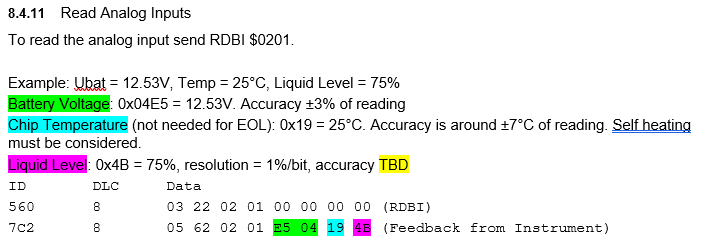


Abbildung 8 PTS Analoge Inputs

Um die Gewünschten Inputs lesen zu können, muss zuerst ein Request Telegramm über den CAN-Bus an den Tankgeber gesendet werden. Das Request Telegramm besteht aus 8 Bytes und muss wie folgt aufgebaut sein:

|  |  |  |
| --- | --- | --- |
| Byte | Wert | Beschreibung |
| [0]  [1] | [0x03]  [0x22] | Identifikationsnummer, dass ein Wert vom Tankfüllstandsgeber angefordert wird. |
| [2]  [3] | [0x02]  [0x01] | Identifikationsnummer, dass die analogen Inputs des Tankfüllstandgebers anfordert werden. |
| [4] – [7] | [0x00] | Die restlichen vier Bytes können auf dem Wert 0x00 belassen werden, da sie für das Anfordern der Daten nicht relevant sind. |

Nach einem erfolgreich gesendeten Request-Telegramm wird der Tankfüllstandsgeber mit einem Telegramm antworten, welche die angeforderten Daten enthält:

|  |  |  |
| --- | --- | --- |
| Byte | Wert | Beschreibung |
| [0]  [1] | [0x05]  [0x62] | Identifikationsnummer, dass es sich um ein Telegramm vom Tankfüllstandsgeber handelt. |
| [2]  [3] | [0x02]  [0x01] | Identifikationsnummer, dass es sich bei den folgenden Daten um die Werte der Analogen Inputs handelt. |
| [4]  [5] | Bsp: [0xE5]  Bsp: [0x04] | Wert Analoger Input: Battery Voltage |
| [6] | Bsp: [0x19] | Wert Analoger Input: Chip Temperature |
| [7] | Bsp: [0x4B] | Wert Analoger Input: Liquid Level |

#### Digitale Inputs

Für die Digitalen Inputs eines Tankfüllstandgebers wird nur ein Telegramm benötigt. Darin enthalten sind die Werte drei verschiedener digitalen Inputs wobei für die EOL jedoch nur die DIP Switches benötigt werden.

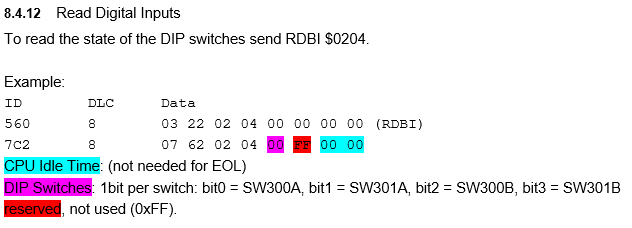


Abbildung 9 PTS Digitale Inputs

Wie auch bei den Analogen Inputs, muss, um die digitalen Inputs lesen zu können, zuerst ein Request-Telegramm an den Tankgeber gesendet werden:

|  |  |  |
| --- | --- | --- |
| Byte | Wert | Beschreibung |
| [0]  [1] | [0x03]  [0x22] | Identifikationsnummer, dass ein Wert vom Tankfüllstandsgeber angefordert wird. |
| [2]  [3] | [0x02]  [0x01] | Identifikationsnummer, dass die digitalen Inputs des Tankfüllstandgebers anfordert werden. |
| [4] – [7] | [0x00] | Die restlichen vier Bytes können auf dem Wert 0x00 belassen werden, da sie für das Anfordern der Daten nicht relevant sind. |

Nach einem erfolgreich gesendeten Request-Telegramm wird der Tankfüllstandsgeber mit einem Telegramm antworten, welche die angeforderten Daten enthält:

|  |  |  |
| --- | --- | --- |
| Byte | Wert | Beschreibung |
| [0]  [1] | [0x05]  [0x62] | Identifikationsnummer, dass es sich um ein Telegramm vom Tankfüllstandsgeber handelt. |
| [2]  [3] | [0x02]  [0x01] | Identifikationsnummer, dass es sich bei den folgenden Daten um die Werte der digitalen Inputs handelt. |
| [4] | Bsp: [0xE5] | Wert digitaler Input: DIP Switches |
| [6] | Bsp: [0x19] | Wert digitaler Input: reserved |
| [7] | Bsp: [0x4B] | Wert digitaler Input: CPU Idle Time |

#### Digitale Outputs

Da der kapazitive Tankgeber nur einen digitalen Output hat benötigt man lediglich ein Telegramm. Beim Digitalem Output handelt es sich um eine LED, welches sich auf der Leiterplatte einer Tankfüllstandgebers befindet. Diese LED kann mit folgendem Telegramm ein oder ausgeschalten werden:

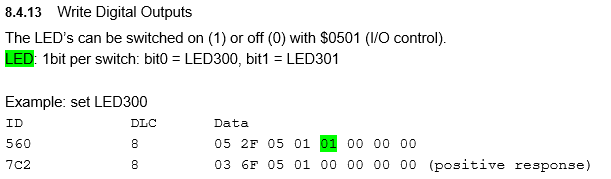


Abbildung 10 PTS Digitale Outputs

Um einen digitalen Output zu setzen, muss man lediglich ein Telegramm an den Tankgeber senden, welches dii korrekten Daten über den Zustand der Outputs enthält:

|  |  |  |
| --- | --- | --- |
| Byte | Wert | Beschreibung |
| [0]  [1] | [0x05]  [0x2F] | Identifikationsnummer, dass man ein Wert im Tankfüllstandsgeber schreiben möchte. |
| [2]  [3] | [0x05]  [0x01] | Identifikationsnummer, dass es sich bei den zu schreibenden Werten, um digitale Outputs handelt. |
| [4] | [0x00]  oder  [0x01] | Das vierte Byte wird dazu verwendet, die LED auf der Leiterplatte des Tankfüllstandgebers entweder ein- oder auszuschalten.  0x00 🡪 Ausschalten  0x01 🡪 Einschalten |

Nachdem das Telegramm mit den Werten der Zustände der digitalen Outputs an den Tankgeber gesendet wurde, erhält man eine Antwort, welche Bescheid gibt, dass das Setzen der Outputs erfolgreich war:

|  |  |  |
| --- | --- | --- |
| Byte | Wert | Beschreibung |
| [0]  [1] | [0x03]  [0x6F] | Identifikationsnummer, dass es sich um ein Telegramm vom Tankfüllstandsgeber handelt. |
| [2]  [3] | [0x05]  [0x01] | Identifikationsnummer, dass es sich beim Telegramm um die Antwort auf das Setzen der digitalen Outputs handelt. |
| [4] – [7] | [0x00] | Wenn die digitalen Outputs ohne Probleme gesetzt werden konnten, haben die Bytes 4 – 7 den Wert 0x00. |

## Tankfüllstandsgeber EOL

### Enviroment

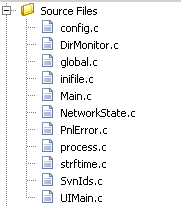
Um die Applikation umzusetzen wird die ANSI-C-Entwicklungsumgebung CVI-Labwindows (C for Virtual Instrumentation) verwendet. Für die Programmierung der Software wird die CVI Version 7.1 auf einem Virtuellen Rechner mit dem Betriebssystem Windows XP verwendet.

Abbildung 6 Dateistruktur

Als Ursprung für die Tankfüllstandsgeber EOL Software wird die „LinkUp-EOL“ verwendet. Die nötige Dateistruktur ist aufgrund dessen bereits vorhanden und muss so nur angepasst werden.

Hier nun eine Auflistung der angepassten Dateistruktur für die Tankfüllstandsgeber EOL:

|  |  |
| --- | --- |
| Datei | Funktion |
| Config.c | In der config.c Datei werden Informationen für die Applikation gespeichert, welche während der Betriebszeit geändert werden können. So ist es möglich bestimmte Einstellung ohne neue Kompilation des Programmes zu ändern und anzupassen. |
| Global.c | Die gobal.c Datei ist für globale Deklarationen gedacht. In dieser Datei findet man globale Strukturen, welche über das gesamte Projekt hinweg benötigt werden. |
| Inifile.c | Die Inifile.c Datei bietet alle Funktionen, welche benötigt werden, um mit einer .ini Datei zu operieren. |
| Main.c | In der Main.c Datei befindet sich die main() Funktion womit die Applikation in dieser Datei startet. Hier werden zudem alle benötigten Initialisationen vorgenommen. |
| NetworkState.c | Die NetworkState.c zeigt den Status der Netzwerkverbindung an. |
| PnlError.c | Zeigt ein Panel mit den aktuellen Fehlern an. |
| Process.c | In der process.c Datei befindet sich der Ablauf der Applikation. Dies geschieht durch eine State-maschine. |
| Strftime.c | Die Strftime.c bietet eine Funktion, welche wie die Funktion sprintf() operiert. Sie gibt die Zeit und das Datum zurück. |
| UiMain.c | In der UiMain.c Datei sind alle Funktionen, welche mit dem Userinterface interagieren deklariert. |

TODO: Anpassen auf Tankgeber EOL

### LibTesterGui Code generator

Um den Ablauf der EOL grafisch darzustellen wird die Applikation «LibTesterGui\_Test» zur Hand genommen.

### EA-Modul

TODO: Zol.h Änderungen beschreiben

### Ablauf - State Maschine

# Kontrollieren

## Prüffälle

Beim Prüfen wird strikt nach dem definierten Prüfplan vorgegangen und es werden die im Voraus definierten Testfälle abgehandelt.

### T001 - Auf Prüfdatei warten

**Beschreibung**

**Erwartetes Ergebnis**

**Test**

|  |  |  |
| --- | --- | --- |
| Test | Ergebnis | Symbol |
|  |  |  |
|  |  |  |

**Auswertung / Massnahmen**

# Auswerten

## Reflexion

### Allgemein

### Gelerntes

#### Dokumentation

#### Funktionen

#### Reibungspunkte

# Abbildungsverzeichnis

[Abbildung 1 kapazitiver Tankfüllstandsgeber 5](file:///E:\02_Arbeit\Veratron\IPA\05_Doku\DokumentationIPA.docx#_Toc36556846)

[Abbildung 2 Vorrichtung EOL kapazitiver Tankfüllstandsgeber 5](file:///E:\02_Arbeit\Veratron\IPA\05_Doku\DokumentationIPA.docx#_Toc36556847)

[Abbildung 3 Terminplan Text 7](#_Toc36556848)

[Abbildung 4 Terminplan Grafisch 7](#_Toc36556849)

[Abbildung 5 Meilenstein Planung 9](#_Toc36556850)

[Abbildung 6 Prüfplan gesamt 12](#_Toc36556851)

[Abbildung 7 Prüfplan einzeln 12](#_Toc36556852)

[Abbildung 8 PTS Analoge Inputs 14](#_Toc36556853)