# IOWarrior Socket-Server

# Version 1-4-005

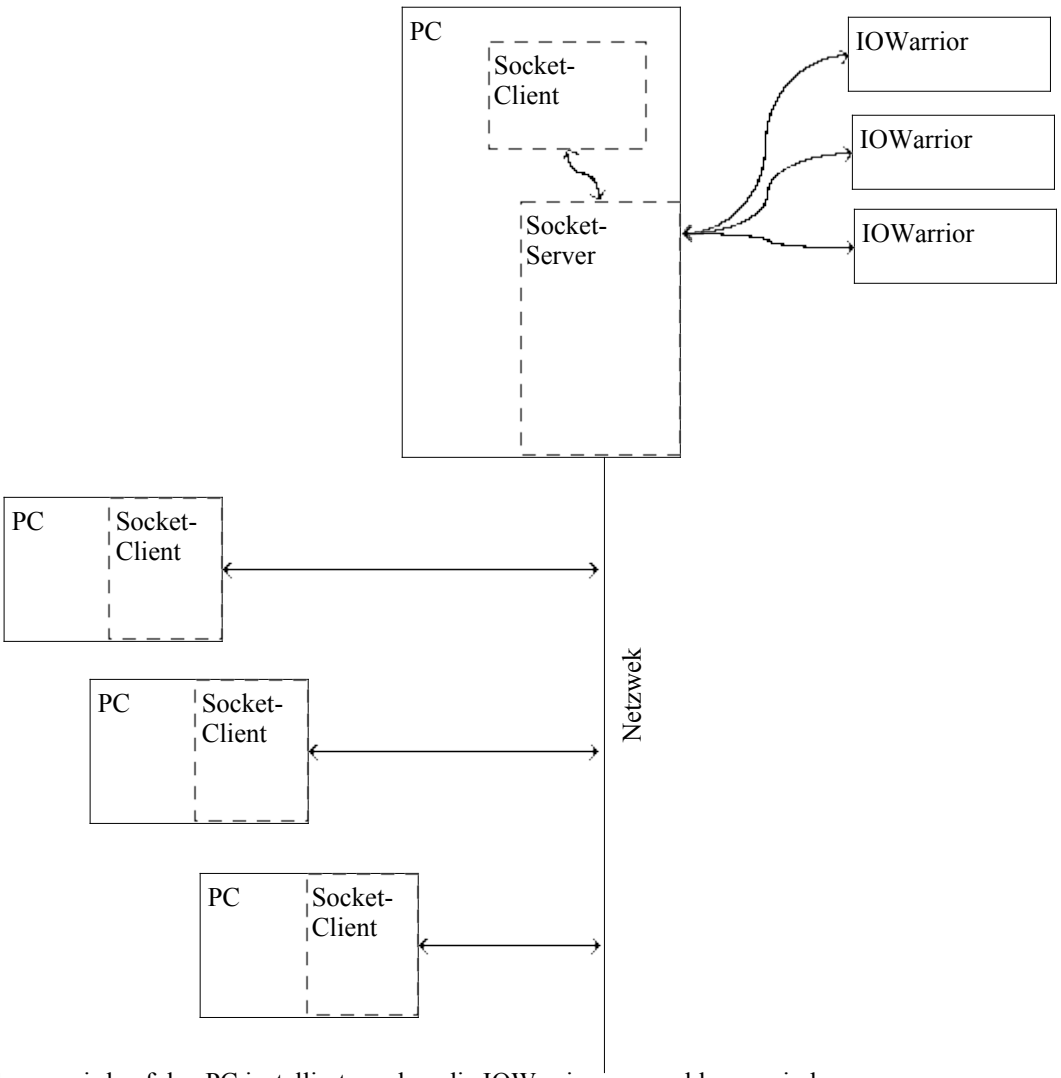
Ralf Greinert
www.greinert-dud.de

# Inhaltsverzeichnis

# Inhaltsverzeichnis

# Inhaltsverzeichnis

# A Funktionsweise




Der Socket-Server wird auf den PC installiert, an dem die IOWarriors angeschlossen sind.
Um eine sichere Funktion zu gewährleisten, dürfen auf diesem PC keine weiteren Programme auf die IOW-DLL zugreifen.
Der Socket-Server ist eine Win32-Applikation und wurde unter Win2000 , W2000-Server , XP ,Server2003 getestet.

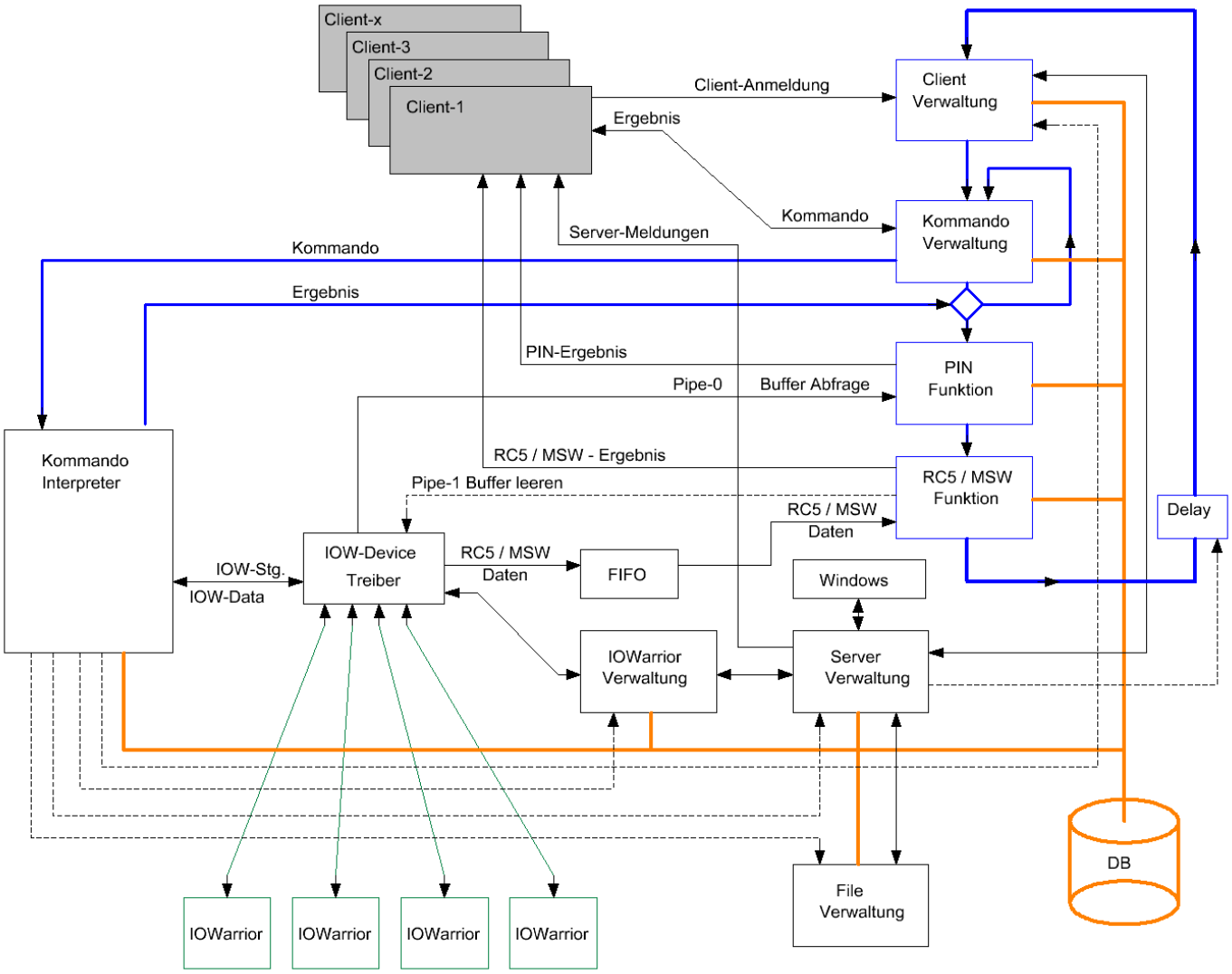
**Vom Programm verwendete Dynamic Link Library**:
user32.dll
kernel32.dll
iowkit.dll

Ein Programm (Client) was den Socket-Server nutzen möchte, muss wie der Name des Servers schon sagt,
eine TCP-Socket Verbindung mit dem im Server eingestellten Port herstellen.

Die maximale Anzahl gleichzeitiger Verbindungen kann über MAXCLIENT eingestellt werden.

# A Funktionsweise

Programm interne Verarbeitung / Ablauf




Die Clients werden in der Reihe wie diese sich an dem Server angemeldet haben, abgearbeitet.
Alle Kommandos eines Clients werden vollständig abgearbeitet,
bevor der Server zum nächsten Client wechselt.
Sollten aus dem IOW-Buffer zwischenzeitlich Informationen für die RC5, MSW Funktion gelesen werden, werden diese in einem FIFO zwischengespeichert.
Sind alle Clients abgearbeitet wird bei aktivierter PIN Funktion die Änderungen der Ports ausgelesen und an die entsprechenden Clients gesendet.
Danach wird der FIFO für die Funktionen RC5 und MSW abgearbeitet und die Daten an die entsprechenden Clients, die einer dieser Funktionen aktiviert haben gesendet.

Die Reaktionszeit ist stark von der Anzahl der Clients und dem Umfang der gesendeten Kommandos abhängig.

# A Funktionsweise

### Programm interne Verarbeitung / Ablauf

Die Antworten können somit asynchron erfolgen.

Beispiel:

Kommando_1
    Antwort_1
Kommando_2
Kommando_3
    Antwort_2
    Antwort_3
    Event_Antwort auf z.B. SVR Ereignisse
Kommando_4
Kommando_5
    Event_Antwort auf z.B. RC5 Ereignisse
    Antwort_4
    Antwort_5

Um eine richtige Zuordnung der Antworten zu gewährleisten, besteht die Möglichkeit, den Kommandos sogenannte {TAGs} mit zu geben.
Diese {TAGs} werden dann bei den entsprechenden Antworten eingefügt.

Beispiel:

Kommando_6 {Tag_101}
Kommando_7 {Tag_401}
Kommando_8 {Test}
    Antwort_6 {Tag_101}
    Antwort_7 {Tag_401}
    Antwort_8 {Test}

# B Kommandos (Schreibweise, Layout)

## Beispiele:

In der folgenden Beschreibung der Befehle sind mögliche Antworten als Beispiele gelistet und **blau** dargestellt.
Die NAK Fehlercode sind vom Socket-Server vorgegeben der dazugehörige Fehlertext kann über eine Datei selber in andere Sprachen angepasst werden.
Der Name dieser Datei muss in der **socket_server.set** hinterlegt werden und wird beim Start des Servers geladen.

**ERROR_TXT = ERROR_MESSAGE.EN.txt**

In der Beschreibung werden Parameter(Werte) in rechteckigen Klammern **[ ]** angegeben

| | |
|---|---|
| `[User-ID]` | Wert von 1 bis (MaxClients) |
| `[IOW-ID]` | Wert von 1 bis 64 |
| `[$string]` | Strings müssen mit $ beginnen, Zeichenfolge aus lesbaren ASCII-Zeichen (keine Steuerzeichen) |

## Kommandos und Antworten :

Alle Kommandos und Antworten bestehen ausschließlich aus lesbaren Zeichen ohne Steuerzeichen und werden durch ein CHR(13) abgeschlossen.
Der Server antwortet immer auf ein Kommando !
Die Antwort beinhaltet immer das dazugehörige Kommando und sind in der Antwort durch ein (=) getrennt.
Sollte das Kommando nicht erkannt werden lautet die Antwort

**Kommando=NAK 001 Unrecognized command**

' **NAK** steht für „not acknowledge“
' **ACK** steht für „acknowledge“

## Mehrzeilige Antworten

z.B. Listings werden nicht mehr wie in den vorherigen Versionen 1-2-xxx,
durch mehrere Zeilen übertragen, sondern nur noch durch eine Zeile beantwortet.
Die einzelnen Sätze (Zeilen) werden ab der Version 1-3-001 durch ein Pipe „|“ chr(124) getrennt
Beispiel Antwort:

```
LISTCLIENTUSER=1 (Client1) *|2 (Client2)|3 (Client3)
```

## Regeln für die Schreibweise der Befehle:

- groß oder klein Schreibung kann gemischt verwendet werden
- die Kommandos und deren Werte werden durch ein oder mehrere Leerzeichen getrennt
- die Kommandos können verschieden geschrieben werden.
  z.B. **Server-close** / **ServerClose** / **SERVER_CLOSE** alle werden als richtig erkannt.
- zu jedem Kommando kann ein TAG {*Marke*r} mit hinzugefügt werden, der auch wieder in der Antwort auftaucht.

## Verwendung von TAGs:

Ein TAG muss mit „**{}**“ z.B. **{TEST}** gekennzeichnet werden und wird vom SocketServer unbeeinflusst durchgereicht.
Der TAG kann an beliebiger Stelle vor oder hinter dem Kommando stehen ,
muss aber vor einem STRING($....) stehen, sonst wird dieser als Bestandteil des Strings behandelt.

Beispiel:

```
Kommando:"ListClientUser"            Antwort:"LISTCLIENTUSER= ...."
Kommando:"ListClientUser{test}"      Antwort:"LISTCLIENTUSER{test}= ...."
Kommando:"{test}ListClientUser"      Antwort:"LISTCLIENTUSER{test}= ...."

Kommando:"SendUserName $Mustermann{test}"
... {test} wird nicht als Tag erkannt sondern als Bestandteil des Username
Username = Mustermann{test}
```

## SERVER-Einstellungen

Server-Einstellungen werden vor dem beenden des Servers
in die **socket_server.set** gespeichert (überschrieben) !

Einige Einstellungen können nur dort (wenn der Server Beendet ist) vorgenommen werden.
z.B. **Port , MaxClient**

# C Server-Meldungen

## Server-Meldungen und Events:

Der Server kann Event gesteuerte Antworten verschicken z.B.
**SVR=** Server Mitteilungen werden an alle Clients versendet.
**MSG=** Mitteilungen von einem Client an einem oder an alle Clients.

Folgende Antworten werden an Clients gesendet die die besagte Funktion aktiviert haben.

**RC5=** RC5 - Funktion.
**MSW=** Matrix-Switch - Funktion.
**PIN=** PIN (Port-Input) - Funktion.
**CAP=** Capture Timer (IOW24 ab der Verion 1.0.3.0)

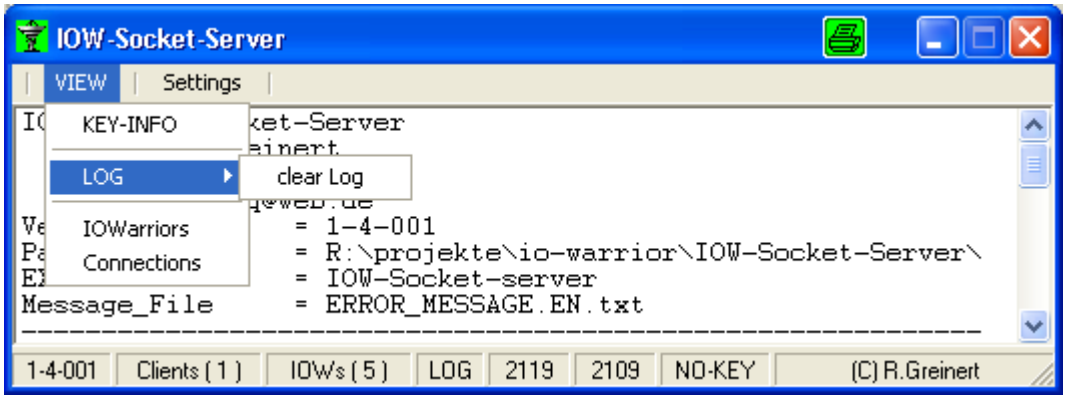
Auf diese Funktionen wird in der Anleitung noch im Detail eingegangen.

## SERVER-Meldungen an alle Clients:

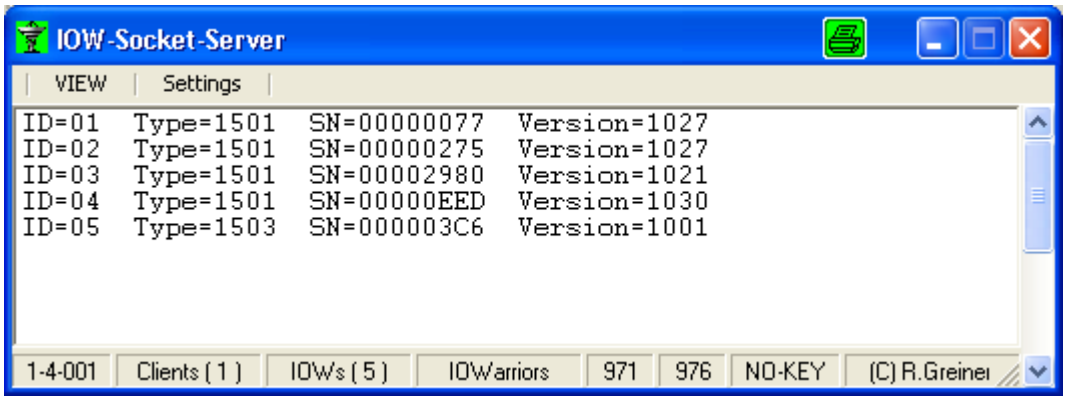
| | |
|---|---|
| **SVR=ADD IOWID[4]** | IOWarrior wurde angeschlossen und bekommt die IOW-ID 4 |
| **SVR=DEL IOWID[2]** | IOWarrior mit der IOW-ID[2] wurde entfernt und kann nicht mehr angesprochen werden. |
| **SVR=DEL CLIENT[2]** | CLIENT mit der CLIENT-ID[2] hat die Verbindung zum Server getrennt. |
| **SVR=ADD CLIENT[4]** | Ein neuer Client hat eine Verbindung zum Server aufgebaut und bekommt die CLIENT-ID 4. |

# D Server-Menüs

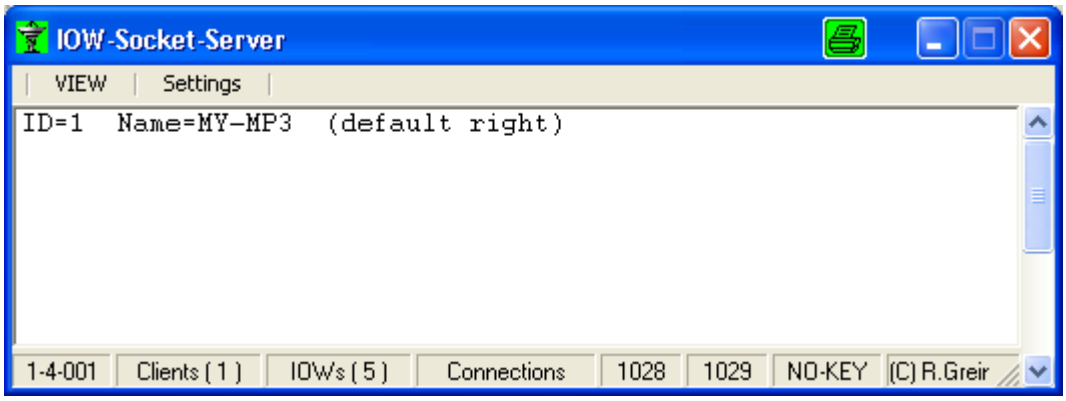
**VIEW >> LOG** zeigt das LOG an.
**VIEW >> LOG >> clear Log** löscht das LOG



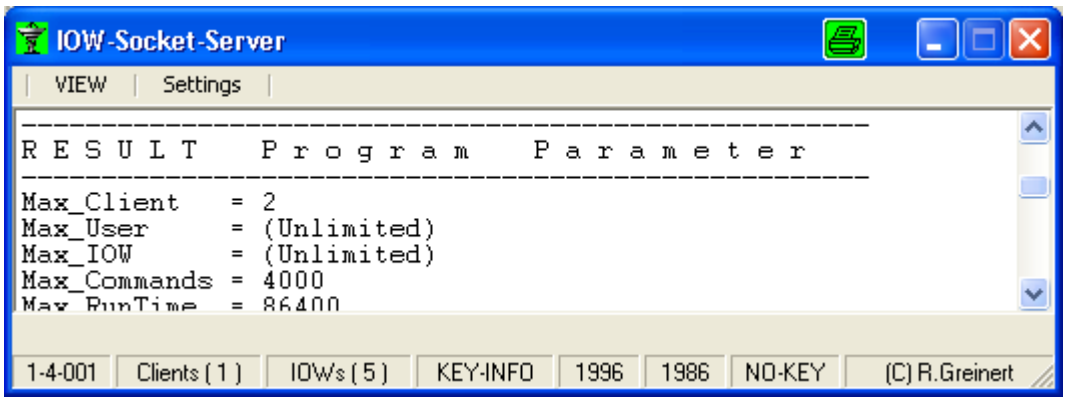

**VIEW >> IOWarrior** zeigt alle angeschlossenen IOWs an.




**VIEW >> Connection** zeigt alle verbundenen Clients an.

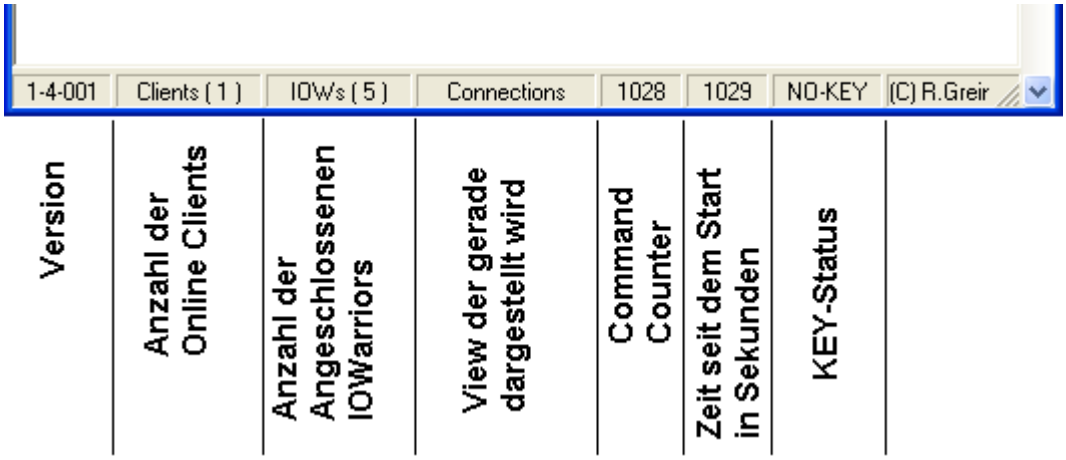



**VIEW >> KEY-Info** zeigt Informationen zum Key / Trial(Testversion) an.

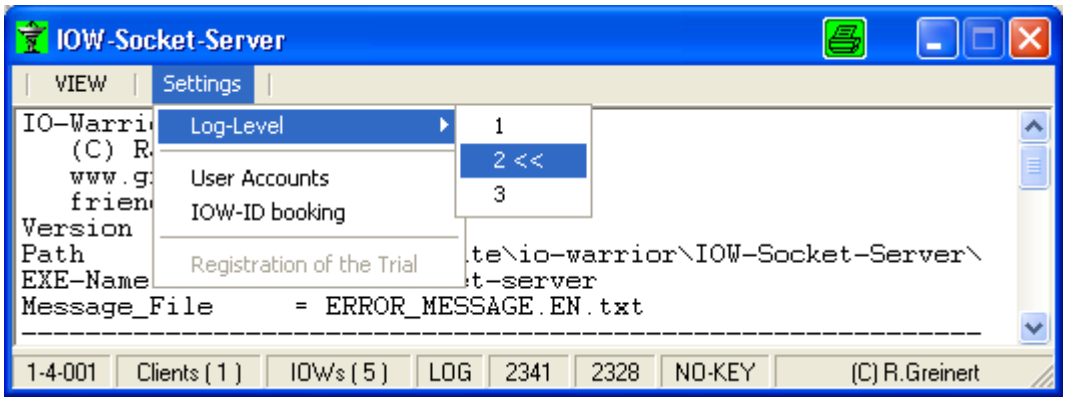


# D Server-Menüs

**Status-Leiste :**



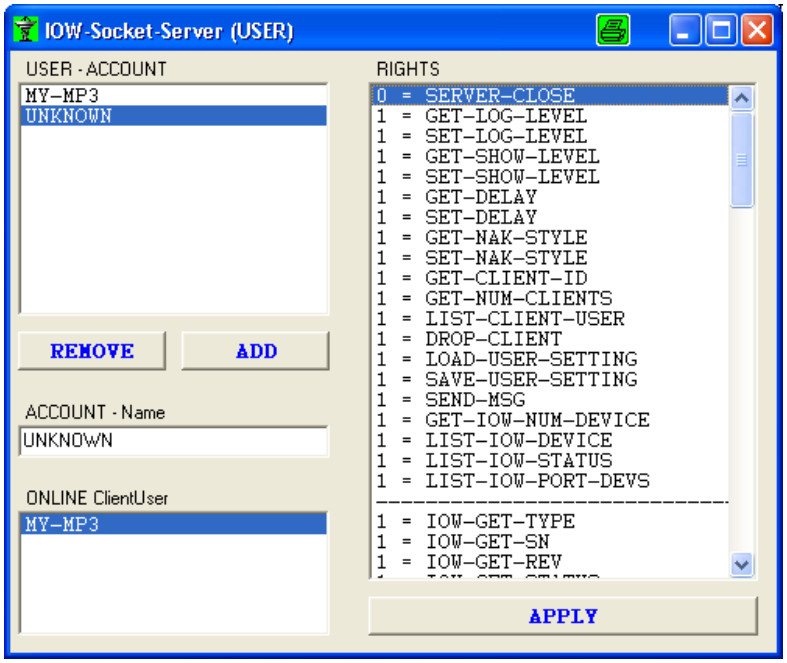

**Settings >> Log-Level** Auswahl des Log Level




1 = Server-Meldungen
2 = 1 + Status-Änderungen
3 = 1 + 2 + Alle ein und ausgehenden Daten

# D Server-Menüs

**Settings >> User Accounts** User-Account Verwaltung




Ab der Version 1-3-36 besteht die Möglichkeit, Kommandos für Clients zu sperren.

Bei einem Client der sich mit dem Kommando „SENDUSERNAME“ und einen Namen am Server Identifiziert, wird als erstes ein passender ACCOUNT (NAME) gesucht, kann KEIN passender ACCOUNT gefunden werden, wird der ACCOUNT „UNKNOWN“ gesucht.
Sollte auch dieser nicht angelegt worden behält der Client seine Default-Rechte.

Sollen ACCOUNT abhängige Rechte verwendet werden, ist es nötig den ACCOUNT „UNKNOWN“ anzulegen, alle Clients die sich nicht oder nicht richtig Identifizieren bekommen dann dessen Rechte.
Die RECHTE für den ACCOUNT „UNKNOWN“ sollten deswegen stark eingeschränkt werden.
Durch ein Doppelklick auf das RECHT ändert sich dieses ( 1=Freigegeben / 0=gesperrt ).
Mit den Button „APPLY“ werden die Änderungen für den ausgewählten ACCOUNT übernommen und sofort angewendet.
Beim Schließen des Fensters werden die ACCOUNTS in der Datei „USERACCOUNT.SET“ gespeichert.

Ist kein ACCOUNT angelegt worden, bekommen alle Clients die Default-Rechte.

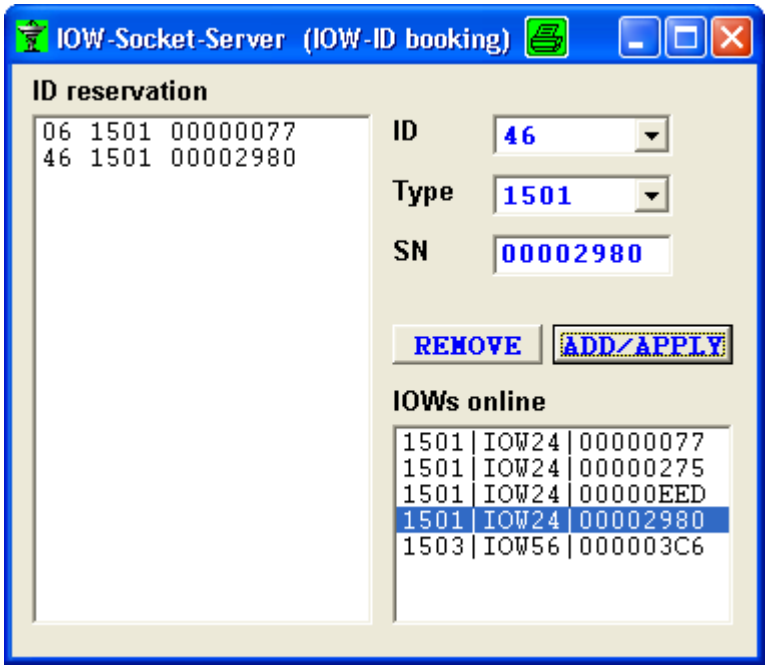
Diese Funktion ist KEY abhängig.
Ist dies nicht freigeschaltet, ist der Menüpunkt deaktiviert und es wird auch die Datei „USERACCOUNT.SET“ nicht beim Server-Start geladen.

Die Default-Rechte sind abhängig vom KEY.
Ist z.B. die I2C-Funktion nicht freigeschaltet, sind alle Kommandos für die I2C-Funktion deaktiviert und können auch nicht über die Account-Verwaltung aktiviert werden.

Für folgende Kommandos gibt es keine Rechte-Einschränkung und sind immer ausführbar:
**„SEND-USER-NAME“ , „INFO“ , „GET-VERSION“ , „GET-KEYID“ , „GET-KEYINDEX“**

# D Server-Menüs

**Settings >> IOW-ID booking** Reservierung von IDs für IOWs




In dem Fenster IOW-ID booking können für IOWs IDs reserviert werden.
Wird ein IOW vom Server erkannt, wird dieser mit der ID-Reservierung verglichen.
Stimmt Type und Seriennummer überein, wird Ihm die reservierte ID zugewiesen.
Werden IOWs erkannt, die nicht in der Reservierung gelistet worden sind,
bekommt dieser eine freie nicht reservierte ID.

Es können maximal 64 IDs reserviert werden.

Sind keine Clients connected wird diese Änderung sofort umgesetzt.
Sind noch Clients connected, werden die gewünschten Änderungen automatisch erst durchgeführt,
wenn der letzte Client sich vom Server getrennt hat.
Der Server muss dafür nicht neu gestartet werden.

Beim Schießen des Fensters werden die Reservierungen in der Datei „IDBOOKING.SET“ gespeichert.

Diese Funktion ist KEY abhängig.
Ist dies nicht freigeschaltet, ist der Menüpunkt deaktiviert und es wird auch
die Datei „IDBOOKING.SET“ nicht beim Server-Start geladen.

# E Nutzungsrecht

**Ab der Version 1-4-001 ist das Programm nur noch über einen Key voll nutzbar.**

**Unregistrierte (TRIAL) Test-Version:**
Es gibt die Möglichkeit das Programm ohne Key zu testen (TRIAL-Version).
Die unregistrierte Trial-Version ist stark eingeschränkt:
Maximale Anzahl der Kommandos ist auf 200 und die maximale Laufzeit ist auf 1 Stunde beschränkt, danach erscheint ein Fenster was auf die Trial-Version hinweist.
Nach der Bestätigung kann wieder mit dem Socket-Server gearbeitet werden.
Es können maximal 2 Clients zur gleichen Zeit den Socket-Server verwenden.

**Registrierte (TRIAL) Test-Version:**
Nach einer online Registrierung erweitern sich die Einschränkungen der Trial-Version,
die Maximale Anzahl der Kommandos auf 4000 und
die maximale Laufzeit ist dann auf 24 Stunden beschränkt.

Die Trial-Version ist nur für den privaten Gebrauch und Testzwecke freigegeben,
ein gewerbliche Nutzung ist ausdrücklich nicht erlaubt !

**Key-Version:**
Nach dem ersten Start mit vorhanden Key wird der Anwender aufgefordert den Key zu aktivieren.
Wurde der Key für das System aktiviert, kann das Programm ohne Einschränkungen gestartet werden.
Ohne Aktivierung ist der Key auch zu verwenden, muss aber nach jedem Start des Socket-Server bestätigt werden.

Nach einem Update des Socket-Server bzw. nach Änderung an der Hardware muss eine Registrierung / Aktivierung erneut vorgenommen werden.

Eine Aktivierung bindet den Key an das System, ein Wechsel auf ein anders System ist nur durch eine Erweiterung der Lizenz möglich.
Ein Key kann für 1-999 Systeme lizenziert werden.

Für die Registrierung / Aktivierung wird ein Internet-Zugang benötigt.
Port 4443(TCP) muss dafür freigeschaltet sein. Alle Daten die dafür benötigt werden,
werden Verschlüsselt über das Internet an einen Registrierungs-Server übertragen.
Die Daten für die Key-Aktivierung werden in Form von Prüfsummen gespeichert.
Die Daten für die Regestrierung der Trial-Sersion werden verschlüsselt gespeichert.
Diese gespeicherten Daten werden nicht an Dritte weiter gegeben und werden ausschließlich nur für die Aktivierung bzw. Registrierung des Programmes verwendet.

Der Key ist für die gewerbliche Nutzung und für die private uneingeschränkte Nutzung vorgesehen.

Die Kosten für den Nutzungs-Key ist abhängig vom Umfang der frei geschalteten Funktionen.

Wenn Sie einen Key für das Programm benötigen, kontaktieren Sie mich bitte per Mail:

INFO@iowss.greinert-dud.de

Sie erhalten mit einem Key ausschließlich nur das Nutzungsrecht,
soweit nichts anders vereinbart wurde, ist dieses Nutzungsrecht nicht übertragbar.
Eine Weitergabe/Verkauf an Dritte ist nicht gestattet und führt zum Erlöschen der Lizenz/ des Nutzungsrecht.

# F Rechtsbelehrung / Haftungsausschluss

**Lizenzgeber:**
Ralf Greinert
Habichtweg 5
37115 Gerblingerode

**Lizenznehmer:**
Die Person / die Gesellschaft,
auf die der KEY/Lizenz ausgestellt ist,
von der die Software genutzt wird.

**Gegestand:**
IOW-Socket-Server
Der IOW-Socket-Server ist ein Software-Tool, was es ermöglicht über eine Socket-Anbindung, lokal oder über eine Netzwerkverbindung, IO-Bausteine, der Firma „Code Mercenaries Hard- und Software GmbH“ anzusprechen und deren Funktionen zu nutzen.

**Vereinbarung:**
Die Software wird nicht verkauft, sondern eine Nutzung lizenziert.
Die Software ist geistiges Eigentum vom Lizenzgeber und unterliegt somit dem Urheberrecht.
Der Lizenzgeber räumt Ihnen als Lizenznehmer ein nicht-exklusives Nutzungsrecht (Lizenz) für das o.g. Softwareprodukt ein.

Die Lizenz gilt nur für den Lizenznehmer.
Sie gibt Ihnen nicht das Recht die Software kommerziell zu verbreiten (Redistribution).
Die Lizenz erstreckt sich ausschließlich auf die oben genannte Software.
Eine Weitergabe dieser Lizenz an Dritte ist ohne Zustimmung des Lizenzgeber nicht gestattet.
Die Software darf in keiner Form modifiziert werden.
Software oder Teile davon dürfen nicht verändert werden. Auch ist eine Dekompilierung oder Deassemblierung der Software oder die Verwendung von Ressourcen verboten.

Der Lizenzgeber übernimmt keine Gewähr für absolute Fehlerfreiheit, völlig unterbrechungsfreien Lauf oder die Kombinationsfähigkeit mit anderen Hardware- und/oder Softwarekomponenten.
Die Software wurden mit größter Sorgfalt programmiert und getestet.
Dennoch ist eine nicht ordnungsgemäße Funktionalität möglich.
Die Benutzung erfolgt auf eigene Gefahr. Für Schäden, Folgeschäden, zufällig oder indirekt entstandene Schäden, wie z.B. beschädigte, veränderte oder gelöschte Daten oder Dateien, etwaigen Schäden am System - Hard-/ Software, entgangenen Gewinnen, Anwendungs- und/oder Berechnungsfehlern, ist der Lizenzgeber nicht haftbar zu machen.
Bei etwaigen Ansprüchen dritter, die durch eine Weitergabe des Nutzungsrecht zustande kommen, haftet die Person, die das Nutzungsrecht weiter gegeben hat und nicht der Lizenzgeber.
Bei einem etwaigen Haftungsanspruch kann die Haftungssumme maximal den gezahlten Betrag für die Lizenz betragen, unabhängig davon, auf welchem Rechtsgrund diese basiert.

Die Lizenz-Vereinbarung unterliegt dem deutschem Recht.

Sollten Teile dieser Vereinbarung einem geltenden Recht widersprechen ist nur dieser Teil der Vereinbarung an dieses Recht angepasst zu sehen.
Die Vereinbarung als ganzes, bleibt als Grundlage des Nutzungsrecht wirksam bestehen.

Bereits durch eine einmalige Nutzung / Registrierung / Aktivierung der Software bestätigen Sie Ihre Zustimmung zu dieser Vereinbarung.

# 0 Server-Kommandos (ohne Rechte-Einschränkung)

## 0.1 INFO

```
Beispiel: Info
INFO{}=
Program:IOWarrior Socket-Server|
Programmed in:RapidQ|
Version:1-4-005|
IOW-DLL:IO-Warrior Kit V1.5|
Delay:50 ms|
MaxClients:25|
Connections:3|
KeyID:123456789012|
KeyIndex:1|                          'siehe GetKeyIndex
Copyright:Ralf Greinert|
Website:www.greinert-dud.de|
eMail:INFO@iowss.greinert-dud.de
```

## 0.2 GET-VERSION

```
Beispiel: GetVersion
GETVERSION{}=1-4-005
```

## 0.3 SEND-USERNAME [$Username]

```
Beispiel: SendUserName $Mustermann
SENDUSERNAME{}=ACK            'Die Verbindung hat jetzt einen Namen (Mustermann).
SENDUSERNAME{}=NAK 013 expect value for $UserName does not exist
```

## 0.4 GET-KEY-ID

```
Beispiel: Get-Key-ID
GETKEYID{}=123456789ABC       '12 Stellige Hex-Zahl
```

## 0.5 GET-KEY-INDEX

```
Beispiel: Get-Key-Index
GETKEYINDEX{}=1

0 = Key nicht vorhanden
1 = Key OK
2 = Key ungültig
3 = falsche Key-Version
4 = Key abgelaufen
```

# 1 Server-Kommandos

## 1.1 SERVER-CLOSE

```
Beispiel: ServerClose
SERVERCLOSE{}=ACK                                  'Server wird beendet
SERVERCLOSE{}=NAK 011 the number of Clients >1  'es sind mehrere Clients angemeldet
```

## 1.2 GET-LOG-LEVEL

```
Beispiel: GET-Log-Level
GETLOGLEVEL{}=3
1 = nur Server Meldungen werden in der Server-Listbox angezeigt
2 = + Client Meldungen (an / Abmeldungen)
3 = + Kommandos und Antworten (Für die Fehlersuche gedacht)

Für den Normalbetrieb sollte der Log_Level auf 1 oder 2 stehen.
```

## 1.3 SET-LOG-LEVEL [LogLevel]

```
Beispiel: SET-Log-Level 2
SETLOGLEVEL{}=ACK
SETLOGLEVEL{}=NAK 029 Expect value (1-3) for LogLevel
```

## 1.4 GET-SHOW-LEVEL

```
Beispiel: GET-Show-Level
GETSHOWLEVEL{}=2
0 = Server ist unsichtbar
1 = Server Fenster minimiert
2 = Server Fenster normal
```

## 1.5 SET-SHOW-LEVEL [ShowLevel]

```
Beispiel: SET-Show-Level 2
SETSHOWLEVEL{}=ACK
SETSHOWLEVEL{}=NAK 030 Expect value (0-2) for ShowLevel
```

## 1.6 GET-DELAY

```
Beispiel: GET-delay
GETDELAY{}=50
```

## 1.7 SET-DELAY [delay]

```
Beispiel: SET-delay 50
SETDELAY{}=ACK
SETDELAY{}=NAK 031 Expect value (1-500) for Delay
```

## 1.8 GET-NAK-STYLE

```
Beispiel: Get-Nak-Style
GETNAKSTYLE{}=1

Beschreibung des Wertes NAK-Style :
0 = NAK              ' kein Fehlercode ,kein Fehlertext
1 = NAK xxx          ' Fehlercode (dreistellig) wird mit übermittelt
2 = NAK xxx TEXT     ' Fehlercode und Text wird übermittelt
```

## 1.9 SET-NAK-STYLE [Wert] [SaveUser]

```
Beispiel: Set-Nak-Style 2
SETNAKSTYLE{}=ACK
SETNAKSTYLE{}=NAK 018 Expect value (0-2) of NAK-Style
```

# 1 Server-Kommandos

## 1.10 GET-CLIENT-ID

```
Beispiel: Get-Client-Id
GETCLIENTID{}=3                 'Abfrage der eigenen Client-ID (3)
```

## 1.11 GET-NUMCLIENTS

```
Beispiel: Get-Num-Clients
GETNUMCLIENTS{}=4               'Anzahl der angemeldeten Clients = 4
```

## 1.12 LIST-CLIENT-USER

```
Beispiel: ListClientUser
LISTCLIENTUSER{}=1 (Peter)|2 (Franz)|3 (Mustermann) *|9 (Unknown)
                 |   |    |
                 |   |     -- Satz-Trennzeichen (Pipe 124)
                 |    -- Username  (wenn gesetzt wurde) , sonst (Unknown)
                  ------ Client-ID   ( 1 bis MaxClients )
(*) ist der Client selber
```

## 1.13 DROP-CLIENT [CLIENT-ID]

```
Beispiel: Drop-Client 3
DROPCLIENT{}=ACK                'Client wurde vom Server getrennt
DROPCLIENT{}=NAK 012 Used value for Client-ID not in use
```

## 1.14 LOAD-USER-SETTING

```
Beispiel: LoadUserSetting
LOADUSERSETTING{}=ACK           'User-Settings wurden geladen
LOADUSERSETTING{}=NAK 023 Usersetting File not exists
LOADUSERSETTING{}=NAK 020 File can not open
LOADUSERSETTING{}=NAK 022 UserName is unknown

Es werden alle mit [SaveUser] markierten Werte geladen.
```

## 1.15 SAVE-USER-SETTING

```
Beispiel: SaveUserSetting
SAVEUSERSETTING{}=ACK           'User-Sets wurden in z.B. USER_Mustermann.set gespeichert
SAVEUSERSETTING{}=NAK 021 File can not save
SAVEUSERSETTING{}=NAK 022 UserName is unknown

Es werden alle mit [SaveUser] markierten Werte gespeichert.
```

## 1.16 SEND-MSG [Client-ID] [$Nachricht]

```
Beispiel: SEND-MSG 0 $TEST
SENDMSG{}=ACK

!! CLIENT-ID (0) sendet an alle Clients außer an den, der den Auftrag gegeben hat.

Der angesprochene Client erhält dann folgende Nachricht:

MSG=1 $TEST
 |  |    |
 |  |     ---- Nachricht
 |   --------- Absender Client-ID
  ------------ Kennung das es sich um eine MSG-Nachricht handelt

SENDMSG{}=NAK 025 Used value(1) for Client-ID not in use
SENDMSG{}=NAK 026 Expect value(2) for $MSG does not exist
```

# 1 Server-Kommandos

## 1.17 GET-IOW-NUM-DEVICE

```
Beispiel: Get-IOW-Num-Device
GETIOWNUMDEVICE{}=3                              'Anzahl der gefundenen IOWarriors
```

## 1.18 LIST-IOW-DEVICE

```
Beispiel: List-IOW-Device
LISTIOWDEVICE{}=1 0000FEC5 1500 IOW40 1021|2 000003FA 1501 IOW24 1021|3 0000027F ....
                |     |      |    |    |  |
                |     |      |    |    |   --- Satz-Trennzeichen (Pipe 124)
                |     |      |    |     --- IOW Revision
                |     |      |     -------- IOW Bezeichnung
                |     |       ------------- IOW Type
                |      --------------------- IOW SerienNummer
                 -------------------------- IOW-ID

IOWDEVICE=NAK 005 No IOWarrior available
```

## 1.19 LIST-IOW-STATUS

**geändert mit Version 1-4-001 !!**

```
Beispiel: List-IOW-Status
IOWLISTSTATUS{}=1 000000000xxx|2 000100100xxx|3 000000100xxx
                | ||||||||| | |
                | ||||||||| |  - Satz-Trennzeichen (Pipe 124)
                | ||||||||| |
                | |||||||||  -----x-zukünftige Funktionen werden dann angefügt !!
                | |||||||||
                | |||||||| -------8- CAT-IOW-Funktion (capture-Timer)
                | ||||||| --------7- MQ -Server-Funktion (LED-Laufschrift)
                | |||||| ---------6- PIN-Server-Funktion (in Planung)
                | ||||| ----------5- SWT-IOW-Funktion (Matrix-Switch)
                | |||| -----------4- LED-IOW-Funktion (LED-Matrix)
                | ||| ------------3- RC5-IOW-Funktion
                | || -------------2- SPI-IOW-Funktion
                | | --------------1- LCD-IOW-Funktion
                |  ---------------0- I2C-IOW-Funktion
                |
                 ------------------- IOW-ID

IOWLISTSTATUS{}=NAK 005 No IOWarrior available
```

## 1.20 LIST-IOW-PORT-DEVS

```
Beispiel: List-IOW-Port-Devs
LISTIOWPORTDEVS{}=1 4 11111111 11111111 11111111 11110000|2 2 11111100 11000011 ...
                  | |    |        |        |        |    |
                  | |    |        |        |        |     --- Satz-Trennzeichen (Pipe)
                  | |    0        1        2        3 Port
                  |  --- Anzahl Ports
                   ----- IOW-ID

              ... |1 7 11111111 11111111 11111111 11111111 11111111 11111111 11111111
                  || |    |        |        |        |        |        |        |
                  || |    0        1        2        3        4        5        6  Port
                  ||  --- Anzahl Ports
                  | ----- IOW-ID
                   --- Satz-Trennzeichen (Pipe)


LISTIOWPORTDEVS{}=NAK 005 No IOWarrior available
```

# 2 IOWarrior bezogene Kommandos allgemein

**Diese Kommandos fangen alle mit IOW an,**
**Mögliche Fehlermeldungen bei allen IOW bezogenen Kommandos:**

```
IOWxxxxxx{}=NAK 005 No IOWarrior available
IOWxxxxxx{}=NAK 007 Selected IOW does not support this feature
IOWxxxxxx{}=NAK 036 expect value(1) for Device (1-64)
IOWxxxxxx{}=NAK 037 expect value(1) for Device[x] not in use
IOWxxxxxx{}=NAK 041 Function on IOW-Device not ON
```

## 2.1 IOW-GET-TYPE [IOW-ID]

```
Beispiel: IOW-GET-Type 2
IOWGETTYPE{}=1501
```

## 2.2 IOW-GET-SN [IOW-ID]

```
Beispiel: IOW-GET-SN 2
IOWGETSN{}=000003FA
```

## 2.3 IOW-GET-REV [IOW-ID]

```
Beispiel: IOW-GET-Rev 2
IOWGETREV{}=1021
```

## 2.4 IOW-GET-STATUS [IOW-ID] geändert mit Version 1-4-001 !!

```
Beispiel: IOW-GET-STATUS 3
IOWGETSTATUS{}=010010001xxx
              |||||||||  |
              |||||||||   ---- x-zukünftige Funktionen werden dann angefügt !!
              ||||||||  -8- CAT-IOW-Funktion (capture-Timer)
              |||||||  --7- MQ -Server-Funktion (LED-Lauflicht)
              ||||||  ---6- PIN-Server-Funktion (Port Input)
              |||||  ----5- SWT-IOW-Funktion (Matrix-Switch)
              ||||  -----4- LED-IOW-Funktion (LED-Matrix)   'ist z.B. aktiviert
              |||  ------3- RC5-IOW-Funktion
              ||  -------2- SPI-IOW-Funktion
              |  --------1- LCD-IOW-Funktion                'ist z.B. aktiviert
               ---------0- I2C-IOW-Funktion
```

## 2.5 IOW-LIST-STATUS [IOW-ID]

```
Beispiel: IOW-LIST-STATUS 3
IOWLISTSTATUS{}=0 0 0 I2C|1 1 1 LCD|2 0 0 SPI|3 0 0 RC5|4 1 2 LED|5 0 0 SWT|
                6 0 0 PIN|7 0 0 MQ|8 0 0 CAP
                | | |  | |
                | | |  |  --- Satz-Trennzeichen (Pipe 124)
                | | |   --- Bezeichnung der Funktion
                | |  ------ Funktion wurde von x Clients aktiviert
                |  -------- Funktion Status 0=OFF 1=ON
                 ---------- No der Funktion
```

## 2.6 IOW-SAVE-DEV-SETS [IOW-ID]

```
Beispiel: IOW-Save-DEV-Sets 2
IOWSAVEDEVSETS{}=ACK          'Device-Sets wurden in 1501_000003FA.set gespeichert
```

Es werden alle mit [SaveIOW] markierten Werte für den ausgewählten IOW gespeichert. Diese Einstellungen werden wieder geladen, beim Starten des Socket-Server oder wenn ein IOW eingesteckt wird.

# 3 IOWarrior bezogene Kommandos I2C und SHT

## 3.1 IOW-I2C-ON [IOW-ID]

```
Beispiel: IOW-I2C-On 2        'IOW-Funktion I2C wird ein-geschaltet
IOWI2CON{}=ACK
```

## 3.2 IOW-I2C-OFF [IOW-ID]

```
Beispiel: IOW-I2C-Off 2       'IOW-Funktion I2C wird aus-geschaltet
IOWi2COFF{}=ACK
```

## 3.3 IOW-GET-I2C-CLOCK [IOW-ID]

```
Beispiel: IOW-Get-I2C-clock 2       'Abfragen Bus-Takt
IOWGETI2CCLOCK{}=100
```

## 3.4 IOW-SET-I2C-CLOCK [IOW-ID] [Wert] [SaveIOW]

```
Beispiel: IOW-Set-I2C-clock 2  100        'setzen des Bus-Takt
IOWSETI2CCLOCK{}=ACK
IOWSETI2CCLOCK{}=NAK 091 Expect value(2) for clock (100/400/50) or (0/1/2)
IOWSETI2CCLOCK{}=NAK 092 IOW24 or IOW40 can't change clock only used 100 kHz
IOWSETI2CCLOCK{}=NAK 093 Can't change clock function is in used
```

## 3.5 IOW-GET-I2C-WAIT [IOW-ID]

```
Beispiel: IOW-GET-I2C-Wait 3
IOWGETI2CWAIT{}=5
```

## 3.6 IOW-SET-I2C-WAIT [IOW-ID] [Wert] [SaveIOW]

```
Beispiel: IOW-SET-I2C-Wait 3 20
IOWSETI2CWAIT{}=ACK
IOWSETI2CWAIT{}=NAK 088 Expect value(2) wait of I2C (0-500) ms
```

I2C-WAIT wird benötigt, bei zu langsamen I2C-Bausteinen. Da im Socket-Server nur der NB-READ verwendet wird, ist ein warten auf Antwort wie beim Read nicht möglich. Nach dem schreiben werden erst die Werte verarbeiten, bevor ein I2C-Baustein antwortet. Fragt der Socket-Server einen Baustein (Buffer) zu früh ab, kommt es zu Fehlermeldungen: z.B. I2C-Device not acknowledged bzw. I2C-Device write error.

## 3.7 IOW-I2C-SCAN [IOW-ID]

```
Beispiel: IOW-I2C-scan 2            'I2C-Bus wird durchsucht und alle gefundenen
                                    'Bausteine aufgelistet
IOWI2CSCAN{}=2 160 162
             |  |   |
             |   --- --- gefundene Adressen
              ---------- Anzahl der gefundenen Adressen
IOWI2CSCAN{}=NAK 097 Commands run not with selected protocol
```

## 3.8 IOW-I2C-READ-H [IOW-ID] [I2C-Adr] [num Bytes]

```
Beispiel: IOW-I2C-ReadH 3 160 10
IOWI2CREADH{}=00FF00050FFF00995533       'Daten-Übergabe in HEX
IOWI2CREADH{}=NAK 082 Expect value(2) for I2C-Addr. (2-254)
IOWI2CREADH{}=NAK 083 Expect value(3) num Bytes (1-128)
IOWI2CREADH{}=NAK 081 I2C-Device doesn't respond
IOWI2CREADH{}=NAK 097 Commands run not with selected protocol
```

## 3.9 IOW-I2C-READ-D [IOW-ID] [I2C-Adr] [num Bytes]

```
Beispiel: IOW-I2C-ReadD 3 160 5
IOWI2CREADD{}=0 255 0 5 15          'Daten-Übergabe in DEC
IOWI2CREADD{}=NAK 082 Expect value(2) for I2C-Addr. (2-254)
IOWI2CREADD{}=NAK 083 Expect value(3) num Bytes (1-128)
IOWI2CREADD{}=NAK 081 I2C-Device doesn't respond
IOWI2CREADD{}=NAK 097 Commands run not with selected protocol
```

### 3.10 IOW-I2C-WRITE [IOW-ID] [I2C-Adr] [$HEX-STRING]

```
Beispiel: IOW-I2C-WRITE 3 160 $00110FF0FF
IOWI2CWRITE{}=ACK
IOWI2CWRITE{}=NAK 081 I2C-Device doesn't respond
IOWI2CWRITE{}=NAK 082 Expect value(2) for I2C-Addr. (2-254)
IOWI2CWRITE{}=NAK 084 expect value(3) String not exist
IOWI2CWRITE{}=NAK 085 I2C-Device write error
IOWI2CWRITE{}=NAK 086 I2C-Device not acknowledged
IOWI2CWRITE{}=NAK 097 Commands run not with selected protocol
```

### 3.11 IOW-I2C-WRITE-SB [IOW-ID] [I2C-Adr] [Wert]

```
Beispiel: IOW-I2C-WRITE-SB 3 160 15            'Write single Byte
IOWI2CWRITESB{}=ACK
IOWI2CWRITESB{}=NAK 081 I2C-Device doesn't respond
IOWI2CWRITESB{}=NAK 082 Expect value(2) for I2C-Addr. (2-254)
IOWI2CWRITESB{}=NAK 087 Expect value(3) Data Bytes (0-255)
IOWI2CWRITESB{}=NAK 085 I2C-Device write error
IOWI2CWRITESB{}=NAK 086 I2C-Device not acknowledged
IOWI2CWRITESB{}=NAK 097 Commands run not with selected protocol
```

### 3.12 IOW-GET-I2C-PROTOCOL [IOW-ID]

```
Beispiel: IOW-GET-I2C-Protocol
IOWGETI2CPROTOCOL{}=0 I2C
IOWGETI2CPROTOCOL{}=1 SHT
```

### 3.13 IOW-SET-I2C-PROTOCOL [IOW-ID] [Wert] [SaveIOW]

```
Beispiel: IOW-SET-I2C-Protocol 3 1
IOWSETI2CPROTOCOL{}=ACK
IOWSETI2CPROTOCOL{}=NAK 093 Can't change I2C-function is in used
IOWSETI2CPROTOCOL{}=NAK 094 Expect value(2) for Protocol (0/1) 0=I2C 1=SHT
IOWSETI2CPROTOCOL{}=NAK 097 Commands run not with selected protocol
```

### 3.14 IOW-GET-I2C-TIMEOUT [IOW-ID]

```
Beispiel: IOW-GET-I2C-TimeOut
IOWGETI2CTIMEOUT{}=128
```

### 3.15 IOW-SET-I2C-TIMEOUT [IOW-ID] [Wert] [SaveIOW]

```
Beispiel: IOW-SET-I2C-TimeOut 3 128
IOWSETI2CTIMEOUT{}=ACK
IOWSETI2CTIMEOUT{}=NAK 093 Can't change I2C-function is in used
IOWSETI2CTIMEOUT{}=NAK 095 Expect value(2) for Timeout (0-255)
```

### 3.16 IOW-GET-I2C-PULLUPS [IOW-ID]

```
Beispiel: IOW-GET-I2C-Pullups
IOWGETI2CPULLUPS{}=0
```

### 3.17 IOW-SET-I2C-PULLUPS [IOW-ID] [Wert] [SaveIOW]

```
Beispiel: IOW-SET-I2C-Pullups 3 1
IOWSETI2CPULLUPS{}=ACK
IOWSETI2CPULLUPS{}=NAK 093 Can't change I2C-function is in used
IOWSETI2CPULLUPS{}=NAK 096 Expect value(2) for PullUps (0/1) 0=enablde 1=disabled
```

## 3.18 IOW-SET-SHT-Heater [IOW-ID] [Wert]

```
Beispiel: IOW-SET-SHT-Heater 3 1          '0=SHT-Heater aus / 1=SHT-Heater ein
IOWSETSHTHEATER{}=ACK
IOWSETSHTHEATER{}=NAK 086 I2C-Device not acknowledged
IOWSETSHTHEATER{}=NAK 097 Commands run not with selected protocol
```

## 3.19 IOW-SET-SHT-Resolution [IOW-ID] [Wert]

```
Beispiel: IOW-SET-SHT-Resolution 3 1      '0=hohe Auflösung / 1=geringe Auflösung
IOWSETSHTRESOLUTION{}=ACK
IOWSETSHTRESOLUTION{}=NAK 086 I2C-Device not acknowledged
IOWSETSHTRESOLUTION{}=NAK 097 Commands run not with selected protocol
```

## 3.20 IOW-GET-SHT-STATUS [IOW-ID]

```
Beispiel: IOW-GET-SHT-Status 3            'SHT Status-Register abfragen
IOWGETSHTSTATUS{}=0 00000000
IOWGETSHTSTATUS{}=4 00000100              'Heater on
```

## 3.21 IOW-GET-SHT-DATA [IOW-ID]

```
Beispiel: IOW-GET-SHT-DATA 3        'SHT Daten-Register abfragen
IOWGETSHTDATA{}=0 6366 1390
               |  |    |
               |  |     --- Humidity
               |   -------- Temp
                ----------- Status
```

## 3.22 IOW-GET-SHT [IOW-ID]

```
Beispiel: IOW-GET-SHT 3             'SHT Daten berechnet abfragen
IOWGETSHT{}=2350 4676
            |    |
            |     ---- Humidity / 100    =     (46,76 %)
             --------- Temp     / 100    =     (23,50 C°)
```

Für die Berechnung werden folgende Daten als Default verwendet: V3-Sensor an 5 Volt.

| 12bit RHT / 14bit Temp | 8bit RHT / 12bit Temp |
|---|---|
| SHT_C1.0 =-4 | SHT_C1.1 =-4 |
| SHT_C2.0 =0.0405 | SHT_C2.1 =0.648 |
| SHT_C3.0 =-0.000002.8 | SHT_C3.1 =-0.00072 |
| SHT_T1.0 =0.01 | SHT_T1.1 =0.01 |
| SHT_T2.0 =0.00008 | SHT_T2.1 =0.00128 |
| SHT_D1 =-40.1 | SHT_D1 =-40.1 |
| SHT_D2.0 =0.01 | SHT_D2.1 =0.04 |

Diese Parameter können nur in der Datei [Type_Serien-Nr.set] angepasst werden.
Diese Datei kann durch das Kommando „IOW-Save-Dev-Sets" erzeugt werden und danach mit einem Editor entsprechend angepasst werden.

# 4 IOWarrior bezogene Kommandos SPI

## 4.1 IOW-SPI-ON [IOW-ID]

```
Beispiel: IOW-SPI-On 3
IOWSPION{}=ACK
```

## 4.2 IOW-SPI-OFF [IOW-ID]

```
Beispiel: IOW-SPI-Off 3
IOWSPIOFF{}=ACK
```

## 4.3 IOW-GET-SPI-CLOCK [IOW-ID]

```
Beispiel: IOW-Get-SPI-Clock 3
IOWGETSPICLOCK{}=ACK
IOWGETSPICLOCK{}=NAK 121 IOW40 not support SPI
```

## 4.4 IOW-SET-SPI-CLOCK [IOW-ID] [CLOCK] [SaveIOW]

```
Beispiel: IOW-Set-SPI-Clock 3  2
IOWSETSPICLOCK{}=ACK
IOWSETSPICLOCK{}=NAK 121 IOW40 not support SPI
IOWSETSPICLOCK{}=NAK 123 Expect value(2) for Clock (1-255)  'IOW56
IOWSETSPICLOCK{}=NAK 122 Expect value(2) for Clock (0-3)    'IOW24
IOWSETSPICLOCK{}=NAK 126 Can't change SPI-clock function is in used

IOW-24  0= 2 MBit/s     1=1 MBit/s        2=500 KBit/s      3=62,5 KBit/s
IOW-56  1= 12 MBit/s >> 255= 93,75 KBit/s   24 MHz /(clock+1) = SPI-CLOCK
```

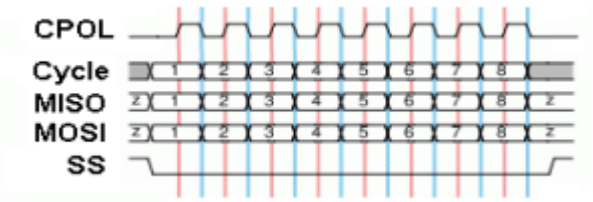
## 4.5 IOW-GET-SPI-MODE [IOW-ID]

```
Beispiel: IOW-Set-SPI-Mode 3
IOWGETSPIMODE{}=ACK
IOWGETSPIMODE{}=NAK 121 IOW40 not support SPI
```

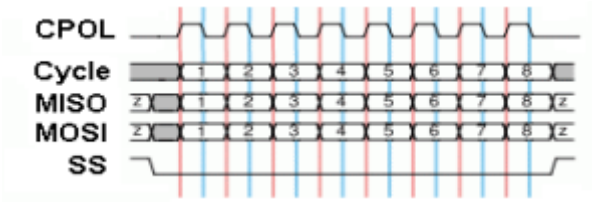
## 4.6 IOW-SET-SPI-MODE [IOW-ID] [MODE] [SaveIOW]

```
Beispiel: IOW-Set-SPI-Mode 3 2
IOWSETSPIMODE{}=ACK
IOWSETSPIMODE{}=NAK 121 IOW40 not support SPI
IOWSETSPIMODE{}=NAK 124 Expect value(2) for MODE (0-3)
IOWSETSPIMODE{}=NAK 127 Can't change SPI-mode function is in used
```

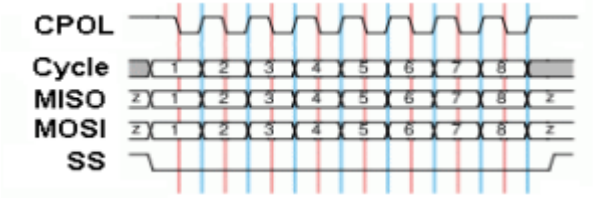
**Mode 0**



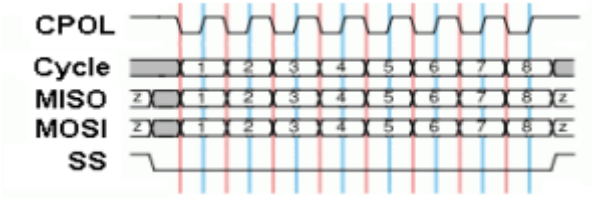

**Mode 1**




**Mode 2**




**Mode 3**



# 4 IOWarrior bezogene Kommandos SPI

## 4.7 IOW-SPI-READH [IOW-ID] [num-Byte]

```
Beispiel: IOW-SPI-READ-H 2 8
IOWSPIREADH{}=1122334455667788
IOWSPIREADH{}=NAK 125 Expect value(2) num Bytes (1-256)
```

## 4.8 IOW-SPI-READD [IOW-ID] [num-Byte]

```
Beispiel: IOW-SPI-READ-D 2 8
IOWSPIREADD{}=17 34 51 68 85 102 119 136
IOWSPIREADD{}=NAK 125 Expect value(2) num Bytes (1-256)
```

## 4.9 IOW-SPI-RWH [IOW-ID] [$DAT-String]

```
Beispiel: IOW-SPI-RWH 2 $0011223344
IOWSPIRWH{}=0011223344                          (MISO und MOSI verbunden)
IOWSPIRWH{}=NAK 125 Expect value(2) num Bytes (1-256)
```

## 4.10 IOW-SPI-READ-BLOCK [IOW-ID] [num-Byte] [num-Block]

```
Beispiel: IOW-SPI-READ-Block 2 4 16
IOWSPIREADBLOCK{}=00112233|00112233|00112233|...|00112233
                  Block-01|Block-02|Block-03|...|Block-16

IOWSPIREADBLOCK{}=NAK 125 Expect value(2) num Bytes (1-256)
IOWSPIREADBLOCK{}=NAK 129 Expect value(3) num Block (1-256)

num-Byte  = Anzahl der Byte pro Block
num-Block = Anzahl der Blöcke

Dieses Kommando ist dafür gedacht, über die SPI-Schnittstelle mehrere Lesevorgänge
hinter einander automatisch durchzuführen.
Die Ergebnisse(Blöcke) werden danach als ein Satz, mit Satz-Trennzeichen übertragen.
```

# 5 IOWarrior bezogene Kommandos LCD

## 5.1 IOW-LCD-ON [IOW-ID]

```
Beispiel: IOW-LCD-ON 3
IOWLCDON{}=ACK
```

## 5.2 IOW-LCD-OFF [IOW-ID]

```
Beispiel: IOW-LCD-OFF 3
IOWLCDOFF{}=ACK
```

## 5.3 IOW-LCD-INIT [IOW-ID]

```
Beispiel: IOW-LCD-INIT 3
IOWLCDINIT{}=ACK
```

## 5.4 IOW-LCD-CLEAR [IOW-ID]

```
Beispiel: IOW-LCD-CLEAR 3
IOWLCDCLEAR{}=ACK
```

## 5.5 IOW-LCD-WRITE [IOW-ID] [Position] [$Text-String]

```
Beispiel: IOW-LCD-Write 3 10 $Text1234
IOWLCDWRITE{}=ACK
IOWLCDWRITE{}=NAK 043 Expect value(2) of LCD-Position (0-103)
IOWLCDWRITE{}=NAK 044 Expect value(3) of LCD-Text does not exist
```

**!INFO!**

**Wird die LCD-Funktion an einem IOWorrior verwendet,**
**ohne das ein funktionsfähiges Display angeschlossen ist, hängt der Server !!**
**Da der entsprechende IOW auf das nicht vorhandene Display wartet !!**
**Einzige Lösung den IOW vom Server trennen !**
**Bei den IOWorrrior 24/40, ab der Version 1.0.3.0 ist dieses Problem behoben worden.**

# 6 IOWarrior bezogene Kommandos LED-Matrix und Laufschrift(MQ)

## 6.1 IOW-LED-ON [IOW-ID]

```
Beispiel: IOW-LED-ON 3
IOWLEDON{}=ACK
IOWLEDON{}=NAK 047 Function is supported by a marquee blocked
```

## 6.2 IOW-LED-OFF [IOW-ID]

```
Beispiel: IOW-LED-OFF 3
IOWLEDOFF{}=ACK
IOWLEDOFF{}=NAK 048 Function is not used by this client
```

## 6.3 IOW-LED-CLEAR [IOW-ID]

```
Beispiel: IOW-LED-CLEAR 3
IOWLEDCLEAR{}=ACK
IOWLEDCLEAR{}=NAK 048 Function is not used by this client
```

## 6.4 IOW-LED-WRITE [IOW-ID] [Position] [$HEX-String]

```
Beispiel: IOW-LED-WRITE 3 30 $00110FF0FF
IOWLEDWRITE{}=ACK
IOWLEDWRITE{}=NAK 048 Function is not used by this client
IOWLEDWRITE{}=NAK 051 Expect value(2) of LED-Matrix columns (1-64)
IOWLEDWRITE{}=NAK 052 Expect value(3) of $HEX-String (00-FF)
```

**Spezial-Server-Funktion MQ (marquee oder Laufllicht)**

## 6.5 IOW-LED-MQ-TEXT [IOW-ID] [$LaufschriftText]

```
Beispiel: IOW-LED-MQ-TEXT 3 $Hello World
IOWLEDMQTEXT{}=ACK
IOWLEDMQTEXT{}=NAK 054 Expect value(2) $MarqueeText does not exist
```

## 6.6 IOW-LED-MQ-INTERVAL [IOW-ID] [Wert in ms]

```
Beispiel: IOW-LED-MQ-INTERVAL 3 35
IOWLEDMQINTERVAL{}=ACK
IOWLEDMQINTERVAL{}=NAK 060 Expect value(2) Marquee interval (20-200)
```

## 6.7 IOW-LED-MQ-SIZE [IOW-ID] [Wert]

```
Beispiel: IOW-LED-MQ-SIZE 3 32
IOWLEDMQSIZE{}=ACK
IOWLEDMQSIZE{}=NAK 057 Expect value(2) of LED-Matrix SIZE (1-64)
```

## 6.8 IOW-LED-MQ-START [IOW-ID]

```
Beispiel: IOW-LED-MQ-START 3
IOWLEDMQSTART{}=ACK
IOWLEDMQSTART{}=NAK 048 Function is not used by this client
IOWLEDMQSTART{}=NAK 049 function is used by several clients,
                        Marquee alone can only use this feature
IOWLEDMQSTART{}=NAK 063 MarqueeText_Buffer is empty
```

## 6.9 IOW-LED-MQ-STOP [IOW-ID]

```
Beispiel: IOW-LED-MQ-STOP 3
IOWLEDMQSTOP{}=ACK
IOWLEDMQSTOP{}=NAK 048 Function is not used by this client
```

# 7 IOWarrior bezogene Kommandos PORTs

## 7.1 IOW-GET-NUMPORTS [IOW-ID]

```
Beispiel: IOW-Get-NumPorts 3
IOWGETNUMPORTS{}=7
```

## 7.2 IOW-READ-PORTS [IOW-ID]

```
Beispiel: IOW-read-Ports 2
IOWREADPORTS{}=11111111 11111111
```

## 7.3 IOW-READ-PORTS-D [IOW-ID]

```
Beispiel: IOW-read-Ports-D 2
IOWREADPORTSD{}=255 255
```

## 7.4 IOW-READ-PORT [IOW-ID] [Port]

```
Beispiel: IOW-read-Port 2 0
IOWREADPORT{}=11111111
IOWREADPORT{}=NAK 071 Expect value(2) for Port IOW24(0-1) IOW40(0-3) IOW56(0-6)
```

## 7.5 IOW-READ-PORT-D [IOW-ID] [Port]

```
Beispiel:   IOW-read-PortD 2 0
IOWREADPORTD{}=255
IOWREADPORTD{}=NAK 071 Expect value(2) for Port IOW24(0-1) IOW40(0-3) IOW56(0-6)
```

## 7.6 IOW-GET-PORTS-DEV [IOW-ID]

```
Beispiel: IOW-GET-PORTS-DEV 2
IOWGETPORTSDEV{}=11111111 11110000
                  |        |
                  |         --- Port 1 (1=Eingang 0=Ausgang)
                   ------------ Port 0 (1=Eingang 0=Ausgang)
```

## 7.7 IOW-GET-PORT-DEV [IOW-ID] [Port]

```
Beispiel: IOW-GET-PORT-DEV 2 1
IOWGETPORTDEV{}=11110000
                |
                 --- ausgewählter Port (1) (1=Eingang 0=Ausgang)
IOWGETPORTDEV{}=NAK 071 Expect value(2) for Port IOW24(0-1) IOW40(0-3) IOW56(0-6)
```

## 7.8 IOW-SET-PORT-DEV [IOW-ID] [Port] [wert/$binstring] [SaveIOW]

```
Beispiel: IOW-SET-PORT-DEV 2 1 15   oder   IOW-SET-PORT-DEV 2 1 $00001111
IOWSETPORTDEV{}=ACK
IOWSETPORTDEV{}=NAK 071 Expect value(2) for Port IOW24(0-1) IOW40(0-3) IOW56(0-6)
IOWSETPORTDEV{}=NAK 072 Expect value(3) for set ( 0-255 or $01001111 )
```

## 7.9 IOW-WRITE-PORT [IOW-ID] [Port] [wert/$binstring]

```
Beispiel: IOW-Write-Port 2 1 15   oder   IOW-Write-Port 2 1 $00001111
IOWWRITEPORT{}=ACK
IOWWRITEPORT{}=NAK 071 Expect value(2) for Port IOW24(0-1) IOW40(0-3) IOW56(0-6)
IOWWRITEPORT{}=NAK 072 Expect value(3) for set ( 0-255 or $01001111 )
```

## 7.10 IOW-WRITE-PBIT [IOW-ID] [Port] [Bit] [Wert]

```
Beispiel: IOW-Write-PBit 2 1 7 0
IOWWRITEPBIT{}=ACK
IOWWRITEPBIT{}=NAK 071 Expect value(2) for Port IOW24(0-1) IOW40(0-3) IOW56(0-6)
IOWWRITEPBIT{}=NAK 073 Expect value(3) for Bit ( 0-7 )

'Ports deren Anschlusse als Eingänge definiert worden sind, werden immer auf 1 gesetzt
'Ports/Bits die nicht geschrieben werden, bleiben unverändert !
```

# 8 IOWarrior bezogene Kommandos RC5

## 8.1 IOW-RC5-ON [IOW-ID]

```
Beispiel: IOW-RC5-ON 2
IOWRC5ON{}=ACK
IOWRC5ON{}=NAK 101 IOW40 or IOW56 not support RC5
```

## 8.2 IOW-RC5-OFF [IOW-ID]

```
Beispiel: IOW-RC5-OFF 2
IOWRC5OFF{}=ACK
IOWRC5OFF{}=NAK 101 IOW40 or IOW56 not support RC5
```

**Alle Clients die diese Funktion an einem IOWarrior (24) aktiviert haben bekommen, bei jedem Event (RC5-Code ist eingegangen) automatisch eine Nachricht vom Server.**

```
RC5=2 12 53 197
RC5=2 12 41 229
  |   | |  |    |
  |   | |  |     -- IOW-DAT(2) Tasten Code
  |   | |   ------ IOW-DAT(1) Geräte Code
  |   |  --------- IOW-DAT(0) Report ID
  |    ------------ IOW-ID
   --------------- Kennung für RC5 Mitteilung
```

# 9 IOWarrior bezogene Kommandos Matrix-Switch

## 9.1 IOW-MSW-ON [IOW-ID]

```
Beispiel: IOW-MSW-ON 1
IOWMSWON{}=ACK
IOWMSWON{}=NAK 111 IOW24 not support Switch-Matrix
```

## 9.2 IOW-MSW-OFF [IOW-ID]

```
Beispiel: IOW-MSW-OFF 1
IOWMSWOFF{}=ACK
IOWMSWOFF{}=NAK 111 IOW24 not support Switch-Matrix
```

## 9.3 IOW-GET-MSW-STYLE [IOW-ID]

```
Beispiel:   IOW-GET-MSW-STYLE 1
IOWGETMSWSTYLE{}=2
```

## 9.4 IOW-SET-MSW-STYLE [IOW-ID] [STYLE] [saveUser]

```
Beispiel: IOW-GET-MSW-STYLE 1 2
IOWSETMSWSTYLE{}=ACK
IOWSETMSWSTYLE{}=NAK 112 Expect value(2) Output Style (0-2)
```

## 9.5 IOW-GET-MSW-TYPE [IOW-ID]

```
Beispiel: IOW-GET-MSW-Type 1
IOWGETMSWTYPE{}=0
```

## 9.6 IOW-SET-MSW-TYPE [IOW-ID] [Type] [saveIOW]

```
Beispiel: IOW-GET-MSW-Type 1 1
IOWGETMSWTYPE{}=ACK
IOWGETMSWTYPE{}=NAK 113 Expect value(2) Matrix-Type (0-1)
IOWGETMSWTYPE{}=NAK 114 Can't change Matrix-Type function is in used

Type=0       8 x 8 Matrix
Type=1      16 x 8 Matrix  IOW 40 ab Version 1.0.3.0
```

Alle Clients die diese Funktion aktiviert haben bekommen bei jedem Event
(an der Matrix hat sich was geändert) automatisch eine Nachricht vom Server.

**MSW-STYLE = 0  Tasten werden Binär(64Bit) übertragen 8x8 Matrix**

```
MSW=2 0000000000000000000000000000000000000000000000000000000000000001  Taste gedrückt
MWS=2 0000000000000000000000000000000000000000000000000000000000000000  losgelassen
 |  | |                                                               |
 |  | |                                                                -- Taste 0.0
 |  |  --------- Taste 8.8
 |   ----------- IOW-ID
  -------------- Kennung für MWS Mitteilung
```

**MSW-STYLE = 0  Tasten werden Binär(2x64Bit) übertragen 16x8 Matrix**

```
MWS=2 0000000000...00000000000000 0000000000...000000000000000
 |  | |                          | |                          |
 |  | |                          |  --------- Taste 16.8       ---- Taste 9.8
 |  |  --------- Taste 8.8        -- Taste 0.0
 |   ----------- IOW-ID
  -------------- Kennung für MWS Mitteilung
```

# 9 IOWarrior bezogene Kommandos Matrix-Switch

**MSW-STYLE = 1 Tasten werden in HEX übertragen 8x8 Matrix**

```
MSW=2 0000000000000001  Taste 0.0 wurde gedrückt
MSW=2 000000000000000F  alle Tasten in der ersten Reihe wurden gedrückt
MSW=2 0000000000000000  Taste(n) losgelassen
  |   | |              |
  |   | |               -- Taste 0.0 bis 0.8
  |   |  --------- Taste 8.0 bis 8.8
  |    ----------- IOW-ID
   -------------- Kennung für MWS Mitteilung
```

**MSW-STYLE = 1 Tasten werden in HEX übertragen 16x8 Matrix**

```
MSW=2 0000000000000000 0000000000000000
  |   | |              | |              |
  |   | |              | |               -- -- Taste 9.0 bis Taste 9.8
  |   | |              |  -- Taste 16.0 bis Taste 16.8
  |   | |               -- Taste 0.0 bis 0.8
  |   |  --------- Taste 8.0 bis 8.8
  |    ----------- IOW-ID
   -------------- Kennung für MWS Mitteilung
```

**MSW-STYLE = 2 Tasten werden dezimal übertragen 8x8 Matrix**

```
MSW=2 0 0 0 0 0 0 0 1    Taste 0.0 wurde gedrückt
MSW=2 0 0 0 0 0 0 0 255  alle Tasten in der ersten Reihe wurden gedrückt
MSW=2 0 0 0 0 0 0 0 0    Taste(n) losgelassen
  |   | |              |
  |   | |               -- Taste 0.0 bis 0.8
  |   |  --------- Taste 8.0 bis 8.8
  |    ----------- IOW-ID
   -------------- Kennung für MWS Mitteilung
```

**MSW-STYLE = 2 Tasten werden dezimal übertragen 16x8 Matrix**

```
MSW=2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  |   | |              | |              |
  |   | |              | |               -- Taste 9.0 bis 9.8
  |   | |              |  -- Taste 16.0 bis 16.8
  |   | |               -- Taste 0.0 bis 0.8
  |   |  --------- Taste 8.0 bis 8.8
  |    ----------- IOW-ID
   -------------- Kennung für MWS Mitteilung
```

# 10 IOWarrior bezogene Kommandos Port-IN (PIN)

## 10.1 IOW-PIN-ON [IOW-ID]

```
Beispiel: IOW-PIN-ON 1
IOWPINON{}=ACK
```

## 10.2 IOW-PIN-OFF [IOW-ID]

```
Beispiel: IOW-PIN-OFF 1
IOWPINOFF{}=ACK
```

## 10.3 IOW-GET-PIN-STYLE [IOW-ID]

```
Beispiel: IOW-GET-PIN-STYLE 1
IOWGETPINSTYLE{}=0
```

## 10.4 IOW-SET-PIN-STYLE [IOW-ID] [STYLE] [saveUser]

```
Beispiel: IOW-SET-PIN-STYLE 1
IOWSETPINSTYLE{}=ACK
IOWSETPINSTYLE{}=150 Expect value(2) PIN-Output Style (0-1)
```

Alle Clients die diese Funktion aktiviert haben, bekommen bei jedem Event (Änderungen an den Ports) automatisch eine Nachricht vom Server.

**STYLE=0 (dezimal)**

```
PIN=1 255 255 255 255 ... 255
    | |   |   |   |       |
    | |   |   |   |        -- Port(6) IOW56
    | |   |   |   |                   ...
    | |   |   |    -- Port(3) IOW40   IOW56
    | |   |    ------ Port(2) IOW40   IOW56
    | |    -- Port(1) IOW24   IOW40   IOW56
    |  ------ Port(0) IOW24   IOW40   IOW56
     -- IOW-ID
```

**STYLE=1 (Hex)**

```
PIN=1 FFFFFFFF...FF
    | | | | |    |
    | | | | |     --------- Port(6) IOW56
    | | | | |                   ...
    | | | |  ------ Port(3) IOW40   IOW56
    | | |  -------- Port(2) IOW40   IOW56
    | |  -- Port(1) IOW24   IOW40   IOW56
    |  ---- Port(0) IOW24   IOW40   IOW56
     -- IOW-ID
```

Event ist nicht Zeitnah, kann somit nicht für Zeitmessungen verwendet werden.
Für solche Anwendungen ist besser der Capture-Timer geeignet.

# 11 IOWarrior bezogene Kommandos Capture Timer

## 11.1 IOW-CAPTURE-ON [IOW-ID]

```
Beispiel: IOW-CAPTURE-ON 6
IOWCAPTUREON{}=ACK
IOWCAPTUREON{}=142 CAPTURE-PORT not set
```

## 11.2 IOW-CAPTURE-OFF [IOW-ID]

```
Beispiel: IOW-CAPTURE-OFF 6
IOWCAPTUREOFF{}=ACK
```

## 11.3 IOW-GET-CAPTURE-PORT [IOW-ID]

```
Beispiel: IOW-GET-CAPTURE-PORT 6
IOWGETCAPTUREPORT{}=
```

## 11.4 IOW-SET-CAPTURE-PORT [IOW-ID][PORT] [saveIOW]

```
Beispiel: IOW-SET-CAPTURE-PORT 6 2
IOWSETCAPTUREPORT{}=ACK
IOWSETCAPTUREPORT{}=140 Expect value(2) for CAPTURE-PORT (1-3)
IOWSETCAPTUREPORT{}=141 Can't change Capture-Port Function is in use

Port = 1 = Port(A)
Port = 2 = Port(B)
Port = 3 = PORT(A)+(B)
```

## 11.5 IOW-GET-CAPTURE-STYLE [IOW-ID]

```
Beispiel: IOW-GET-CAPTURE-STYLE 6
IOWGETCAPTURESTYLE{}=0
```

## 11.6 IOW-SET-CAPTURE-STYLE [IOW-ID][STYLE] [saveUser]

```
Beispiel: IOW-SET-CAPTURE-STYLE 6 1
IOWSETCAPTURESTYLE{}=ACK
IOWSETCAPTURESTYLE{}=143 Expect value(2) Capture-Output Style (0-1)
```

**Alle Clients die diese Funktion aktiviert haben, bekommen bei jedem Event (Änderungen an den Capture-Pins) eine Nachricht automatisch vom Server gesendet.**

**STYLE=0 (dezimal)**

```
CAP=6 1 1 11736905 11713457
CAP=6 1 2 11736905 11759558
    | | | |         |
    | | | |          -- RISE-Timer (0-16777215)
    | | |  -- FALL-Timer (0-16777215)
    | |  -- FLAG( 1-15)
    |  -- Port 1(A) / 2(B)
     -- IOW-ID
```

**STYLE=1 (Hex)**

```
CAP=6 1 1 C01D12 C01BF0
CAP=6 1 F C01D12 C08960
    | | | |         |
    | | | |          -- RISE-Timer (0-FFFFFF)
    | | |  -- FALL-Timer (0-FFFFFF)
    | |  -- FLAG( 1-F)
    |  -- Port 1(A) / 2(B)
     -- IOW-ID
```

```
Flag:
Bit(0) 1 = falling edge dedected
Bit(1) 2 = rising  edge dedected
Bit(2) 4 = falling edge overrun
Bit(3) 8 = rising  edge overrun
```

# R Release Beschreibung

**Version 1-2-001**: (27) Kommandos (15.11.2008)
- Server Steuerung
- IOW LCD
- IOW LED-Matrix

**Version 1-2-005**: (41) Kommandos (17.11.2008)
- Port Read / Write
- Fehler (Server hängt sporadisch beim Verbinden neuer Clients) behoben

**Version 1-2-008**: (53) Kommandos (18.11.2008)
- IOW I2C

**Version 1-2-013**: (65) Kommandos (24.11.2008)
- Server erkennt hinzufügen und entfernen eines IOW
- RC5
- Switch-Matrix
- send Message
- LED-Matrix (Laufschrift)

**Version 1-3-025**: (79) Kommandos (22.02.2009)
- Programm wurde kompl. überarbeitet, ist dadurch nicht mehr mit den vorherigen Versionen 1-2-xxx kompatibel !!
- SPI Funktion integriert
- Tags können den Kommandos beigefügt werden
- NAK Fehlermeldungen durch ERROR-CODE ersetzt

**Version 1-3-036**: (81) Kommandos (07.05.2009)
- ID Reservirung
- Rechteverwaltung
- kleinere Fehler behoben

**Version 1-4-001**: (106) Kommandos (23.09.2009)
- neue Feature der IOW24/40 Version 1.0.3.0 eingefügt
- Funktion CAP wurde hinzugefügt
- Funktion PIN wurde hinzugefügt
- KEY- und TRIAL- Nutzung wurde implementiert

**Version 1-4-005**: (106) Kommandos (01.06.2011)
- kleinere Fehler behoben

**geplante Erweiterungen:**
- verschlüsselte Übertragung
- Kommandos für die Fern-Wartung
- Fern-Wartungs-Konsole

# Z Datei Beschreibungen

## (NAK) Fehlerbeschreibungen :

Datei **ERROR_MESSAGE.EN.txt** kann beliebig angepasst werden z.B. durch übersetzt in eine andere Sprache.

Die dreistellige Zahl gibt den Fehlercode an, der dahinter stehende Fehlertext beschreibt diesen.
Code und Text sind durch ein (;) `chr(59)` als Trennzeichen getrennt.

```
001;Unrecognized command
002;Access denied
005;No IOWarrior available
006;ERROR
007;Selected IOW does not support this feature
011;The number of Clients > 1
012;Used value for Client-ID not in use
013;Expect value for $UserName does not exist
018;Expect value (0-2) of NAK-Style
020;File can not be opened
021;File can not be saved
022;UserName is unknown
023;Usersetting File not exists
025;Used value(1) for Client-ID not in use
026;Expect value(2) for $MSG does not exist
029;Expect value (1-3) for LogLevel
030;Expect value (0-2) for ShowLevel
031;Expect value (1-500) for Delay
036;Expect value(1) for IOW-Device-ID (1-64)
037;Expect value(1) for IOW-Device-ID (x) not in use
041;Function on IOW-Device not ON
043;Expect value(2) of LCD-Position (0-103)
044;Expect value(3) of LCD-Text does not exist
047;Function is supported by a marquee blocked
048;Function is not used by this client
049;Function is used by several clients, Marquee can only use this feature
051;Expect value(2) of LED-Matrix columns (1-64)
052;Expect value(3) of $HEX-String (00-FF)
054;Expect value(2) $MarqueeText does not exist
057;Expect value(2) of LED-Matrix SIZE (1-64)
060;Expect value(2) Marquee interval (20-200)
063;MarqueeText_Buffer is empty
071;Expect value(2) for Port IOW24(0-1) IOW40(0-3) IOW56(0-6)
072;Expect value(3) for set ( 0-255 or $01001111 )
073;Expect value(4) for Bit ( 0-7 )
075;IOW Write ERROR
081;I2C-Device doesn't respond
082;Expect value(2) for I2C-Addr. (2-254)
083;Expect value(3) num Bytes (1-128)
084;Expect value(3) String does not exist
085;I2C-Device write error
086;I2C-Device not acknowledged
087;Expect value(3) Data Bytes (0-255)
088;Expect value(2) wait of I2C (0-500) ms
091;Expect value(2) for clock (100/400/50) or (0/1/2)
092;IOW24 or IOW40 can't change clock only supports 100 kHz
093;Can't change I2C-function is in use
094;Expect value(2) for Protocol (0/1) 0=I2C 1=SHT
095;Expect value(2) for Timeout (0-255)
096;Expect value(2) for PullUps (0/1) 0=enablde 1=disabled
097;Commands run not with selected protocol
101;IOW40 or IOW56 not supports RC5
111;IOW24 not supports Switch-Matrix
112;Expect value(2) Output Style (0-2)
113;Expect value(2) Matrix-Type (0-1)
114;Can't change Matrix-Type function is in use
121;IOW40 not supports SPI
122;Expect value(2) for Clock (0-3)
123;Expect value(2) for Clock (1-255)
124;Expect value(2) for MODE (0-3)
125;Expect value(2) num Bytes (1-256)
126;Can't change SPI-clock function is in use
127;Can't change SPI-mode function is in use
128;Expect value(2) Data-String not exists
129;Expect value(3) num Block (1-256)
140;Expect value(2) for CAPTURE-PORT (1-3)
141;Can't change Capture-Port Function is in use
142;CAPTURE-PORT not set
143;Expect value(2) Output Style (0-1)
```

# Z Datei Beschreibungen

## Socket_Server.set

Die Datei wird beim starten eingelesen und beim beenden des Socket-Server überschrieben !
Änderungen sind somit nur im nicht gestarteten Zustand wirksam.

```
MAX_CLIENT = 2          'Anzahl der Clients die maximal den Server gleichzeitig verwenden dürfen
DELAY      = 5          'Wert ermöglicht auf das Tempo der Verarbeitung Einfluss zu nehmen.
PORT       = 5100       'Port des Socket-Server
SHOW_LEVEL = 2          'siehe set-show-level
LOG_LEVEL  = 2          'siehe set-log-level
NAKSTYLE   = 2          'Default Wert für die Fehlerdarstellung
KEY_REG    = 205712     'Prüfsumme die vom Reg-Server ermittelt und übertragen wurde.
KEY_REG_ID = 001        'Laufende Nummer der Lizenz (1-999),  wird vom Reg-Server vergeben.
TRIAL_TXT  = ||||||     'Zwischenspeicher für TRIAL-Registrierung
W_TOP      = 61         |
W_LEFT     = 288        |  ' Position und Größe des Programm-Fenster
W_HEIGHT   = 195        |  ' wird beim Start einmalig verwendet.
W_WIDTH    = 529        |
ERROR_TXT  = ERROR_MESSAGE.EN.txt        'Dateiname ERROR-Text
```

## UserAccount.set

```
MY-MP3|11111111111111111111;111111;11111111111111111111;111111111;11111;111111111;1111111111;11;111111;1111;111111
UNKNOWN|01111111111111111111;111111;11111111111111111111;111111111;11111;111111111;1111111111;11;111111;1111;111111
```

Keine Leer oder Steuerzeichen erlaubt.
Trennzeichen „|“ chr(124) und „;“ chr(59).

```
UserName|Block(1);Block(2);Block(3);Block(4);Block(5);Block(6);Block(7);Block(8);Block(9);Block(10);Block(11);xxx
```

```
Block(1) = 1111111111111111111xxx
           |                  |
           |                   -- (20) ListIOW-Port-Devs
            -- (1) Server-Close
```

Die Blöcke und die Kommandos entsprechen der Nummerierung in dieser Anleitung

```
z.B. 1.1   Server-CLose        (Block=1 / Recht=1)
     1.20  List-IOW-Port-Devs  (Block=1 / Recht=20)
```

Da zukünftige Versionen eine Erweiterung der Rechte zu folge haben,
werden die bestehenden Rechte übernommen und mit Default-werten erweitert.
**Neue Rechte(Funktionen) in einem bestehenden Block werden immer rechts angehängt.**
**Neue Funktionen mit einer neuen Gruppe werden als neuer Block rechts angehängt.**

Fehlende Blöcke oder Rechte werden von links nach recht mit Default-Werten
(1) ergänzt oder mit (0) überschrieben.

Die Default-Werte sind abhängig von den frei geschalteten IOW-Funktionen,
ist eine dieser Funktion im Key deaktiviert sind alle Rechte in diesem Block=0.
Bestehende Rechte werden in diesem Block dann mit (0) überschrieben.

## IDBOOKING.SET

```
6|1501|00000077
46|1501|00002980
```

```
ID|Type|Seriennummer
```

Keine Leer oder Steuerzeichen erlaubt, Trennzeichen „|“ chr(124).

# Z Datei Beschreibungen

## USER_USERNAME.set

```
MSW-STYLE=0
CAP-STYLE=0
NAK-STYLE=2
PIN-STYLE=1
```

Werte die mit den Kommandos **IOWSET...STYLE** gesetzt werden können.
Keine Leer oder Steuerzeichen erlaubt.

siehe auch: **SaveUserSetting** / **LoadUserSetting**

## TYPE_SerienNummer.set
**(1501_00000077.set)**

```
PORTDEVS=255|255|
I2CWAIT=5
I2CCLOCK=0
I2CPROTOCOL=0
I2CPULLUPS=0
I2CTIMEOUT=0
SPICLOCK=0
SPIMODE=0
MQSIZE=0
CAPPORT=3
MSWTYPE=0
SHT_C1.0=-2.0468
SHT_C1.1=-2.0468
SHT_C2.0=0.0367
SHT_C2.1=0.5872
SHT_C3.0=1.5955E-6
SHT_C3.1=0.00040845
SHT_T1.0=0.01
SHT_T1.1=0.01
SHT_T2.0=8E-5
SHT_T2.1=0.00128
SHT_D2.0=0.01
SHT_D2.1=0.04
SHT_D1=-40.1

---------------------
```

**SHT-Werte** bitte aus der Anleitung zum SHT-xx entnehmen.
Die Bezeichnung sind identisch (**C1,C2,C3,T1,T2,D1,D2**).

**xx.0 Wert für die hohe Auflösung**
**xx.1 Wert für die geringe Auflösung**

**Alle anderen Werte können sie den entsprechenden Kommandos IOWSET... entnehmen, die Bezeichnug sind mit den Kommandos identisch.**