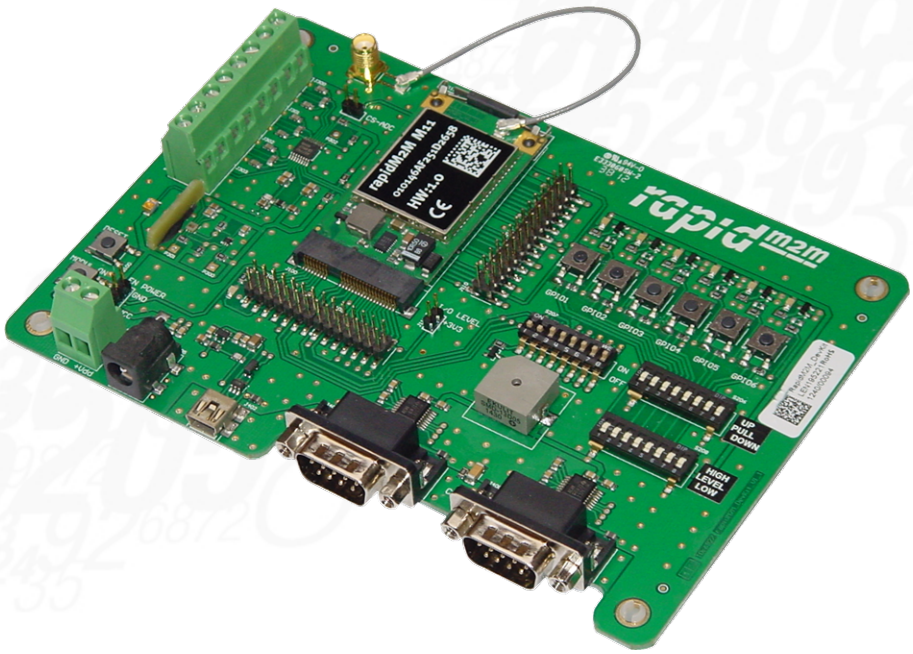
Betriebsanleitung für das rapidM2M Developer Paket M11



Rev. 02 gültig ab:

- Firmware Version: ---
- Modem Version: 03v002
- Hardwareversion: 1.0

Microtronics Engineering GmbH | www.microtronics.at
Hauptstrasse 7 | 3244 Ruprechtshofen | Austria
Tel +43 2756 77180 23 | Fax +43 2756 77180 33 | support@microtronics.at

Kapitel 1 Inhaltsverzeichnis

Betriebsanleitung für das rapidM2M Developer Paket M11 1

Kapitel 1 Inhaltsverzeichnis 3

Kapitel 2 Technische Daten 11

Kapitel 3 Allgemeine Angaben 13

3.1 Allgemeine Produktinformationen 13

3.2 Lieferumfang 13

3.3 Abmessungen 14

3.4 Gerätekennzeichnung 14

3.5 Aufbewahrung des Produkts 15

Kapitel 4 Funktionsprinzip 17

4.1 ALOHA-Übertragungsmodus 17

4.2 Auswahl des GSM-Netzes 18

4.2.1 Banlist 19

4.3 Script-Abarbeitung 19

4.3.1 Instruction List 19

4.3.2 Pawn 19

Kapitel 5 rapidM2M Developer Board 21

5.1 Übersicht 21

5.1.1 Blockschaltbild 23

5.1.2 RS232-Schnittstelle COM1 (J400) 23

5.1.3 Jumper zur Versorgung der Level Shifter (J111) 24

5.1.4 RS232-Schnittstelle COM2 (J401) 24

5.1.5 Stiftleiste auf der die Pins mit den ungeraden Nummern des M11 Moduls aufgelegt sind (J106 und J109) 25

5.1.6 Status-LED (LED100) 27

5.1.7 Versorgung (J105 bzw. J108) 27

5.1.7.1 Klemmleiste (J108) 27

5.1.7.2 DC-Hohlstecker (J105) 28

5.1.8 Jumper zum Aktivieren der Energieversorgung des M11 Moduls (J104) ... 28
5.1.9 Taster zum Einschalten des Wireless GSM Controllers am rapidM2M Modul M11 (S101) ... 28
5.1.10 Taste zum Reseten des Wireless GSM Controllers am rapidM2M Modul M11 (S100) ... 29
5.1.11 Aufnahme des M11 Moduls ... 29
5.1.12 Stiftleiste auf der die Pins mit den geraden Nummern des M11 Moduls aufgelegt sind (J107 und J110) ... 29
5.2 Schaltplan ... 31
5.2.1 Übersicht ... 31
5.2.2 Connector ... 32
5.2.3 USB RS232 ... 33
5.2.4 Sensoren ... 34
5.2.5 GPIO ... 35
5.3 Layout ... 36
5.4 Elektrische Installation ... 37
5.4.1 Hinweise zur Vermeidung elektrostatischer Entladungen (ESE) ... 37
5.4.2 Einsetzen des M11 Moduls ... 38
5.4.3 Anschluss des Sensors und der Versorgung ... 39
5.4.4 Anschluss der GSM-Antenne ... 40
Kapitel 6 rapidM2M Modul M11 ... 43
6.1 Technische Daten ... 43
6.2 Übersicht ... 44
6.2.1 Blockschaltbild ... 45
6.2.2 52pin Steckerleiste (X2) ... 46
6.2.3 Gerätekennzeichnung ... 47
6.3 Abmessungen ... 47
6.3.1 Abmessungen der 52pin Steckerleiste (X2) ... 48
6.4 Technische Details ... 48
6.4.1 Technische Details zur Energieversorgung ... 48
6.4.1.1 Versorgung über VIN ... 48

6.4.1.2 Versorgung über VAUX ... 48
6.4.2 ENCPU-Signal ... 48
6.4.3 Reset-Signal ... 49
6.4.4 Status-LED ... 49
6.4.5 Technische Details zu den Level Shiftern ... 50
6.4.6 Technische Details zur UART1 ... 50
6.4.7 Technische Details zur UART2 ... 51
6.4.7.1 Fehlercodes der UART2-Schnittstelle ... 51
Kapitel 7 Inbetriebnahme ... 53
7.1 Inbetriebnahme des Systems ... 53
7.2 Kommunikation mit dem Gerät testen ... 53
Kapitel 8 Benutzerschnittstelle am myDataweb-Server ... 55
8.1 Messstellenkonfiguration ... 55
8.1.1 Messstelle ... 55
8.1.2 Kommentar ... 55
8.1.3 Steuerung ... 56
8.1.4 Interface Kanäle 1-32 ... 56
8.1.4.1 Basis ... 56
8.1.4.2 Alarme ... 57
8.1.4.3 Trigger ... 57
8.1.5 Interface Kanäle 33-64 ... 57
8.1.5.1 Basis ... 57
8.1.6 Interface Ausgabekanäle 1-32 ... 57
8.1.7 Interface Ausgabekanäle 33-64 ... 58
8.1.8 Berechnete Kanäle ... 58
8.1.8.1 Basis ... 58
8.1.8.2 Berechnung ... 59
8.1.8.3 Alarme ... 59
8.1.9 Interne Kanäle ... 60
8.1.9.1 Basis ... 60

8.1.9.2 Alarme ... 60
8.1.9.3 Trigger ... 60
8.1.10 Serielle Schnittstelle ... 61
8.1.11 Alarmierung ... 66
8.1.12 Grundeinstellung ... 67
8.1.13 FTP Export Einstellungen ... 68
8.2 Gerätekonfiguration ... 69
8.2.1 Kommentar ... 69
8.2.2 Messgerät ... 69
8.2.3 Gerätespezifische Einstellungen ... 70
8.2.4 GPRS ... 70
Kapitel 9 Instruction List ... 71
9.1 Allgemein ... 71
9.1.1 Direkte Eingabe einer Instruction List ... 71
9.1.2 Verwendung eines Script-Templates ... 71
9.2 Implementierung-Details ... 72
9.2.1 Gemeinsame Elemente ... 72
9.2.2 IL-Syntaxelemente ... 74
9.3 Instruction List Fehlercodes ... 76
9.4 Erweiterungen ... 78
9.5 Geräteabhängige Implementierung ... 80
9.6 Beispiele ... 82
9.6.1 Sägezahngenerator ... 82
9.6.2 Wochentagbestimmung ... 82
Kapitel 10 Pawn Script ... 85
10.1 Allgemein ... 85
10.1.1 Direkte Eingabe eines Pawn Scripts ... 85
10.1.2 Verwendung eines Script-Templates ... 85
10.2 Allgemeine Syntax ... 85
10.2.1 Format ... 85

10.2.2 Optionale Semikolons ... 86
10.2.3 Kommentare ... 86
10.2.4 Bezeichner ... 86
10.2.5 Reservierte Schlüsselworte ... 86
10.2.6 Numerische Konstanten ... 87
10.2.6.1 Numerische Integer Konstanten ... 87
10.2.6.2 Numerische Gleitkomma-Konstanten ... 87
10.3 Variablen ... 87
10.3.1 Deklaration ... 87
10.3.2 Lokale Deklaration ... 87
10.3.3 Globale Deklaration ... 88
10.3.4 Statische lokale Deklaration ... 88
10.3.5 Statische globale Deklaration ... 88
10.3.6 Gleitkommawerte ... 88
10.4 Konstante Variablen ... 88
10.5 Array Variablen ... 89
10.5.1 Eindimensionales Array ... 89
10.5.2 Initialisierung ... 89
10.5.3 Progressive Initialisierung für Arrays ... 90
10.5.4 Mehrdimensionale Arrays ... 90
10.5.5 Arrays und der "sizeof"-Operator ... 90
10.6 Operatoren und Ausdrücke ... 91
10.6.1 Zeichenerklärung ... 91
10.6.2 Ausdrücke ... 91
10.6.3 Arithmetik ... 92
10.6.4 Bit-Manipulation ... 92
10.6.5 Zuweisung ... 93
10.6.6 Vergleichsoperatoren ... 94
10.6.7 Boolean ... 94
10.6.8 Sonstiges ... 95

10.6.9 Priorität der Operatoren ... 95
10.7 Anweisungen ... 97
10.7.1 Statement-Etikett ... 97
10.7.2 Zusammengesetzte Anweisungen ... 97
10.7.3 Ausdrucksanweisung ... 97
10.7.4 Leeres Statement ... 97
10.7.5 assert Ausdruck ... 97
10.7.6 break ... 98
10.7.7 continue ... 99
10.7.8 do Statement while (Ausdruck) ... 99
10.7.9 exit Ausdruck ... 100
10.7.10 for (Ausdruck 1 ; Ausdruck 2 ; Ausdruck 3) Statement ... 100
10.7.11 goto Etikett ... 101
10.7.12 if (Ausdruck) Statement 1 else Statement 2 ... 101
10.7.13 return Ausdruck ... 101
10.7.14 switch (Ausdruck) { case Liste } ... 101
10.7.15 while (Ausdruck) Statement ... 102
10.8 Funktionen ... 103
10.8.1 Funktionsargumente ("call-by-value" versus "call-by-reference") ... 104
10.8.2 Benannte Parameter versus positionsgebundene Parameter ... 105
10.8.3 Standardwerte von Funktionsargumenten ... 106
10.9 Erweiterungen ... 108
10.9.1 Konstanten ... 108
10.9.2 Geräteabhängige Kanalnummern ... 109
10.9.3 Funktionen ... 110
10.9.3.1 Core-Funktionen ... 110
10.9.3.2 Consolen-Funktionen ... 113
10.9.3.3 Mdn-Funktionen ... 114
10.9.3.4 Mathematische Funktionen ... 117
10.9.4 Pawn Script Fehlercodes ... 119

10.10 Beispiele ... 121
10.10.1 Sägezahngenerator ... 121
10.10.2 Wochentagbestimmung ... 121
10.10.3 Durchflussberechnung mit Tabelle ... 123
10.11 Unterschiede zu C ... 125
Kapitel 11 Fehlersuche und Behebung ... 129
11.1 Allgemeine Probleme ... 129
11.2 Log-Eingräge und Fehlercodes ... 132
11.2.1 Modemfehler ... 135
11.3 Auswerten des Gerätelogs ... 138
11.3.1 Auswerten des Gerätelogs am myDataweb-Server ... 138
11.3.2 Auswerten des Gerätelogs mittels myDatanetDeviceConfig ... 139
Kapitel 12 Ersatzteile und Zubehör ... 141
12.1 rapid M2M-Module ... 141
12.2 Antennen ... 141
12.3 Versorgung ... 141
12.4 Adapter ... 141
12.5 Sonstiges Zubehör ... 141
Kapitel 13 Applikationsbeispiele ... 143
13.1 Laternensteuerung ... 143
13.1.1 Anforderung ... 143
13.1.2 Erforderliche Konfiguration ... 144
13.1.3 Instruction List ... 145
13.1.4 Kommunikation auf der Schnittstelle ... 146
Kapitel 14 Dokumentenhistorie ... 147
Kapitel 15 Kontaktinformationen ... 149

Kapitel 2 Technische Daten

Hinweis: *Die folgenden technischen Spezifikationen gelten für das rapidM2M Developer Board mit eingesetztem rapidM2M Modul M11 . Eine detaillierte technische Beschreibung des M11 Moduls finden Sie im Kapitel "Technische Daten" auf Seite 43.*

Spannungsversorgung	5...32 VDC , max. t.b.d. mA@12V
Gehäuse	Gewicht: 134g (inkl. rapidM2M Modul M11) Abmessungen (BHT): 160 x 25 x 110mm (ohne Antenne und Anschlusskabel)
Betriebstemperatur	-5...+40°C, 15...90%rH nicht kondensierend
Lagertemperatur	-20...+70°C
Anzeige	LED zur Signalisierung des Betriebszustandes
Antennenanschluss	SMA
Schnittstellen	1x RS232 für den Anschluss eines digitalen Sensors • Baudrate: 300-115200 • Stopbits: 1,2 • Parität: N, E, O • Datenbits: 7, 8 Modi: • ASCII Anzahl der verarbeitbaren Interface Kanäle • Eingänge: 64 • Ausgänge: 64 1x RS232 für die Ausgabe von Debug-Informationen Zusätzliche Informationen finden Sie unter "RS232-Schnittstelle COM1 (J400)" auf Seite 23 und "RS232-Schnittstelle COM1 (J400)" auf Seite 23.
Datenspeicher	interner Flash-Speicher für bis zu 2620 Messzyklen
Datentyp	f32 (32 Bit floating point)
Datenübertragung	mittels GSM/GPRS Quad-Band Modem an den jeweiligen myDataweb-Server
SIM	Das rapidM2M Modul M11 ist mit einem integrierten SIM-Chip versehen.
Monatliches Datenvolumen	t.b.d. bei 2min Messintervall und 120min Übertragungsintervall

Kapitel 3 Allgemeine Angaben

3.1 Allgemeine Produktinformationen

Es handelt sich um ein Developer Paket, das den Kunden bei der Integration des M11 Moduls in eigene Geräte unterstützen soll. Das rapidM2M Developer Board ist mit standardisierten Steckern, Stiftleisten, Klemmleisten, Tastern und LEDs (siehe "Übersicht" auf Seite 21) ausgestattet, um einen einfachen und schnellen Aufbau von Testschaltungen zu gewährleisten. Bei dem im Lieferumfang des rapidM2M Developer Paket M11 enthaltenen rapidM2M Modul M11 handelt es sich um ein Modul zur Aufzeichnung und Übertragung von Messdaten, die aus der ASCII-Datenmessage eines digitalen Sensors entnommen werden. Dafür stehen 64 Interface Kanäle zur Verfügung. Die physikalische Verbindung zwischen dem rapidM2M Developer Board und dem digitalen Sensor erfolgt dabei über eine RS232 Schnittstelle.

> ***Hinweis:*** *Bitte beachten Sie für die spätere Integration des M11 Moduls in eine eigene Hardware, dass der RS232-Pegelwandler ins rapidM2M Developer Board integriert ist. Der Ausgangspegel der UART des M11 Moduls ist abhängig von der Versorgung der Level Shifter des Moduls (siehe "Technische Details zu den Level Shiftern" auf Seite 50).*

Über dieselbe RS232 Schnittstelle ist es auch möglich, Daten an den digitalen Sensor zurück zu liefern. Dazu stehen 64 Ausgabekanäle zur Verfügung. Die auszugebenden Daten können entweder über die Eingabemaske am myDataweb festgelegt werden oder mittels Script (siehe "Instruction List" auf Seite 71 bzw. "Pawn Script" auf Seite 85) vom rapidM2M Modul M11 nach jedem Messzyklus berechnet werden. Die Messdaten der Interface Kanäle werden in einem internen Datenspeicher zwischengespeichert und in einem frei wählbaren Intervall drahtlos an eine zentrale Stelle übermittelt. Über dieselbe Verbindung erfolgt auch die Konfiguration des M11 Moduls. Das Modul ist mit einem integrierten SIM-Chip versehen.

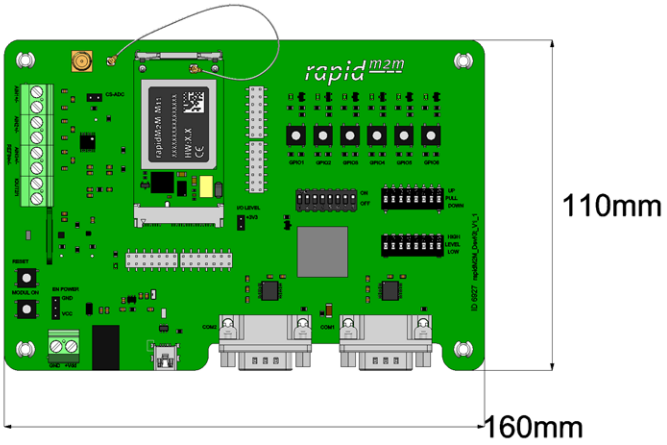
3.2 Lieferumfang

Zum Standardlieferumfang des Developer Pakets (300070) gehören:

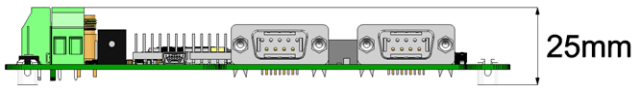
- rapidM2M Developer Board
- rapidM2M Modul M11 (300068)
- Steckernetzgerät 6W 12V
- Standantenne Quad Band SMA (300085)
- 10 x Steckbrücke
- U.FL Antennenkabel
- myDatanet Tool Pen (206.646)
- 4 x Distanzhülsen
- Kurzanleitung (300084)

Weiteres Zubehör wie M2M-Module, Antennen, Netzteile, Sensoren usw. je nach Bestellung. Diese bitte anhand des Lieferscheins prüfen.

3.3 Abmessungen



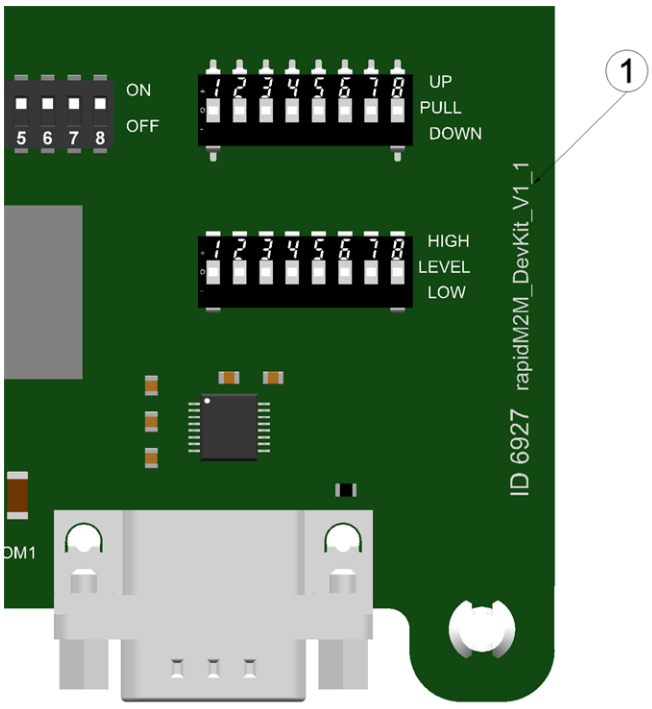
Abmessungen: Breite und Tiefe



Abmessungen: Höhe

3.4 Gerätekennzeichnung

Die Angabe, um welche Hardwareversion des Developer Boards es sich handelt, finden Sie auf der rechten Seite der Platine in der Nähe der Dip-Switches S208 und S206.



Hardwarekennzeichnung des Developer Boards

1	Hardwareversion

Wichtig für alle Rückfragen und Ersatzteilbestellungen ist die richtige Angabe der Typenbezeichnung und der Hardwareversion. Nur so ist eine einwandfreie und schnelle Bearbeitung möglich.

3.5 Aufbewahrung des Produkts

Zur Aufbewahrung des Developer Pakets stellen Sie sicher, dass alle relevanten Daten zum myDataweb-Server übertragen wurden. Da das rapidM2M Modul M11 den Anschluss einer Taste zum Auslösen einer Übertragung der zwischengespeicherten Messdaten nicht unterstützt, müssen Sie unter Umständen bis zur nächsten planmäßigen Datenübertragung warten, bis alle Daten zum myDataweb–Server gesendet wurden. Dies gilt speziell für die Verbindungsart „Intervall“ (siehe "Grundeinstellung" auf Seite 67). Wurde die Verbindungsart „Intervall & Wakeup“ ausgewählt, können Sie den ALOHA–Übertragungsmodus über den myDataweb–Server auslösen, um die Daten zu übertragen. Kontrollieren Sie anschließend erneut, ob nun alle relevanten Daten übertragen wurden. Bei der Verbindungsart „online“ werden die ermittelten Messdaten sofort zum myDataweb–Server übertragen. Die Daten am Server sind immer aktuell und das Gerät kann jederzeit abgeschaltet werden. Als nächstes trennen Sie das rapidM2M Developer Board von der Versorgungsspannung. Wenn möglich, sollte auch die Versorgung des digitalen Sensors abgeschaltet werden bevor Sie die RS232 Verbindung zw. dem rapidM2M Developer Board und dem digitalen Sensor trennen. Anschließend kann die Antenne entfernt werden. Bewahren Sie das rapidM2M Developer Paket M11 inkl. Zubehör und Anschlusskabel in der Originalverpackung auf.

Die Konfiguration und die zuletzt ermittelten Daten bleiben erhalten. Die Systemzeit läuft allerdings ohne Energieversorgung nicht weiter. Das bedeutet, dass nach der Wiederinbetriebnahme eine Zeitsynchronisation erforderlich ist, damit die Zeitstempel der Mess- und Log-Daten wieder korrekt sind. Diese erfolgt automatisch bei der ersten Verbindung zum myDataweb–Server.

Kapitel 4 Funktionsprinzip

4.1 ALOHA-Übertragungsmodus

Beim ALOHA-Übertragungsmodus handelt es sich um einen speziellen Verbindungsmodus bei dem das rapidM2M Modul M11 für eine über den Konfigurationsabschnitt „Grundeinstellungen“ konfigurierbare Zeit (siehe „Aloha/Wakeup Dauer“ im Kapitel "Grundeinstellung" auf Seite 67) eine Verbindung zum myDataweb–Server aufbaut.

Auslösen des ALOHA–Übertragungsmodus:

- Über den myDataweb–Server, wenn die Verbindungsart „Intervall & Wakeup“ (siehe "Grundeinstellung" auf Seite 67) ausgewählt wurde.



Dass sich ein Gerät im ALOHA-Übertragungsmodus befindet, wird in der Messgeräteliste (siehe "Benutzerhandbuch für myDataweb-Server" 206.886) durch eine Sprechblase mit der Beschriftung „Aloha“ angezeigt.



Durch Klicken auf die Sprechblase mit der Beschriftung „Aloha“ öffnen Sie das ALOHA–Datenfenster (siehe "Benutzerhandbuch für myDataweb-Server" 206.886). Es beinhaltet den internen Messwert „GSM Stärke“. Im ALOHA–Datenfenster werden immer die Messwerte der letzten 30min., unter Umständen somit auch jene einer vorangegangenen ALOHA-Übertragung, angezeigt. Die ALOHA–Daten werden unabhängig von der normalen Messwerterzeugung alle 3sec. generiert und somit auch nicht zusammen mit den Standardmessdaten abgespeichert.

Wird während des ALOHA–Übertragungsmodus ein Messdatensatz aufgezeichnet, wird dieser sofort zum myDataweb–Server übertragen und in den Standardmessdaten abgespeichert.

Ist auf der Serveroberfläche ein Anzeigeelement sichtbar, das einen der Messwerte des Geräts verwendet, werden die Daten auch nach jeder Messwerterzeugung an den myDataweb–Server übertragen. Erfolgt die Messung öfters als die Aufzeichnung („Messintervall“ ungleich 0 und/oder „Burst Intervall“ ungleich 0) (siehe "Beispiel zur Erklärung Aufzeichnungs-, Mess-, Burstintervall" auf Seite 68), kommt es dadurch zu einem erhöhten Datentransfer. Diese zusätzlichen Daten werden aber weder am Server noch im Gerät dauerhaft gespeichert, sondern am Server nur zur Aktualisierung der Anzeigeelemente (siehe "Benutzerhandbuch für myDataweb-Server" 206.886) verwendet.

4.2 Auswahl des GSM-Netzes

Da das rapidM2M Developer Paket M11 mit einem SIM-Chip ausgestattet ist, der eine Mobilfunkverbindung über eine Vielzahl internationaler Serviceprovider gewährleistet (siehe www.microtronics.at/footprint), ist eine Auswahl des GSM-Netzes in das sich das Gerät einbuchen soll erforderlich. Diese erfolgt automatisch vom Gerät. Dabei wird auch berücksichtigt, wenn die Verbindung über ein bestimmtes GSM-Netz zwar möglich ist, es aber bei Verwendung dieses GSM-Netzes immer wieder zu Übertragungsfehlern kommt. Ein solches GSM-Netz wird in eine "Banlist" eingetragen und beim nächsten Einbuchungsversuch ignoriert.

Folgende Bedingungen führen dazu, dass sich das rapidM2M Developer Paket M11 in ein GSM-Netz einbucht:

- PowerOn
- Ablauf des Übertragungsintervalls, wenn die Verbindungsart „Intervall“ (siehe "Grundeinstellung" auf Seite 67) ausgewählt wurde.
- Aktivieren der Verbindungsart „Intervall & Wakeup“ oder „Online“ (siehe "Grundeinstellung" auf Seite 67)

1. Das rapidM2M Developer Paket M11 überprüft, ob es beim letzten Einbuchungsversuch zu Fehlern kam.

   Keine Fehler:

   Das rapidM2M Developer Paket M11 bucht sich in dasselbe GSM-Netz, das auch beim letzten Versuch verwendet wurde, ein.

   Fehler:

   Weiter mit Schritt 2

   > ***Hinweis:*** *Ein Übertragungsfehler ist nicht gleichbedeutend mit einem Fehler beim Einbuchen in das GSM-Netzwerk. Es ist möglich, dass das Einbuchen in das GSM-Netzwerk korrekt funktioniert hat, aber es dennoch zu einem Übertragungsfehler kommt.*

Die folgenen Schritte werden nur ausgeführt, wenn bei Schritt 1 ein Fehler erkannt wurde:

2. Das Gerät sucht nach GSM-Netzen in seinem Empfangsbereich und erstellt eine Liste der verfügbaren GSM-Netze.
3. Die Liste mit den verfügbaren GSM-Netzen wird durchsucht und das erste GSM-Netz gewählt, das sich nicht auf der "Banlist" befindet und zu einem der vom integrierten SIM-Chip unterstützten Serviceprovider (siehe www.microtronics.at/footprint) gehört.

   > ***Hinweis:*** *Eine Erklärung dazu unter welchen Bedingungen ein GSM-Netz in die "Banlist" eingetragen wird, finden Sie unter "Banlist" auf Seite 19.*

4.2.1 Banlist

Die Banlist bietet Platz für 5 Einträge und wird nicht flüchtig gespeichert. D.h. sie steht auch nach einem PowerOn weiter zur Verfügung.

Kommt es zu einem "BEARER GPRS FAILED"-Fehler (siehe "Modemfehler" auf Seite 135) wird ein interner Zähler um 2 erhöht. Wurde eine Übertragung fehlerfrei abgeschlossen, d.h. alle Daten wurden übertragen, wird dieser interne Zähler um 1 reduziert. Erreicht der interne Zähler den Wert 6 , wird das aktuelle GSM-Netzwerk in die Banlist eingetragen.

Die Banlist wird gelöscht, wenn aus der Liste der verfügbaren GSM-Netze im Empfangsbereich unter Berücksichtigung der Banliste kein GSM-Netzwerk ausgewählt werden kann. Nach dem Löschen der Banlist wird erneut nach GSM-Netzwerken im Empfangsbereich gesucht und die Netzauswahl mit der jetzt leeren Banlist durchgeführt (siehe Schritt 3 im Kapitel "Auswahl des GSM-Netzes" auf Seite 18).

4.3 Script-Abarbeitung

4.3.1 Instruction List

Zu jedem Messzeitpunkt, nachdem alle Messwerte erzeugt wurden, wird die Instruction List durchlaufen. Sollte dabei ein Fehler auftreten, stoppt die Scriptausführung und trägt den entsprechenden Fehlercode in das Gerätelog ein (siehe "Instruction List Fehlercodes" auf Seite 76). Erst danach erfolgt, falls auch das Aufzeichnungsintervall abgelaufen ist, die Aufzeichnung der gemessenen sowie durch die Instruction List berechneten Daten.

> ***Hinweis:*** *Wird mittels Instruction List direkt auf einen Ausgang geschrieben, überschreibt der ermittelte Wert den über die Eingabemaske am myDataweb-Server eingegebenen Stellwert.*

4.3.2 Pawn

Das PAWN-Script muss folgende 2 Funktionen enthalten:

main();

Diese Funktion wird beim PowerOn und beim Austausch des PAWN-Scripts ausgeführt. Sie sollte alle Initialisierungen enthalten, die nur ein einziges Mal beim Programmstart durchgeführt werden müssen.

Mdn_CtrlFinsih();

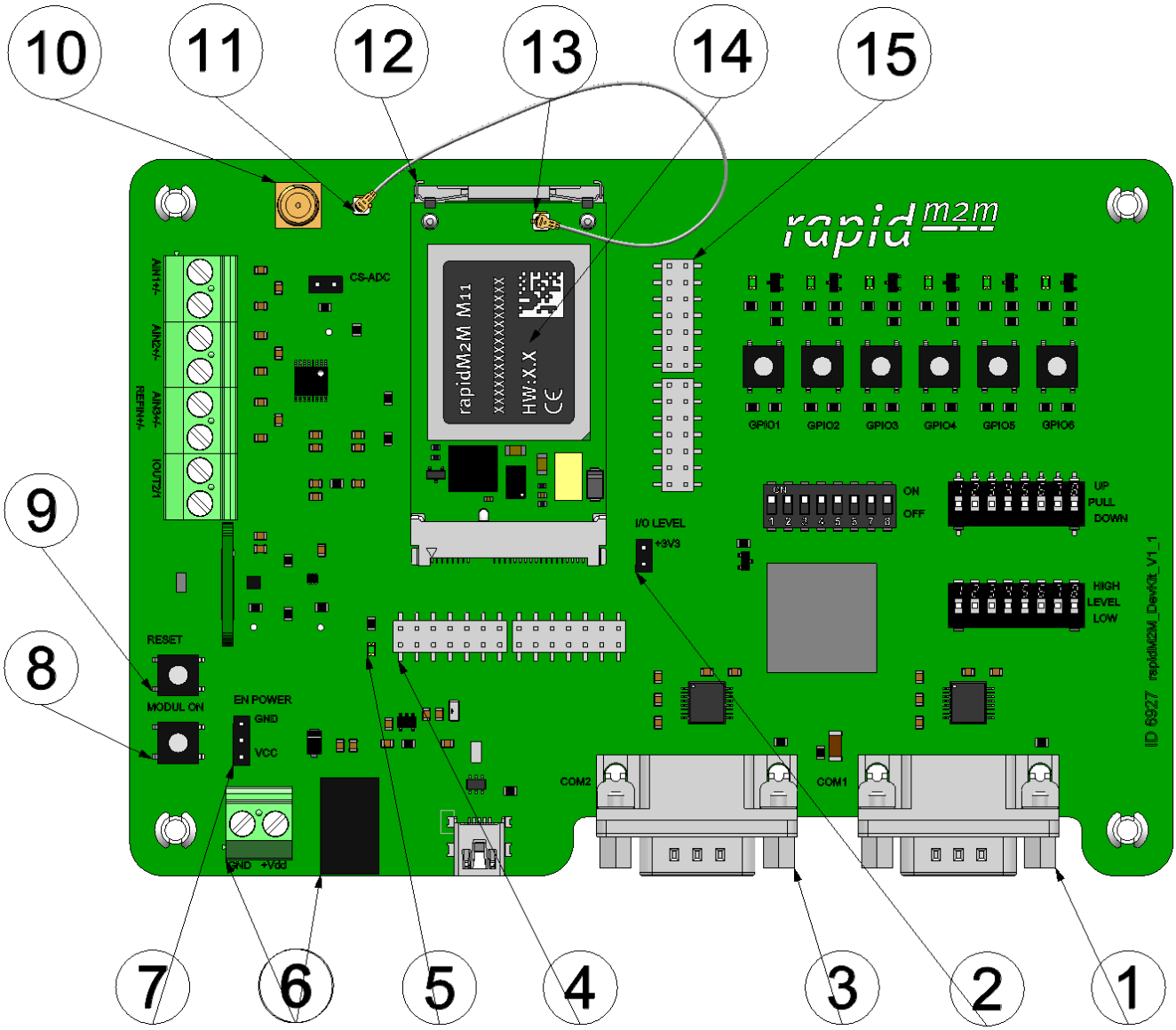
Diese Funktion wird zu jedem Messzeitpunkt, nachdem alle Messwerte erzeugt wurden, aufgerufen. Sie sollte alle Berechnungen und Funktionen enthalten, die zyklisch durchgeführt werden sollen.

> ***Hinweis:*** *Wird mittels Pawn Script direkt auf einen Ausgang geschrieben, überschreibt der ermittelte Wert den über die Eingabemaske am myDataweb-Server eingegebenen Stellwert.*

Sollte bei der Scriptausführung ein Fehler auftreten, wird die Scriptausführung gestoppt und deaktiviert. Zudem wird in das Gerätelog der entsprechende Fehlercode eingetragen (siehe "Pawn Script Fehlercodes" auf Seite 119).

Kapitel 5 rapidM2M Developer Board

5.1 Übersicht

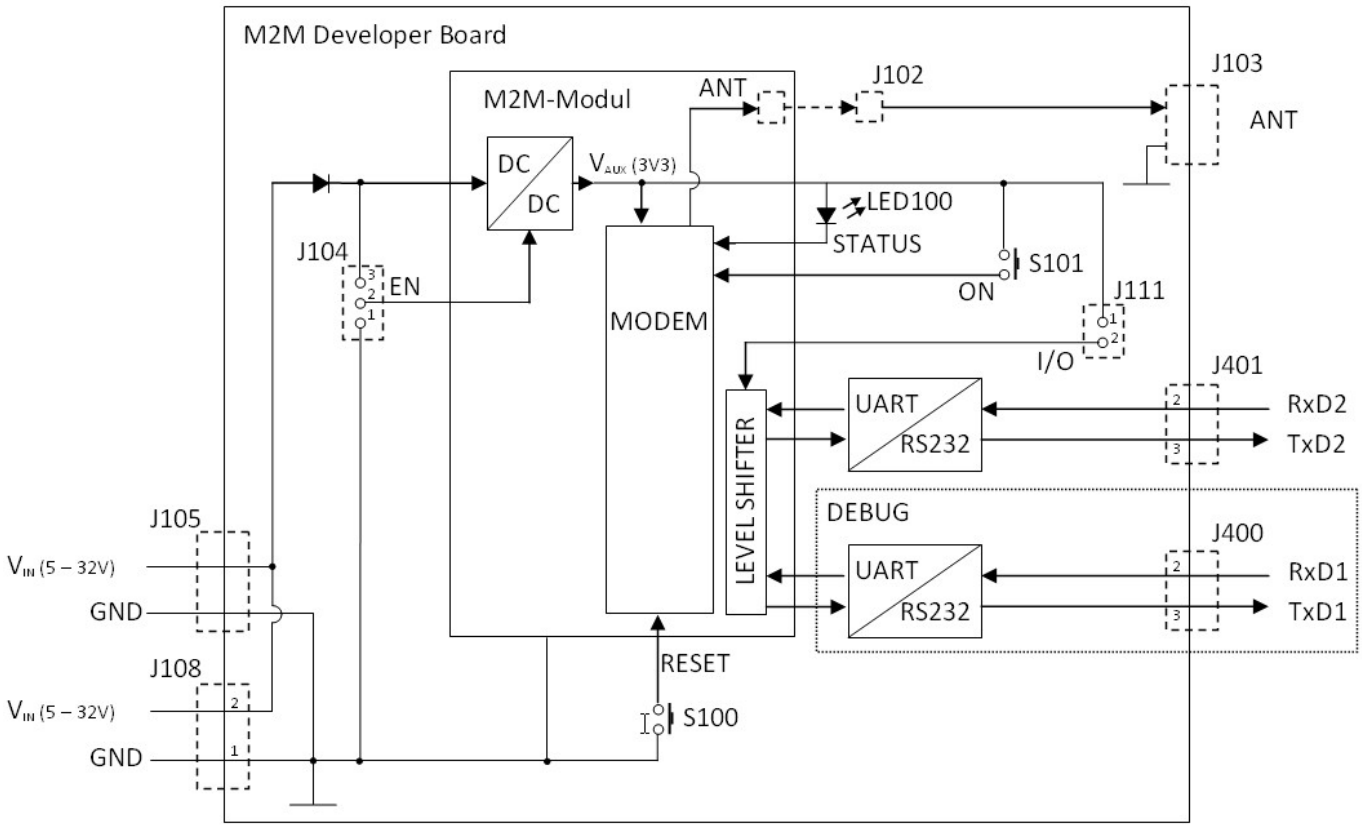


Übersicht rapidM2M Developer Paket M11

1	COM1 (J400)	**9**	Taste zum Reseten des Wireless GSM Controllers am rapidM2M Modul M11 (S100)
2	Jumper zur Versorgung der Level Shifter (J111)	**10**	SMA Antennenanschluss (J103) (siehe "Anschluss der GSM-Antenne" auf Seite 40)
3	COM2 (J401)	**11**	U.FL Antennenanschluss zum Verbinden des M11 Moduls mit dem rapidM2M Developer Board (J102)
4	Stiftleiste auf der die Pins mit den ungeraden Nummern des M11 Moduls aufgelegt sind. (J106 und J109)	**12**	Aufnahme des M11 Moduls (siehe "Einsetzen des M11 Moduls" auf Seite 38)
5	Status-LED (LED100)	**13**	U.FL Antennenanschluss des M11 Moduls
6	Versorgung (DC-Hohlstecker (J105) oder Klemmleiste (J108))	**14**	rapidM2M Modul M11 (siehe "rapidM2M Modul M11 " auf Seite 43)
7	Jumper zum Aktivieren der Energieversorgung des M11 Moduls (J104)	**15**	Stiftleiste auf der die Pins mit den geraden Nummern des M11 Moduls aufgelegt sind (J107 und J110)
8	Taster zum Einschalten des Wireless GSM Controllers am rapidM2M Modul M11 (S101)		

5.1.1 Blockschaltbild

__Hinweis:__ Ein detailliertes Blockschaltbild des ins rapidM2M Developer Board eingesetzten M11 Moduls finden Sie im Kapitel "Blockschaltbild" auf Seite 45.



Blockschaltbild des Developer Boards

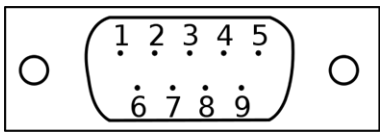
5.1.2 RS232-Schnittstelle COM1 (J400)

***Wichtiger Hinweis:** Die UART1 des M11 Moduls dient nur der Ausgabe von Debug-Informationen. Das Hardware-Handshake ist dabei nicht aktiv. Die UART1 kann nicht für die Datenaufzeichnung verwendet werden.*

Am Anschluss COM1 des Developer Boards ist die UART1 des M11 Moduls zugängig. Die Umwandlung auf den RS232-Pegel erfolgt dabei durch einen Pegelwandler, der sich auf dem rapidM2M Developer Board befindet (siehe "Blockschaltbild" auf Seite 23).

Baudrate	115200
Stopbits	2
Parität	N
Datenbits	8
Modi	ASCII

Die Richtung der Signale entspricht jener eines DTE (z.B. PC).



9pol. Sub-D(m)

Belegung des Sub-D Steckers

Pin	Signal
1	NC
2	RXD
3	TXD
4	NC
5	GND
6	NC
7	RTS [1)]
8	CTS [1)]
9	NC

[1)] Das Hardware-Handeshake wird von der aktuellen Firmware des M11 Moduls nicht unterstützt.

5.1.3 Jumper zur Versorgung der Level Shifter (J111)

Um die Signalpegel des M11 Moduls an die umgebende Hardware anzupassen, ist das Modul mit Level Shifter für alle I/O-Signale ausgestattet. Für die Anpassung muss von der umgebenden Hardware eine zusätzliche Versorgungsspannung für die Level Shifter im Bereich von 2,8...5,5V bereitgestellt werden. Über den Jumper J111 des Developer Boards wird V_{AUX} des M11 Moduls als Versorgung für die Level Shifter zurück ins Modul geführt (siehe "Blockschaltbild" auf Seite 23).

> ***Hinweis:*** *Der Jumper J111 sollte immer geschlossen sein.*

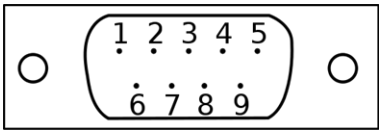
5.1.4 RS232-Schnittstelle COM2 (J401)

> ***Wichtiger Hinweis:*** *Das im Lieferumfang des Developer Pakets enthaltene rapidM2M Modul M11 unterstützt kein Hardware-Handshake.*

Am Anschluss COM2 des Developer Boards ist die UART2 des M11 Moduls zugängig. Die Umwandlung auf den RS232-Pegel erfolgt dabei durch einen Pegelwandler, der sich auf dem rapidM2M Developer Board befindet (siehe "Blockschaltbild" auf Seite 23).

Baudrate	300-115200
Stopbits	1,2
Parität	N, E, O
Datenbits	7, 8
Modi	ASCII

Die Richtung der Signale entspricht jener eines DTE (z.B. PC).



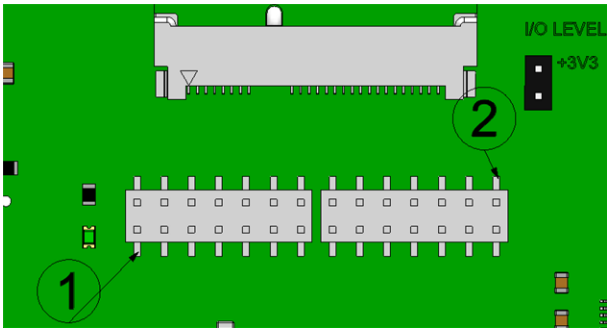
9pol. Sub-D(m)

Belegung des Sub-D Steckers

Pin	Signal
1	NC
2	RXD
3	TXD
4	NC
5	GND
6	NC
7	RTS [1)]
8	CTS [1)]
9	NC

[1)] Das Hardware-Handeshake wird von der aktuellen Firmware des M11 Moduls nicht unterstützt.

5.1.5 Stiftleiste auf der die Pins mit den ungeraden Nummern des M11 Moduls aufgelegt sind (J106 und J109)



Stiftleiste auf der die Pins mit den ungeraden Nummern des M11 Moduls aufgelegt sind

1 Pin1	**2** Pin28

Pin	Typ	Bezeichnung	Beschreibung
1	I	Modul_ON	Pin zum Einschalten des Wireless GSM Controllers am rapidM2M Modul M11
2	I	EN_Power	Pin zum Aktivieren des DC/DC-Wandlers der aus der an V_{IN} angelegten Spannung die interne Versorgungsspannung V_{AUX} erzeugt
3	O	LED_Status	Open Drain Ausgang zum Anschluss eines Status-LED
4	I	Reset	Pin zum Reseten des Wireless GSM Controllers am rapidM2M Modul M11
5-6		GND	Masse
7-8			reserviert für Erweiterungen[1)]
9	I	V_{IN}	Versorgungsspannung (4,5...32VDC) für das rapidM2M Modul M11
10,15	I/O	V_{AUX}	3,3V Spannungsausgang, wenn das rapidM2M Modul M11 über V_{IN} versorgt wird Versorgungsspannungseingang (3,2...4,5V), wenn an V_{IN} keine Spannung angelegt wird
11-14			reserviert für Erweiterungen[1)]
16	I/O	I/O Level	2,8...5,5V Versorgung der Level Shifter zur Anpassung der Signalpegel des M11 Moduls an die umgebende Hardware
17-18			reserviert für Erweiterungen[1)]
19-20		GND	Masse
21	O	RTS2	RTS Leitung der COM2[2)]
22	I	CTS2	CTS Leitung der COM2[2)]
23	O	TxD2	TxD Leitung der COM2
24	I	RxD2	RxD Leitung der COM2
25	O	RTS1	RTS Leitung der COM1[2)]
26	I	CTS1	CTS Leitung der COM1[2)]
27	O	TxD1	TxD Leitung der COM1
28	I	RxD1	RxD Leitung der COM1

[1)] Es dürfen keine Signale an diese Pins angelegt werden.

[2)] Das Hardware-Handeshake wird von der aktuellen Firmware des M11 Moduls nicht unterstützt.

5.1.6 Status-LED (LED100)

Die Status-LED dient der Signalisierung des aktuellen Betriebszustandes.

Betriebszustände

Status-LED	Beschreibung
aus	Das rapidM2M Modul M11 ist nicht an die Stromversorgung angeschlossen.
leuchtet	Das rapidM2M Modul M11 ist an die Stromversorgung angeschlossen, aber noch nicht ins Netzwerk eingebucht.
blinkt t_{on}/t_{off} : 0,2/2sec.	Das rapidM2M Modul M11 ist ins Netzwerk eingebucht.
schnelles Blinken t_{on}/t_{off} : 0,2/0,6sec.	Das rapidM2M Modul M11 ist mit dem myDataweb-Server Verbunden. (z.B. Online-Mode oder ALOHA-Übertragungsmodus)

5.1.7 Versorgung (J105 bzw. J108)

Die Versorgung des Developer Boards kann entweder über den DC-Hohlstecker (J105) oder die Klemmleiste (J108) erfolgen. Neben dem im Lieferumfang enthaltenen Steckernetzgerät können auch Netzgeräte verwendet werden, die die in den folgenden Kapiteln beschriebenen Bedingungen erfüllen.

5.1.7.1 Klemmleiste (J108)

U	5...32 VDC
I_{min}	t.b.d. mA@5V t.b.d. mA@12V t.b.d. mA@24V

Belegung der Klemmleiste (J108)

1	GND	Masse
2	$+V_{dd}$	Versorgungsspannung:5...32 VDC

5.1.7.2 DC-Hohlstecker (J105)

U	5...32 VDC
I_{min}	t.b.d. mA@5V t.b.d. mA@12V t.b.d. mA@24V
Außen-/Innen-Ø	5,5/2,1mm
Polarität	+Innen

5.1.8 Jumper zum Aktivieren der Energieversorgung des M11 Moduls (J104)

Ist der Jumper J104 geöffnet oder sind Pin1 und Pin2 durch eine Steckbrücke verbunden, ist der DC/DC-Wandler des M11 Moduls inaktiv und somit die interne Versorgung des Moduls abgeschaltet.

Werden Pin2 und Pin3 verbunden, wird der DC/DC-Wandler aktiviert. Durch den Soft-Start des Wandlers dauert es 50ms bis die interne Versorgungsspannung des M11 Moduls den Maximalwert erreicht.

5.1.9 Taster zum Einschalten des Wireless GSM Controllers am rapidM2M Modul M11 (S101)

50ms (Soft-Start) nachdem die Energieversorgung des M11 Moduls mittels Jumper J104 (siehe "Jumper zum Aktivieren der Energieversorgung des M11 Moduls (J104) " auf Seite 28 aktiviert wurde, kann der Wireless GSM Controller des M11 Moduls durch einen 1,5sec. langen Druck der Taste S101 eingeschaltet werden. Während der Startphase sollte die Reset-Taste S100 nicht betätigt werden. Auf der Debug-Schnittstelle COM1 (J401) wird während des Boot-Vorgangs folgende Meldung ausgegeben:

```
############################################################
# myDatalogAIR232       Build Aug 28 2012 13:06:25 03v000 #
############################################################
# OS        # OK    # 7.46.0
# S/N       # OK    # 010146AF25120BCD
# GSM       # OK    # -69 dBm
# IMSI      # OK    # 232034011697750
# GPRS      # OK    # IP = xxx.xxx.xxx.xxx
# MNT SERVER # OK    # connected
# MDN SERVER # OK    # connected
```

> ***Hinweis:*** *"myDatalogAIR232 " ist die Bezeichnung der Firmware, die am rapidM2M Modul M11 installiert ist. Später, beim Anlegen der Messstelle am myDataweb-Server (siehe " Benutzerhandbuch für myDataweb-Server" 206.886), müssen Sie eine Messstelle dieses Typs erstellen.*

Ein Abschalten des Wireless GSM Controllers ist bei der aktuellen Firmware des M11 Moduls nicht vorgesehen.

Ein detaillierte Beschreibung des Timeings finde Sie unter "ENCPU-Signal" auf Seite 48.

5.1.10 Taste zum Reseten des Wireless GSM Controllers am rapidM2M Modul M11 (S100)

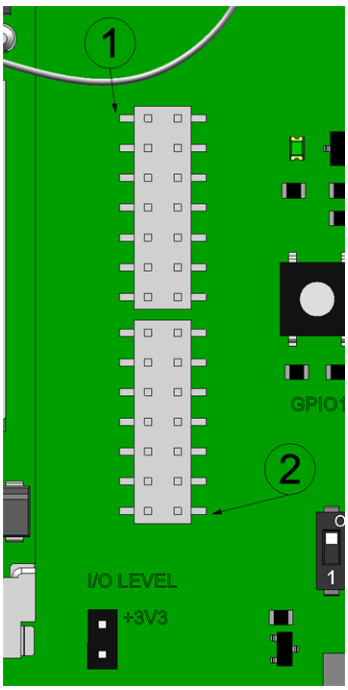
Diese Taste sollte nur als Notfall-Reset betrachtet werden, da eine Reset-Prozedur vom Wireless GSM Controller während des Boot-Vorgangs selbstständig durchgeführt wird. Sollte ein Reset dennoch erforderlich werden muss die Taste S100 kurz (200µs) gedrückt werden. Danach muss der Wireless GSM Controller des M11 Moduls mittels Taste S101 (siehe "Taster zum Einschalten des Wireless GSM Controllers am rapidM2M Modul M11 (S101)" auf Seite 28) wieder eingeschaltet werden.

Ein detaillierte Beschreibung des Timeings finde Sie unter "Reset-Signal" auf Seite 49.

5.1.11 Aufnahme des M11 Moduls

Der Hersteller der beim rapidM2M Developer Board verwendeten Aufnahme für das rapidM2M Modul M11 ist "JAE Connectors". Die Herstellerbezeichnung dieses Steckers ist "MM60-52B1-G1-R850".

5.1.12 Stiftleiste auf der die Pins mit den geraden Nummern des M11 Moduls aufgelegt sind (J107 und J110)



Stiftleiste auf der die Pins mit den geraden Nummern des M11 Moduls aufgelegt sind

1 Pin1	**2** Pin28

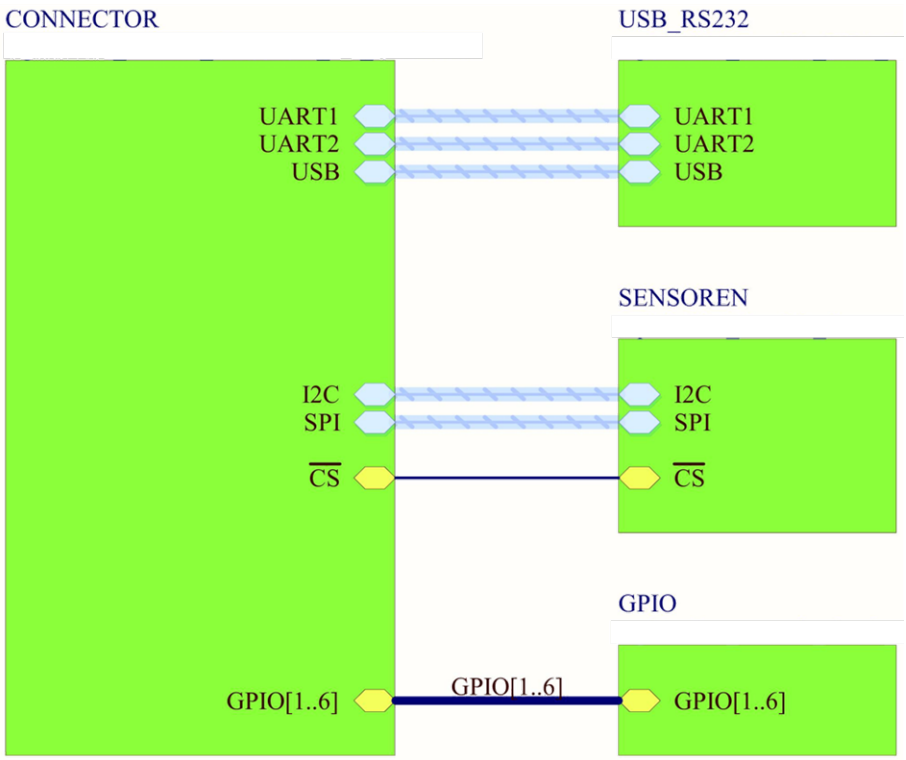
Pin	Typ	Bezeichnung	Beschreibung
1,27-28	I	V_{IN}	Versorgungsspannung (4,5...32VDC) für das rapidM2M Modul M11
2,18	I/O	V_{AUX}	3,3V Spannungsausgang, wenn das rapidM2M Modul M11 über V_{IN} versorgt wird Versorgungsspannungseingang (3,2...4,5V), wenn an V_{IN} keine Spannung angelegt wird
3-8			reserviert für Erweiterungen[1)]
9-10		GND	Masse
11-12			reserviert für Erweiterungen[1)]
13		GND	Masse
14		NC	
15-17			reserviert für Erweiterungen[1)]
19-20			reserviert für Erweiterungen[1)]
21-22		GND	Masse
23-26			reserviert für Erweiterungen[1)]

[1)] Es dürfen keine Signale an diese Pins angelegt werden.

5.2 Schaltplan

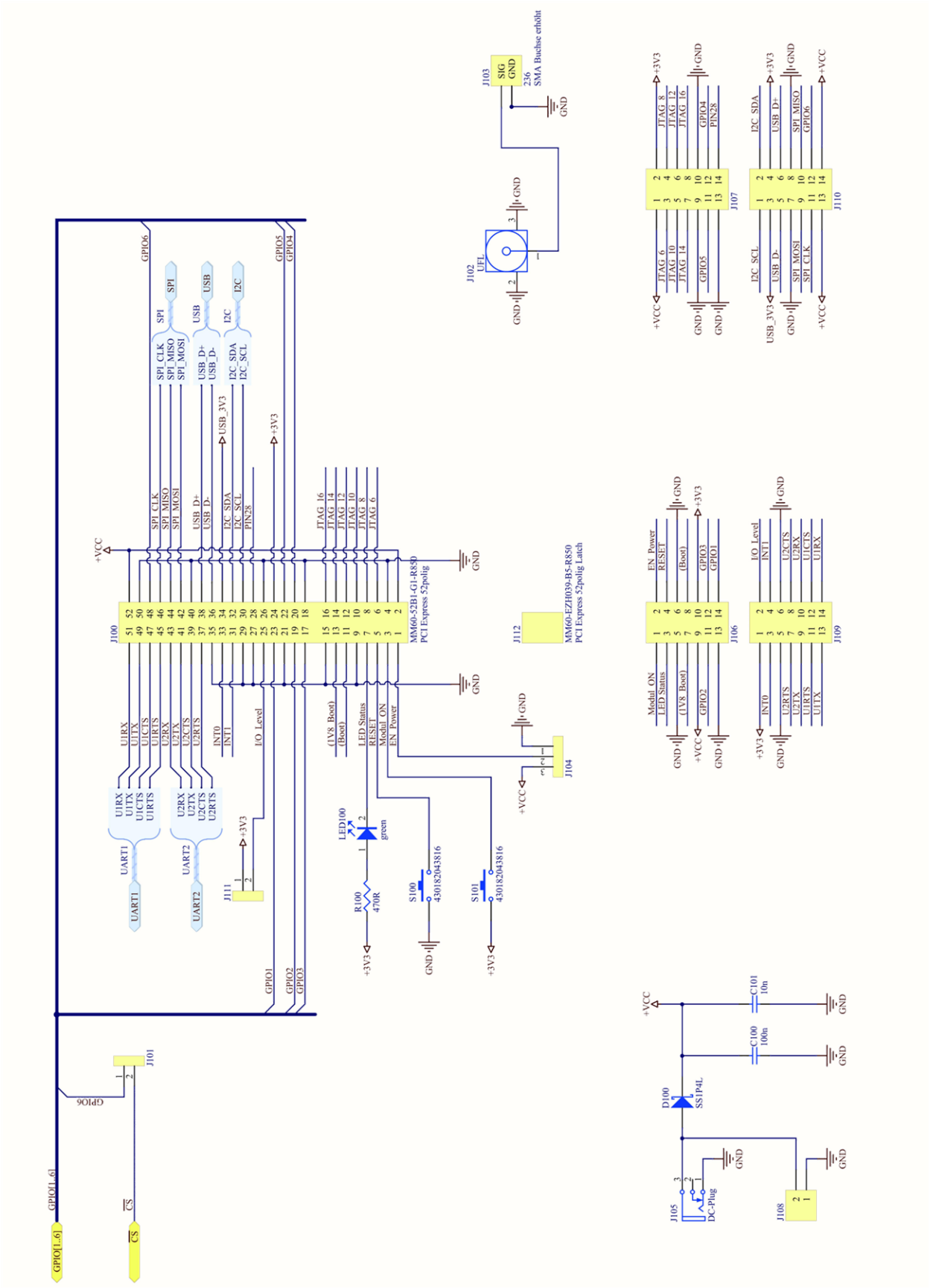
Hinweis: *Das folgende Kapitel enthält den kompletten Schaltplan des Developer Boards. Einige der darin enthaltenen Baugruppen werden vom rapidM2M Modul M11 nicht unterstützt und sind für Erweiterungen vorgesehen.*

5.2.1 Übersicht

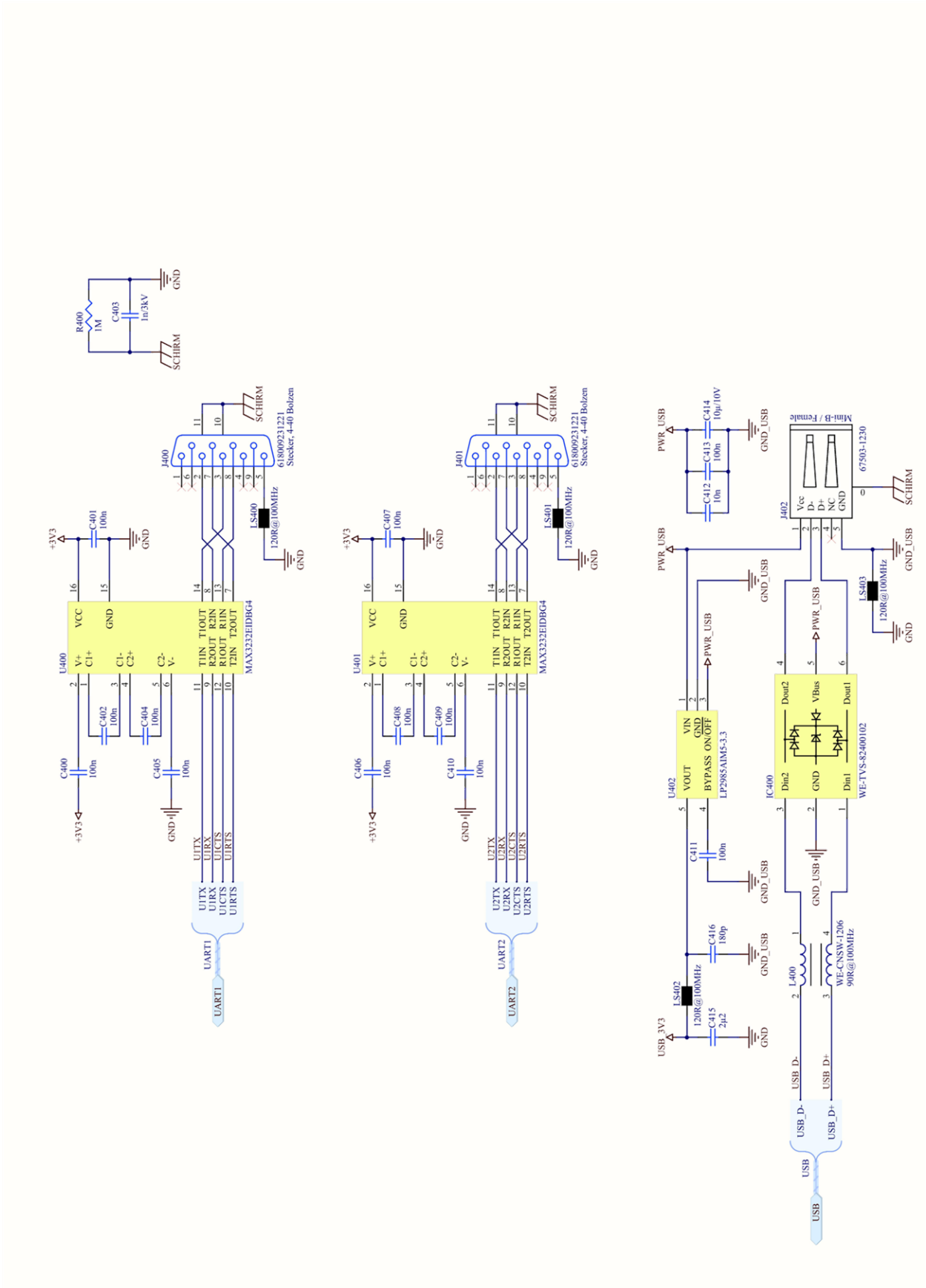


Schaltplanübersicht des Developer Boards

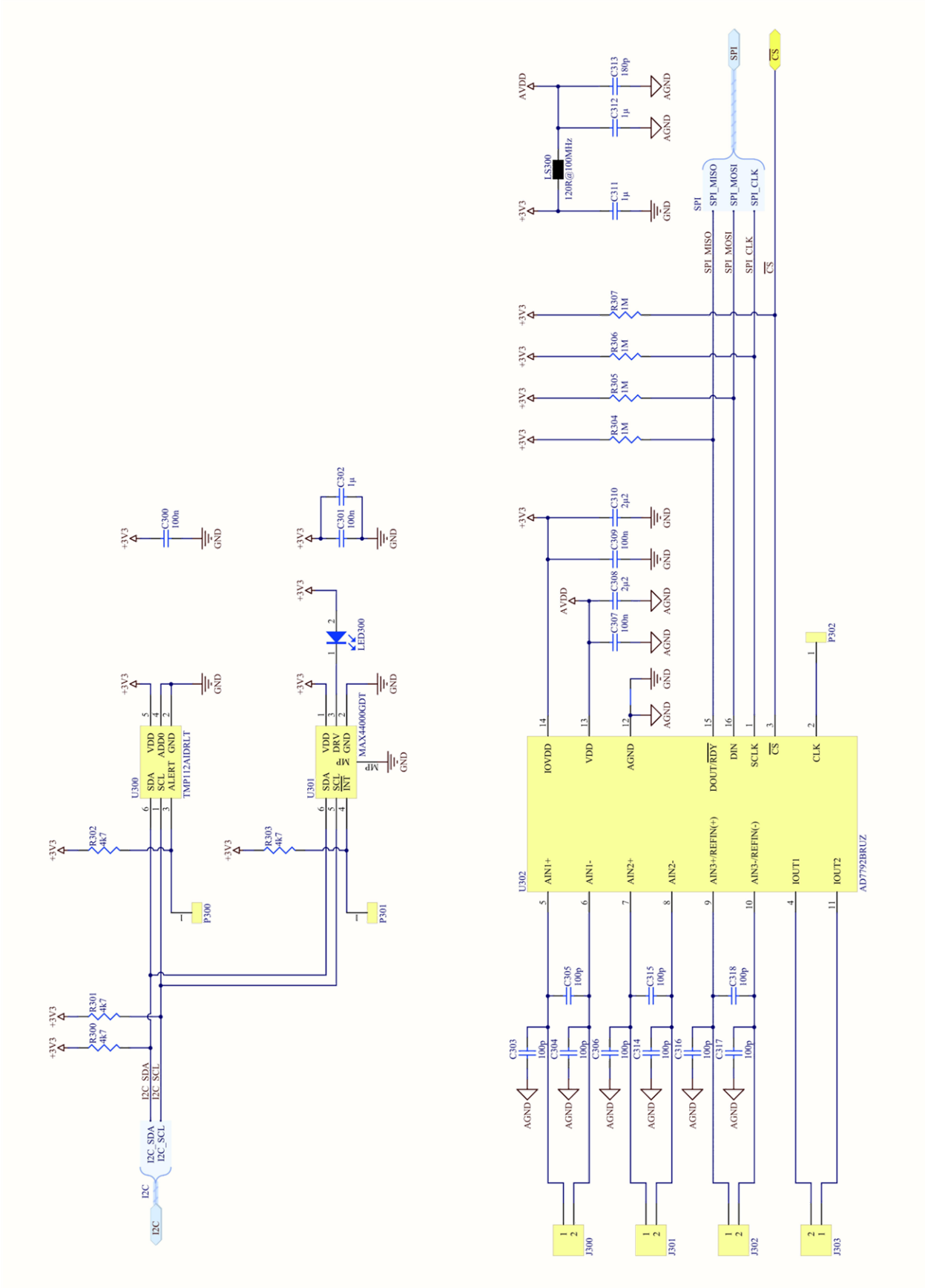
5.2.2 Connector



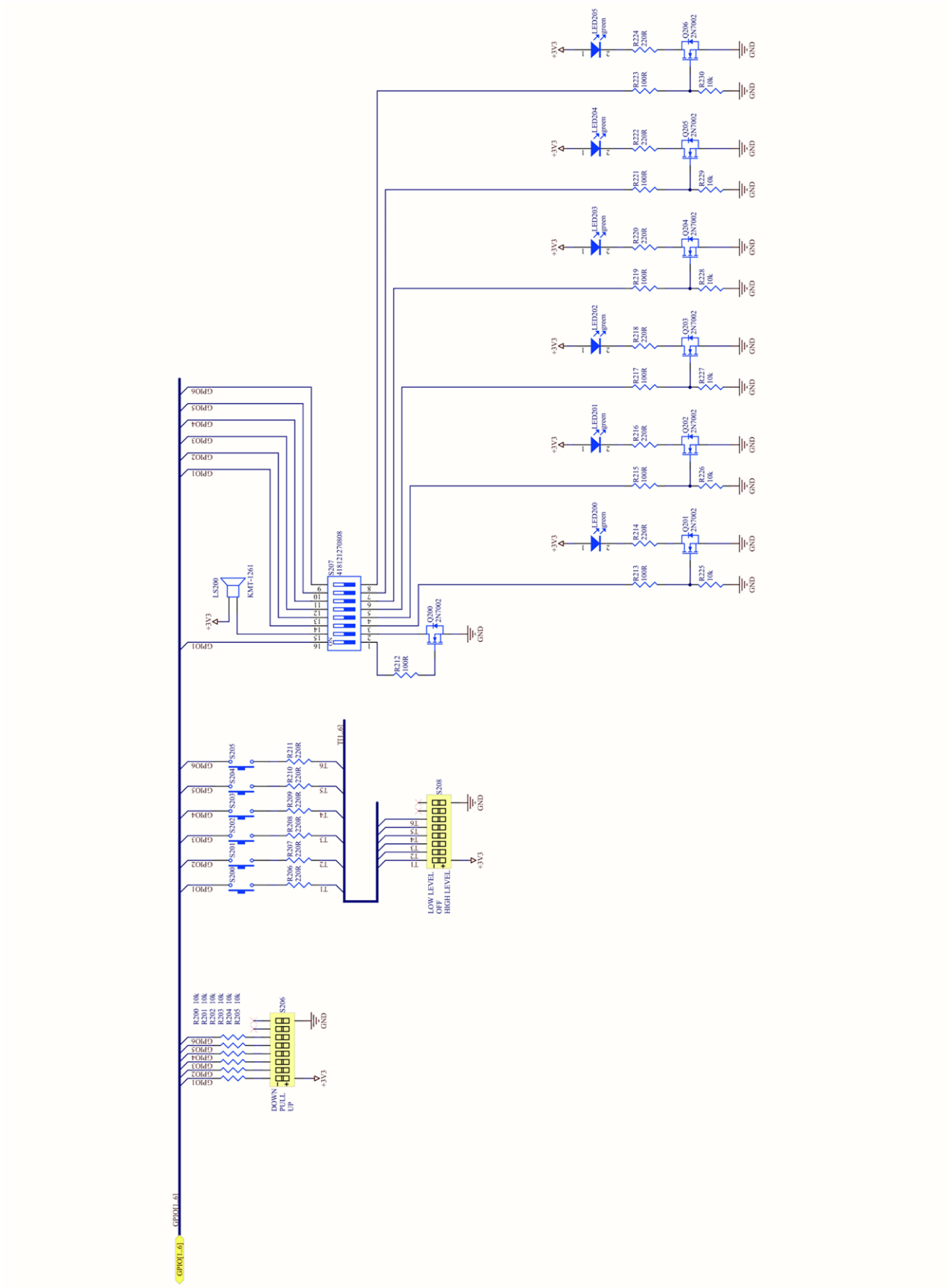
5.2.3 USB RS232



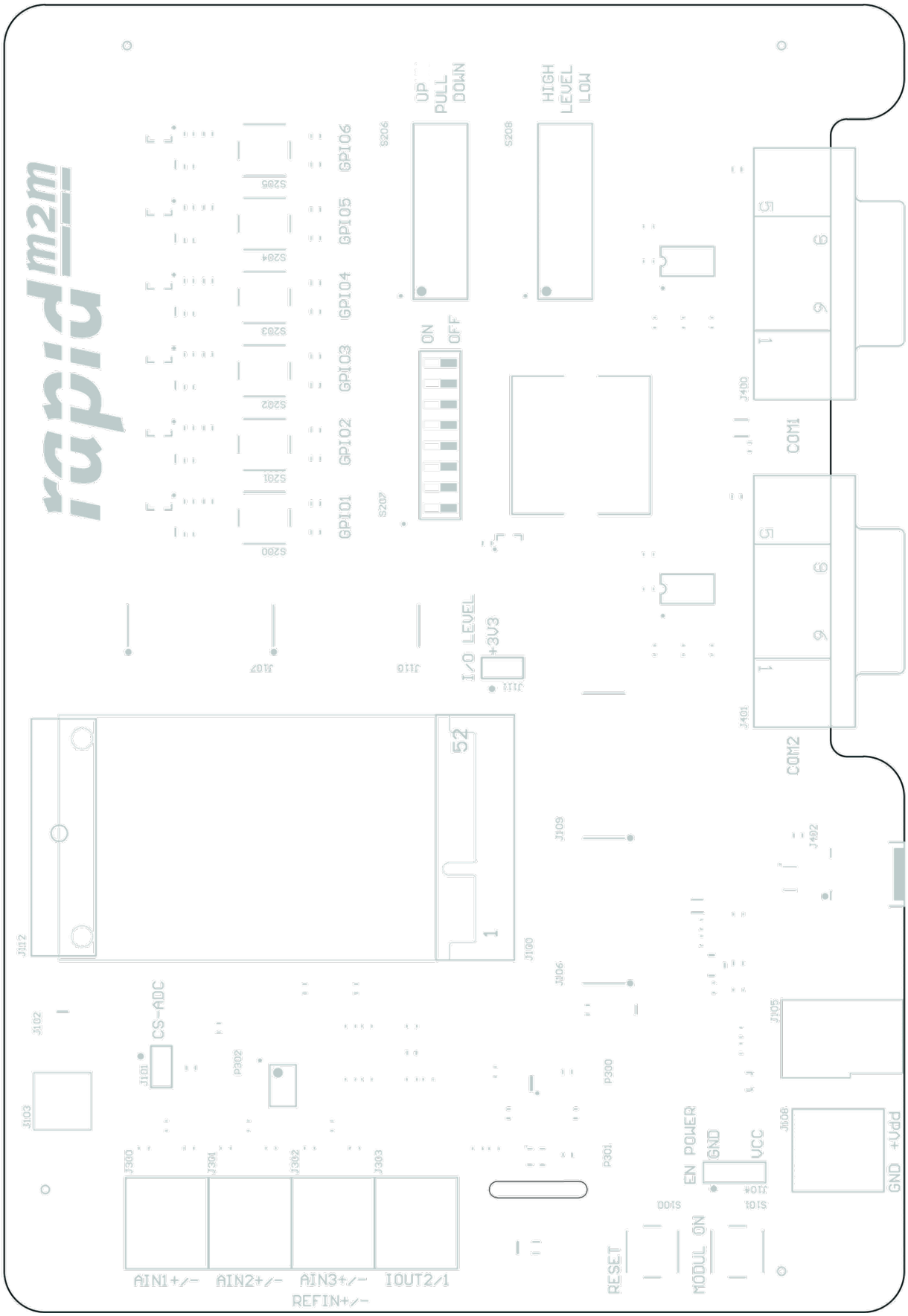
5.2.4 Sensoren



5.2.5 GPIO


LED205
LED204
LED203
LED202
LED201
LED200
Q206
Q205
Q204
Q203
Q202
Q201
Q200
2N7002
R224
R223
R222
R221
R220
R219
R218
R217
R216
R215
R214
R213
R212
R210
R209
R208
R207
R206
R205
R204
R203
R202
R201
R200
R230
R229
R228
R227
R226
R225
220R
100R
10k
+3V3
GND
GPIO6
GPIO5
GPIO4
GPIO3
GPIO2
GPIO1
LS200
KMT-1261
S207
41812127808
S205
S204
S203
S202
S201
S200
S208
S206
TH_61
GPIOH_61
LOW LEVEL
OFF
HIGH LEVEL
DOWN
PULL
UP

5.3 Layout


rapid m2m
GPIO1 GPIO2 GPIO3 GPIO4 GPIO5 GPIO6
S200 S201 S202 S203 S204 S205
S206
UP
PULL
DOWN
S208
HIGH
LEVEL
LOW
S207
ON
OFF
I/O LEVEL
+3V3
J111
J107
J110
COM1
COM2
J400
J401
J402
J112
J100
J109
J106
J105
J108
J102
J103
J101
CS-ADC
P302
P300
P301
J300
J304
J302
J303
AIN1+/-
AIN2+/-
AIN3+/-
IOUT2/1
REFIN+/-
EN POWER
GND
VCC
J104
S101
S100
RESET
MODUL ON
GND +Vdd

5.4 Elektrische Installation

> ***Wichtiger Hinweis:*** *Um Schäden am Gerät zu vermeiden, sollte nur qualifiziertes Personal die in diesem Kapitel der Bedienungsanleitung beschriebene Installation durchführen.*

5.4.1 Hinweise zur Vermeidung elektrostatischer Entladungen (ESE)

> ***Wichtiger Hinweis:*** *Um Gefahren und ESE-Risiken zu minimieren, sollten Wartungsprozeduren, für die keine Stromversorgung des Geräts erforderlich ist, nur nach Trennung vom Stromnetz ausgeführt werden.*

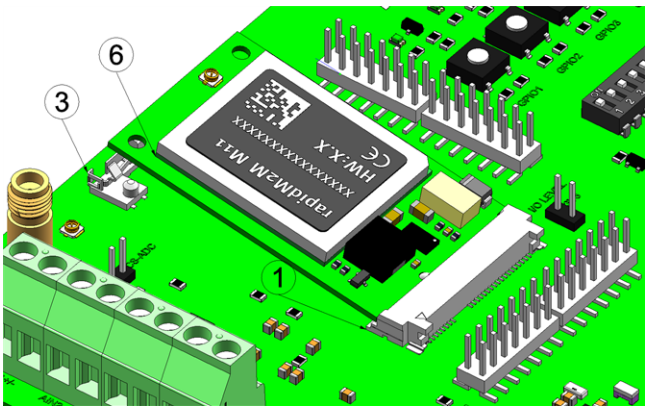
Die empfindlichen elektronischen Komponenten im Geräteinneren können durch statische Elektrizität beschädigt werden, was zur Beeinträchtigung der Geräteleistung bis hin zum Ausfall des Geräts führen kann. Der Hersteller empfiehlt die folgenden Schritte zur Vermeidung von Beschädigungen des Geräts durch elektrostatische Entladungen:

- Leiten Sie eventuell auf Ihrem Körper vorhandene statische Elektrizität ab, bevor Sie elektronische Komponenten des Geräts (wie z.B. Leiterplatten und die Komponenten darauf) berühren. Hierzu können Sie eine geerdete metallische Oberfläche berühren, wie etwa den Gehäuserahmen eines Geräts oder ein Metallrohr.
- Vermeiden Sie unnötige Bewegungen, um den Aufbau statischer Ladungen zu vermindern.
- Transportieren Sie statisch-empfindliche Komponenten in antistatischen Behältnissen oder Verpackungen.
- Tragen Sie ein Antistatikarmband, das über ein Kabel geerdet ist, um Ihren Körper zu entladen und von statischer Elektrizität freizuhalten.
- Fassen Sie Komponenten, die gegen Aufladungen empfindlich sind, nur in einem Antistatik-Arbeitsbereich an. Verwenden Sie, falls möglich, antistatische Fußbodenbeläge und Arbeitsunterlagen.

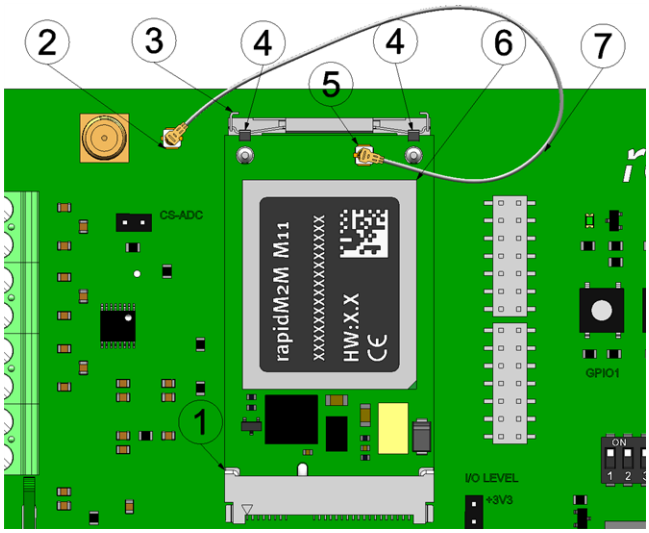
5.4.2 Einsetzen des M11 Moduls

Wichtiger Hinweis:

- *Das Einsetzen des M11 Moduls muss im stromlosen Zustand erfolgen!*
- *Unsachgemäße Handhabung kann zu Verletzungen und/oder Beschädigungen der Instrumente führen!*



Einsetzen des M11 Moduls Schritt 1



Einsetzen des M11 Moduls Schritt 2

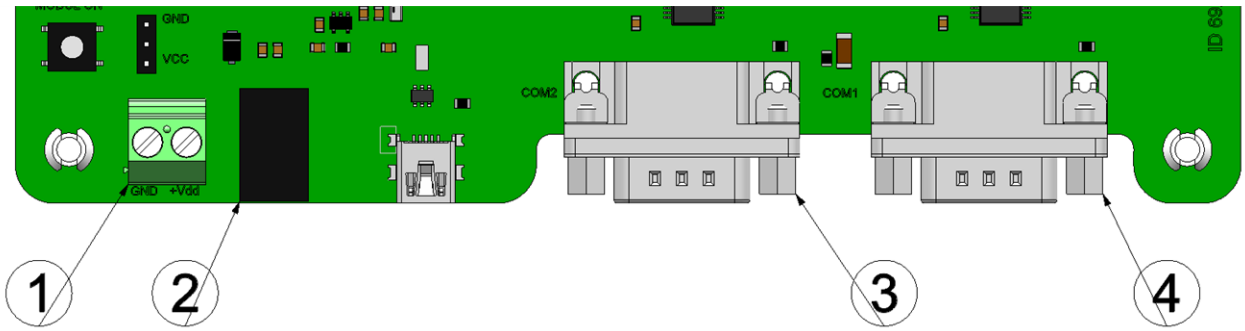
1	52pin Steckverbinder (mechanisch kompatibel zu "PCI Express Mini Card")	5	U.FL Antennenanschluss des rapidM2M Modul M11
2	U.FL Antennenanschluss des Developer Boards (J102)	6	rapidM2M Modul M11
3	Arretierung	7	U.FL Antennenkabel
4	Haken der Arretierung		

1. Setzen Sie das rapidM2M Modul M11 (6) entsprechend der Abbildung "Einsetzen des M11 Moduls Schritt 1 " auf Seite 38 in den 52pin Steckverbinder (1) des Developer Boards ein. Das Modul muss dabei so eingesetzt werden, dass das Typenschild nach dem einrasten sichtbar bleibt.
2. Üben Sie dann auf die am weitesten vom 52pin Steckverbinder (1) entfernte Seite des M11 Moduls so lange einen leichten Druck aus, bis das Modul in die Arretierung (3) einrastet. Zum Lösen des Moduls drücken Sie die beiden Haken der Arretierung (4) nach hinten.
3. Verbinden Sie anschließend den U.FL Antennenanschluss des M11 Moduls (5) mittels des im Lieferumfang enthaltenen U.FL Antennenkabels mit dem U.FL Antennenanschluss des Developer Boards (2).

5.4.3 Anschluss des Sensors und der Versorgung

Wichtiger Hinweis:

- *Alle Verkabelungsarbeiten sollten im stromlosen Zustand erfolgen!*
- *Befolgen Sie bestehende gesetzliche bzw. betriebliche Richtlinien!*
- *Unsachgemäße Handhabung kann zu Verletzungen und/oder Beschädigungen an den Instrumenten führen!*



Anschluss des Sensors und der Versorgung

1	Versorgung: Klemmleiste (J108)	**3**	Sensoranschluss COM2: 9pol. D-Sub male (J400)
2	Versorgung: DC-Hohlstecker (J105) 5,5/2,1mm (Außen-/Innendurchmesser)	**4**	Debug-Schnittstelle COM1: 9pol. D-Sub male (J401)

Belegung der Klemmleiste (J108)

1	GND	Masse
2	$+V_{dd}$	Versorgungsspannung:5...32 VDC

Belegung des DC-Hohlsteckers (J105) 5,5/2,1mm (Außen-/Innendurchmesser)

Außenkontakt	GND	Masse
Innenkontakt	$+V_{dd}$	Versorgungsspannung:5...32 VDC

1. Verbinden Sie zuerst den auszulesenden digitalen Sensor mit dem Sensoranschluss (4) des Developer Boards. Wenn möglich, sollte die Versorgung des digitalen Sensors abgeschaltet werden bevor Sie die RS232 Verbindung zw. dem rapidM2M Developer Board und dem digitalen Sensor herstellen.

   Hinweis: *Unter Umständen ist dazu ein Adapter erforderlich. Im Kapitel Ersatzteile und Zubehör (siehe "Ersatzteile und Zubehör" auf Seite 141) finden Sie eine Auswahl an verfügbaren Adaptern.*

   Wichtiger Hinweis: *Das im Lieferumfang des Developer Pakets enthaltene rapidM2M Modul M11 unterstützt kein Hardware-Handshake.*

2. Schließen Sie die Antenne an (siehe "Anschluss der GSM-Antenne" auf Seite 40).

3. Versorgen können Sie das rapidM2M Developer Board entweder über den DC-Hohlstecker (2) oder die Klemmleiste (1). Verwenden Sie dazu entweder das im Lieferumfang enthaltene Steckernetzgerät oder jedes beliebige Netzgerät, das eine Spannung im Bereich von 5...32 VDC liefert. Soll die Versorgung des Developer Boards über die Klemmleiste (1) erfolgen, sollten Sie die Kabel im stromlosen Zustand mit der Klemmleiste (1) verbinden.

4. Vergewissern Sie sich, dass der Jumper zur Versorgung der Level Shifter (J111) geschlossen ist und beim Jumper zum Aktivieren der Energieversorgung (J104) Pin2 und Pin3 verbunden sind (siehe "Übersicht" auf Seite 21).

5. Schalten Sie jetzt die Spannungsversorgung des digitalen Sensors und des Developer Boards ein.

6. Drücken Sie als nächstes die Taste zum Einschalten des Wireless GSM Controllers S101 (siehe "Taster zum Einschalten des Wireless GSM Controllers am rapidM2M Modul M11 (S101)" auf Seite 28). Daraufhin sollte das Status-LED zu leuchten beginnen (siehe "Status-LED (LED100)" auf Seite 27), um zu signalisieren, dass sich das rapidM2M Developer Paket M11 ins Netz einbucht und die Verbindung zum myDataweb aufnimmt.

   Hinweis: Über die Debug-Schnittstelle COM1 (J401) erhalten Sie zusätzliche Statusinformationen.

Der folgende Schritt ist nicht zwingend erforderlich.

7. Überprüfen Sie, ob die Verbindung zum myDataweb korrekt funktioniert hat (siehe "Kommunikation mit dem Gerät testen" auf Seite 53).

5.4.4 Anschluss der GSM-Antenne

Wichtiger Hinweis: Um eine korrekte Funktion zu gewährleisten, benutzen Sie nur Antennen, die vom Hersteller geliefert werden.

Die im Lieferumfang enthaltene Standantenne Quad Band SMA (300085) wird direkt mit dem Antennenstecker (siehe "Übersicht" auf Seite 21) des Developer Boards verbunden.

Sollte die Länge des 2,5m Anschlusskabels der Standantenne Quad Band SMA (300085) nicht ausreichen, können Sie eine 2,5m Verlängerung für Antenne 900 SMA (206.807) verwenden.

1. Stellen Sie sicher, dass das rapidM2M Developer Board spannungslos ist.

2. Verbinden Sie, falls Sie eine Antennenverlängerung benötigen, diese zuerst mit der Antenne.

3. Verbinden Sie die Antennenverlägerung bzw. die Antenne direkt mit dem Antennenanschluss des Developer Boards (siehe "Übersicht" auf Seite 21).

   Wichtiger Hinweis: Vermeiden Sie zu starke Krafteinwirkung beim Festziehen der Antenne. Benutzen Sie kein Werkzeug zum Festziehen der Antennen bzw. der Antennenverlängerung sondern ziehen Sie die Antenne mit der Hand fest.

4. Stellen Sie die Spannungsversorgung des Developer Boards wieder her.

5. Drücken Sie als nächstes die Taste zum Einschalten des Wireless GSM Controllers S101 (siehe "Taster zum Einschalten des Wireless GSM Controllers am rapidM2M Modul M11 (S101)" auf Seite 28).

Der folgende Schritt ist nicht zwingend erforderlich.

6. Überprüfen Sie, ob die Verbindung zum myDataweb korrekt funktioniert hat (siehe "Kommunikation mit dem Gerät testen" auf Seite 53).

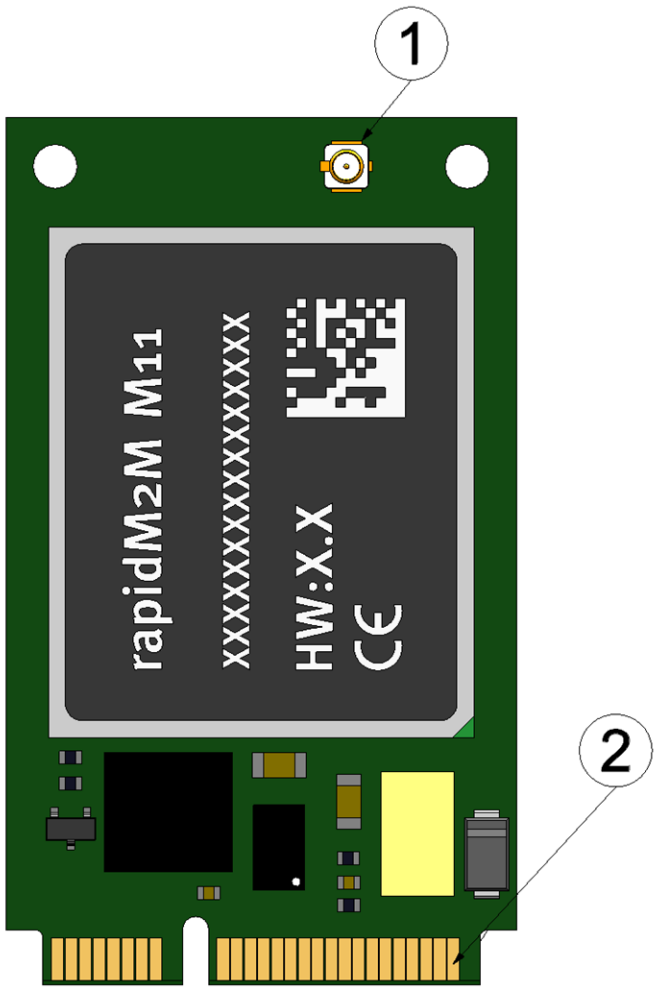
Kapitel 6 rapidM2M Modul M11

6.1 Technische Daten

Spannungsversorgung	Versorgung über V_{IN}: 4,5...32VDC , max. t.b.d. mA Versorgung über V_{AUX}: 3,2...4,5V , max. t.b.d. mA Zusätzliche Informationen finden Sie unter "Technische Details zur Energieversorgung" auf Seite 48.
Ausgangsspannung (Nur bei Versorgung über V_{IN})	3,3V , max 300mA
Eingangsbereich der Pegelwandler für I/Os	2,8...5,5V Zusätzliche Informationen finden Sie unter "Technische Details zu den Level Shiftern" auf Seite 50.
Gehäuse	Gewicht: 11g Interface: 52pin Steckerleiste Abmessungen (BHT): 30 x 5,5 x 51mm(ohne Antenne)
Betriebstemperatur	-20...+60°C, 15...90%rH nicht kondensierend
Lagertemperatur	-40...+85°C
Anzeige	Open Drain Ausgang zum Anschluss eines Status-LEDs Zusätzliche Informationen finden Sie unter "Status-LED (LED100)" auf Seite 27.
Antennenanschluss	U.FL
Schnittstellen	1x UART für den Anschluss eines digitalen Sensors • Baudrate: 300-115200 • Stopbits: 1,2 • Parität:N, E, O • Datenbits: 7, 8 Modi: • ASCII 1x UART für die Ausgabe von Debug-Informationen Zusätzliche Informationen finden Sie unter "Technische Details zur UART1" auf Seite 50 und "Technische Details zur UART2" auf Seite 51.

Datenspeicher	Interner Flash-Speicher für bis zu 2620 Messzyklen
Datentyp	f32 (32 Bit floating point)
Datenübertragung	Mittels GSM/GPRS Quad-Band Modem an den jeweiligen myDataweb-Server
SIM	Das rapidM2M Modul M11 ist mit einem integrierten SIM-Chip versehen.
Monatliches Datenvolumen	t.b.d. bei 2min Messintervall und 120min Übertragungsintervall

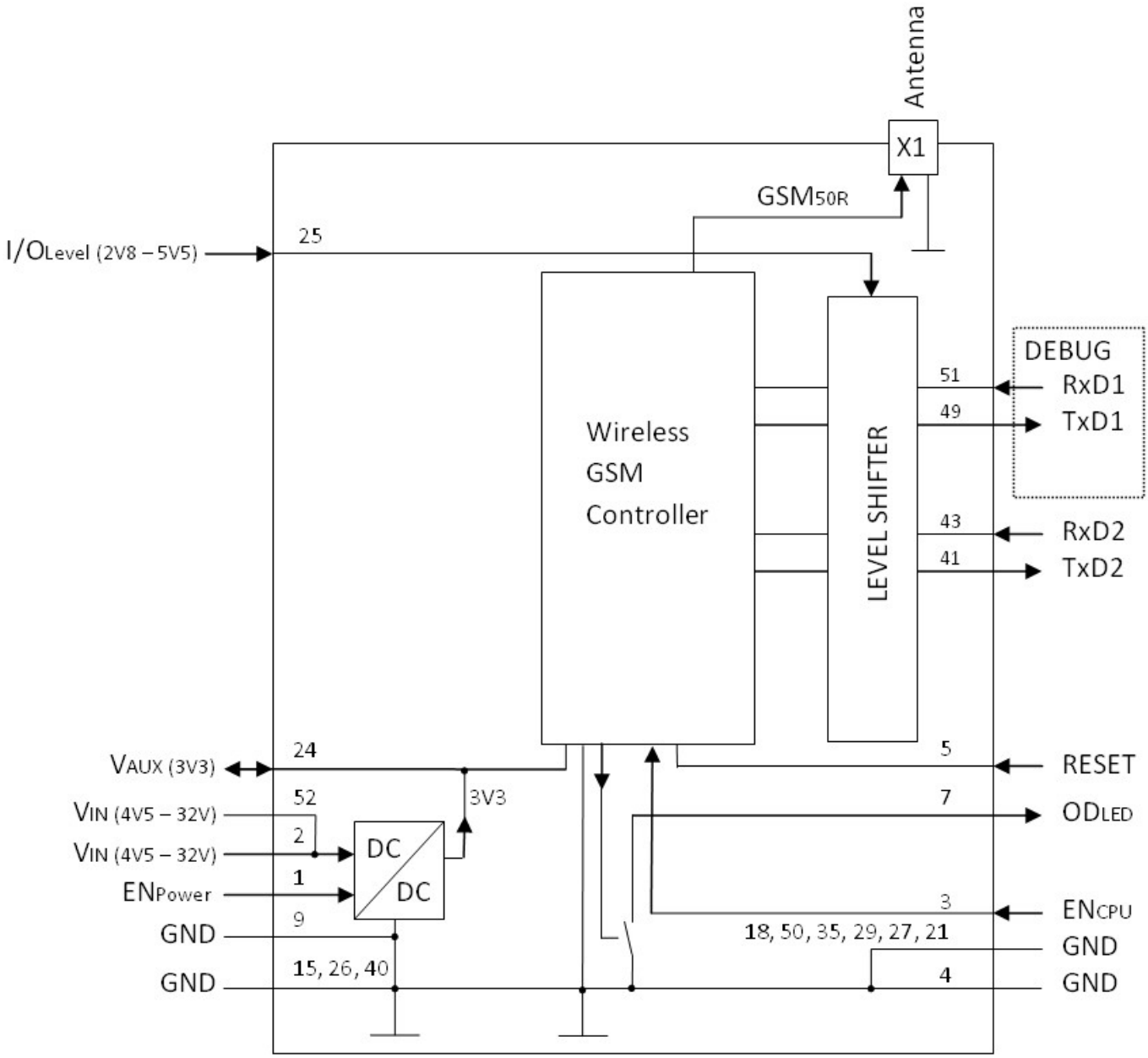
6.2 Übersicht



Übersicht rapidM2M Modul M11

1 U.FL Antennenanschluss (X1)	**2** 52pin Steckerleiste (X2)

6.2.1 Blockschaltbild



Blockschaltbild des M11 Moduls

6.2.2 52pin Steckerleiste (X2)

Pin	Typ	Bezeichnung	Beschreibung
1	I	EN_{Power}	Pin zum Aktivieren des DC/DC-Wandlers, der aus der an V_{IN} angelegten Spannung die interne Versorgungsspannung V_{AUX} erzeugt.
2, 52	I	V_{IN}	Versorgungsspannung (4,5...32VDC) für das rapidM2M Modul M11
3	I	EN_{CPU}	Pin zum Einschalten des Wireless GSM Controllers am rapidM2M Modul M11
4, 9, 15, 18, 21, 26, 27, 29, 35, 40, 50		GND	Masse
5	I	Reset	Pin zum Reseten des Wireless GSM Controllers am rapidM2M Modul M11
6, 8, 10, 12, 14, 16, 28		NC	
7	O	OD_{LED}	Open Drain Ausgang zum Anschluss eines Status-LEDs
11, 13, 17, 19, 20, 22, 23, 30-34, 36-39, 42, 44-48			reserviert für Erweiterungen Es dürfen keine Signale an diese Pins angelegt werden.
24	I/O	V_{AUX}	3,3V Spannungsausgang, wenn das rapidM2M Modul M11 über V_{IN} versorgt wird. Versorgungsspannungseingang (3,2...4,5V), wenn an V_{IN} keine Spannung angelegt wird.
25	I/O	I/O Level	2,8...5,5V Versorgung der Level Shifter zur Anpassung der Signalpegel des M11 Moduls an die umgebende Hardware
41	O	TxD2	TxD Leitung der COM2
43	I	RxD2	RxD Leitung der COM2
49	O	TxD1	TxD Leitung der COM1
51	I	RxD1	RxD Leitung der COM1

6.2.3 Gerätekennzeichnung

Das Typenschild befindet sich auf der Oberseite des Moduls und beinhaltet folgende Angaben:

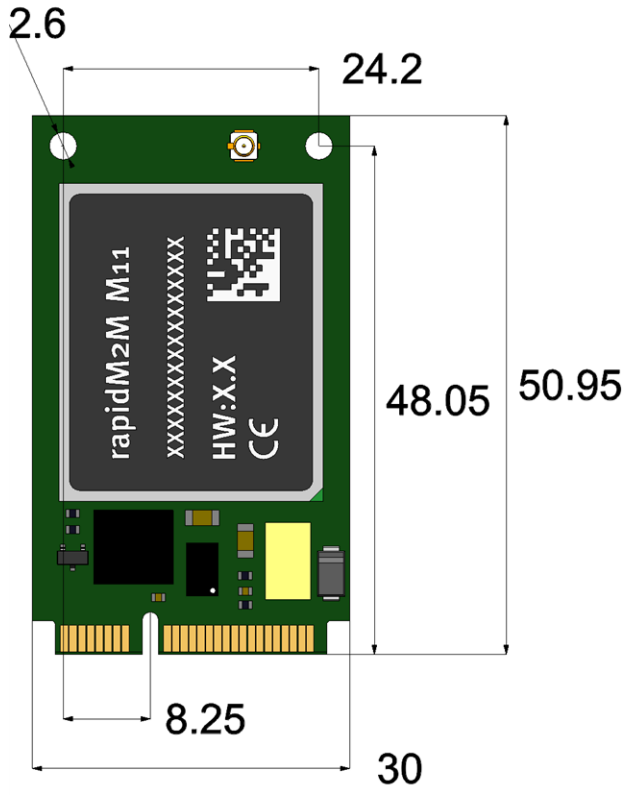
- Typenbezeichnung
- Seriennummer
- CE-Kennzeichnung
- Hardwarerevision

Wichtig für alle Rückfragen und Ersatzteilbestellungen ist die richtige Angabe der Typenbezeichnung und der Seriennummer. Nur so ist eine einwandfreie und schnelle Bearbeitung möglich.

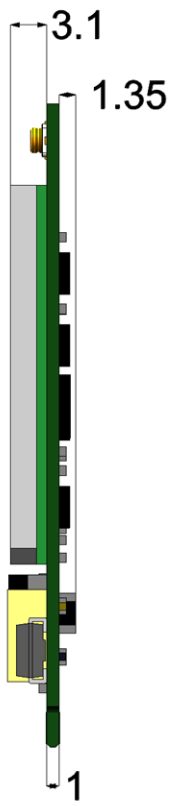


Typenschild rapidM2M Modul M11

6.3 Abmessungen

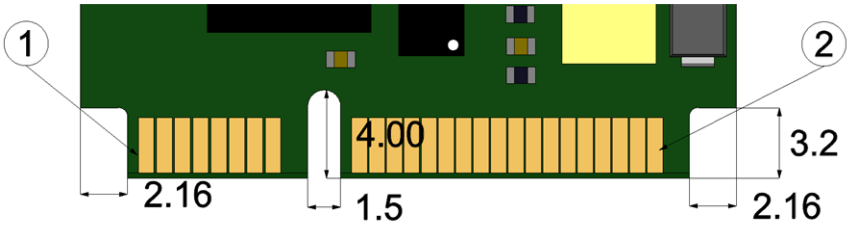


Abmessungen: Breite und Tiefe (Ansicht von Oben)



Abmessungen: Höhe

6.3.1 Abmessungen der 52pin Steckerleiste (X2)



Steckerleiste Detailansicht (Ansicht von Oben)

1 Pin 1	2 Pin 51

6.4 Technische Details

6.4.1 Technische Details zur Energieversorgung

Abhänig von der zur Verfügung stehenden Spannung gibt es zwei Möglichkeiten das rapidM2M Modul M11 zu versorgen:

- V_{AUX}: 3,2...4,5V
- V_{IN}: 4,5...32VDC

6.4.1.1 Versorgung über V_{IN}

Die interne Versorgungsspannung (3,3V) wird vom DC/DC-Wandler des M11 Moduls (siehe "Blockschaltbild" auf Seite 45) erzeugt. Dieser muss dazu allerdings über den Pin "EN_{Power}" aktiviert werden. Um die Belastung der Energiequelle beim Einschalten gering zu halten, ist der DC/DC-Wandler mit einer Soft-Start-Funktion ausgestattet. Dadurch dauert es 50ms bis die interne Versorgungsspannung den Maximalwert erreicht. Wird das rapidM2M Modul M11 über V_{IN} versorgt, fungiert der Pin "V_{AUX}" als Spannungsausgang für die interne 3,3V Versorgungsspannung.

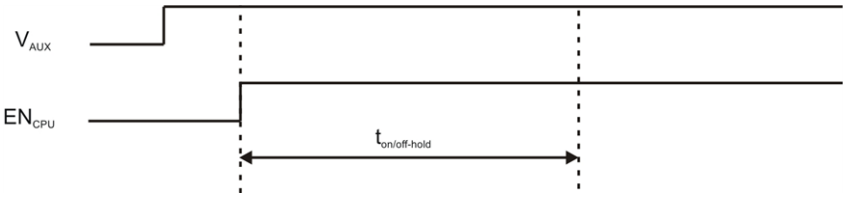
6.4.1.2 Versorgung über V_{AUX}

Der DC/DC-Wandler des M11 Moduls (siehe "Blockschaltbild" auf Seite 45) wird übergangen und die Versorgungsspannung direkt über den Pin "V_{AUX}" in das Modul eingespeist.

6.4.2 EN_{CPU}-Signal

Parameter	min.	typ.	max.
V_{IL}			V_{AUX} * 0,2
V_{IH}	V_{AUX} * 0,8		V_{AUX}
$t_{on/off-hold}$	1,5sec.		
t_{rst}		40ms	

Durch ein "High" für mindestens 1,5sec. ($t_{on/off-hold}$) am Pin "EN_{CPU}" wird der Wireless GSM Controller eingeschaltet. Der Pin "EN_{CPU}" sollte danach, solange das rapidM2M Modul M11 aktiv ist, auf dem "High" Pegel bleiben.



Power-ON Sequenz des Wireless GSM Controllers

Während der Startphase führt der Wireless GSM Controller einen internen Reset durch. Dieser dauert typischerweise 40ms (t_{rst}) und sollte nicht durch einen externen Reset (Pin "Reset") unterbrochen werden.

Ein Abschalten des Wireless GSM Controllers ist bei der aktuellen Firmware des M11 Moduls nicht vorgesehen.

6.4.3 Reset-Signal

Parameter	min.	typ.	max.
V_{IL}	0V		0,57V
V_{IH}	1,33V		
Reset time	200µs		
Int. pull-up		100k	

Um einen Reset auszulösen, muss der Pin "Reset" für min. 200µs mittels Open Collector oder Open Drain auf "Low" gesetzt werden. Der Pin "Reset" sollte nur als Notfall-Reset betrachtet werden, da eine Reset-Prozedur vom Wireless GSM Controller während des Boot-Vorgangs selbstständig durchgeführt wird.

Der Wireless GSM Controller bleibt, solange der Pin "Reset" auf "Low" ist, im Reset-Zustand.

Nach einem Reset ist der Wireless GSM Controller im Zustand "OFF", d.h. er muss mittels EN_{CPU}-Signal (siehe "ENCPU-Signal" auf Seite 48) wieder eingeschaltet werden. Der Pin "EN_{CPU}" kann aber während des Resets auf "High" verbleiben, wodurch der Wireless GSM Controller nachdem er das Signal am Pin "EN_{CPU}" erkannt hat (mind. 1,5sec.) sofort wieder startet.

6.4.4 Status-LED

Das rapidM2M Modul M11 ist mit einem Open Drain Ausgang zum Anschluss eines Status-LEDs (Pin "OD_{LED}") für die Signalisierung des Betriebszustands ausgestattet.

Betriebszustände

Status-LED	Beschreibung
aus	Das rapidM2M Modul M11 ist nicht an die Stromversorgung angeschlossen.
leuchtet	Das rapidM2M Modul M11 ist an die Stromversorgung angeschlossen, aber noch nicht ins Netzwerk eingebucht.
blinkt t_{on}/t_{off} : 0,2/2sec.	Das rapidM2M Modul M11 ist ins Netzwerk eingebucht.
schnelles Blinken t_{on}/t_{off} : 0,2/0,6sec.	Das rapidM2M Modul M11 ist mit dem myDataweb-Server Verbunden. (z.B. Online-Mode oder ALOHA-Übertragungsmodus)

Ein LED und ein Widerstand können direkt zwischen V_{AUX} und dem Open Drain Ausgang OD_{LED} geschaltet werden.

Parameter	min.	typ.	max.
V_{OL}			0,4V
I_{out}			8mA

6.4.5 Technische Details zu den Level Shiftern

Parameter	min.	typ.	max.
$V_{I/O\ Level}$	2,8V		5,5V
V_{IL}			0,15V
V_{IH}	$V_{I/O\ Level}$ - 0,4V		

Um die Signalpegel des M11 Moduls an die umgebende Hardware anzupassen, ist das Modul mit Level Shifter für alle I/O-Signale ausgestattet. Für die Anpassung muss von der umgebenden Hardware eine zusätzliche Versorgungsspannung für die Level Shifter im Bereich von 2,8...5,5V bereit gestellt werden. Über den Pin "I/O Level" kann entweder eine Fremdspannung oder die vom rapidM2M Modul M11 am Pin "V_{AUX}" ausgegebene Spannung eingespeist werden (siehe "Blockschaltbild" auf Seite 45).

6.4.6 Technische Details zur UART1

> ***Wichtiger Hinweis:*** *Die UART1 des M11 Moduls dient nur der Ausgabe von Debug-Informationen. Das Hardware-Handshake ist dabei nicht aktiv. Die UART1 kann nicht für die Datenaufzeichnung verwendet werden.*

Die Pegel der UART1 des M11 Moduls ist abhängig von der am Pin "I/O Level" eingespeisten Versorgungsspannung für die Level Shifter (siehe "Blockschaltbild" auf Seite 45).

Baudrate	115200
Stopbits	2
Parität	N
Datenbits	8
Modi	ASCII

6.4.7 Technische Details zur UART2

Wichtiger Hinweis: *Das im Lieferumfang des Developer Pakets enthaltene rapidM2M Modul M11 unterstützt kein Hardware-Handshake.*

Die Pegel der UART2 des M11 Moduls ist abhängig von der am Pin "I/O Level" eingespeisten Versorgungsspannung für die Level Shifter (siehe "Blockschaltbild" auf Seite 45).

Baudrate	300-115200
Stopbits	1,2
Parität	N, E, O
Datenbits	7, 8
Modi	ASCII

6.4.7.1 Fehlercodes der UART2-Schnittstelle

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
2000	MODULE ERR	0	CHAR TIMEOUT	Der zeitliche Abstand zwischen zwei empfangenen Zeichen ist größer als der über die Eingabemaske zur Konfiguration der Schnittstelle (siehe "Serielle Schnittstelle" auf Seite 61) eingegebene Zeichen-Timeout.
		1	FRAME TIMEOUT	Innerhalb des Datensatz-Timeouts wurde nicht die komplette Antwort empfangen. Diese Zeitspanne wird über die Eingabemaske zur Konfiguration der Schnittstelle (siehe "Serielle Schnittstelle" auf Seite 61) eingegebenen.
		2	INVALID FRAME	Der Antwort-Frame des Sensors ist ungültig, beispielsweise wenn die Spaltenanzahl des Sensors nicht mit der über die Konfigurationsoberfläche eingegebenen übereinstimmt (siehe "Spalte" im Kapitel "Interface Kanäle 1-32" auf Seite 56).

Kapitel 7 Inbetriebnahme

7.1 Inbetriebnahme des Systems

1. Haben Sie alle im Kapitel "Anschluss des Sensors und der Versorgung" auf Seite 39 beschriebenen Schritte durchgeführt, ist das rapidM2M Modul M11 betriebsbereit und sollte bereits die erste Verbindung zum myDataweb-Server durchgeführt haben.
2. Legen Sie eine Messstelle für den Betrieb am myDataweb-Server an (siehe "Benutzerhandbuch für myDataweb-Server" 206.886).

   Hinweis: *Für den Betrieb des M11 Moduls muss eine Messstelle des Typs "myDatalogAIR232 " erstellt werden.*

3. Konfigurieren Sie die erstellte Messstelle entsprechend Ihren Anforderungen (siehe "Messstellenkonfiguration" auf Seite 55).
4. Verknüpfen Sie das rapidM2M Modul M11 mit der erstellten Messstelle (siehe "Messstelle" auf Seite 55).
5. Lösen Sie den ALOHA-Übertragungsmodus über den myDataweb–Server aus (siehe "Grundeinstellung" auf Seite 67), damit die Konfiguration der Messstelle zum rapidM2M Modul M11 übertragen wird. Dies ist nur möglich, wenn die Verbindungsart „Intervall & Wakeup“ konfiguriert ist. Ist "Intervall" als Verbindungsart ausgewählt, warten Sie entweder bis zur nächsten planmäßigen Verbindung oder trennen Sie das rapidM2M Modul M11 von der Versorgungsspannung und drücken Sie nach dem Wiederherstellen der Versorgungsspannung die Taste zum Einschalten des Wireless GSM Controllers S101 (siehe "Taster zum Einschalten des Wireless GSM Controllers am rapidM2M Modul M11 (S101)" auf Seite 28). Bei der Verbindungsart "online" werden die Konfigurationen sofort nach der Eingabe am myDataweb–Server zum rapidM2M Modul M11 übertragen.

7.2 Kommunikation mit dem Gerät testen

1. Legen Sie eine Messstelle für den Betrieb am myDataweb-Server an (siehe "Benutzerhandbuch für myDataweb-Server" 206.886).
2. Konfigurieren Sie die erstellte Messstelle entsprechend Ihren Anforderungen (siehe "Messstellenkonfiguration" auf Seite 55).
3. Verknüpfen Sie das rapidM2M Modul M11 mit der erstellten Messstelle (siehe "Messstelle" auf Seite 55).
4. Lösen Sie den ALOHA-Übertragungsmodus über den myDataweb–Server aus (siehe "Grundeinstellung" auf Seite 67), damit die Konfiguration der Messstelle zum rapidM2M

Modul M11 übertragen wird. Dies ist nur möglich, wenn die Verbindungsart „Intervall & Wakeup“ konfiguriert ist.

5. Warten Sie bis in der Messgeräteliste angezeigt wird, dass sich das Gerät im ALOHA-Übertragungsmodus befindet. Angezeigt wird dieser durch eine Sprechblase mit der Beschriftung "Aloha".



6. Prüfen Sie die "GSM Stärke" im ALOHA-Datenfenster des myDataweb-Servers, welches Sie durch einen Klick auf die Sprechblase mit der Beschriftung "Aloha" erreichen (siehe "Benutzerhandbuch für myDataweb-Server" 206.886).

Hinweis: Ergänzende Erklärung zur Bewertung der "GSM Stärke":

"GSM Stärke"	
› -64dBm	
-64...-73dBm	
-74...-83dBm	
-84...-93dBm	
-94...-107dBm	
‹= -108dBm	

Kapitel 8 Benutzerschnittstelle am myDataweb-Server

8.1 Messstellenkonfiguration

> ***Hinweis:*** *Abhängig vom jeweiligen Benutzerlevel sind einige der in den folgenden Unterkapiteln erwähnten Konfigurationsfelder unter Umständen ausgeblendet. Wenden Sie sich in diesem Fall an den Administrator des myDataweb-Servers.*

Die Eingabemaske zur Konfiguration der Messstelle erreichen Sie durch Klicken auf den Messstellennamen in der Messstellenliste (siehe "Benutzerhandbuch für myDataweb-Server" 206.886).

8.1.1 Messstelle

Kunde
gibt an, welchem Kunden die Messstelle zugeordnet ist

**-Symbol**
Messstelle einem anderen Kunden zuweisen

Name
Messstellenbezeichnung (nicht relevant für die Geräte- oder Datenzuordnung)

Gerät S/N
Seriennummer des Geräts, das mit der Messstelle verknüpft ist (Gerätezuordnung!)

8.1.2 Kommentar

Kommentar
freies Kommentarfeld (wird auch unterhalb des Gerätetyps in der Messstellenliste angezeigt)

8.1.3 Steuerung

Script Typ	aus	Script deaktiviert
	IL	Aktiviert die Scriptabarbeitung und teilt dem rapidM2M Developer Paket M11 mit, dass es sich bei dem Script um eine „Instruction List“ handelt.
	Pawn	Aktiviert die Scriptabarbeitung und teilt dem rapidM2M Developer Paket M11 mit, dass es sich bei dem Script um ein „Pawn Script“ handelt.
Script Vorlage	"Name eines Templates"	Liste der Script-Temlates, passend zur Auswahl des Script Typs, die für das rapidM2M Developer Paket M11 am myDataweb-Server angelegt wurden (zum Anlegen eines Templates siehe "Benutzerhandbuch für myDataweb-Server" 206.886). Das Template wird in das Eingabefenster zum Editieren des Scripts kopiert und kann dort auch noch angepasst werden bevor es ins Gerät geladen wird.
Script	Eingabefenster zum Editieren des Scripts, das in das rapidM2M Developer Paket M11 geladen wird	

8.1.4 Interface Kanäle 1-32

8.1.4.1 Basis

Bezeichnung 1-32
frei wählbare Kanalbezeichnung für die Interface Kanäle

Interface
Basiseinstellung für den Interface Kanal

aus	*---*	*Interface Kanal deaktiviert*
RS232	*Spalte*	*Durch das "Zahlen Trennzeichen" (siehe "Serielle Schnittstelle" auf Seite 61) wird die ASCII-Datenmessage des digitalen Sensors in einzelne Messwerte unterteilt. Der Konfigurationsparameter „Spalte“ gibt an, welcher dieser Messwerte mit dem Interface Kanal verknüpft/aufgezeichnet wird.*
	Min	*Definiert das untere Skalenende der Zeigerinstrumente*
	Max	*Definiert das obere Skalenende der Zeigerinstrumente*
	Einheit	*String, der als Messwerteinheit von allen Anzeigeelementen des Servers verwendet wird*
	Nachkomma	*Anzahl der Nachkommastellen, die von allen Anzeigeelementen des Servers verwendet wird*

8.1.4.2 Alarme

Warnung	Wert niedrig	Wenn der Messwert auf oder unter diesen Wert fällt, wird eine Warnung ausgelöst.
	Wert hoch	Wenn der Messwert auf oder über diesen Wert steigt, wird eine Warnung ausgelöst.
Alarm	Wert niedrig	Wenn der Messwert auf oder unter diesen Wert fällt, wird ein Alarm ausgelöst.
	Wert hoch	Wenn der Messwert auf oder über diesen Wert steigt, wird ein Alarm ausgelöst.
Hyst %	Hysterese für Entwarnung (z.B. Hyst=5%, Alarm od. Warnung bei 100 -› Entwarnung bei 95)	

8.1.4.3 Trigger

QU	Schnelle Aufzeichnung (Aufzeichnungsintervall = Aufzeichnungsintervall / Faktor)	
SL	Langsame Aufzeichnung (Aufzeichnungsintervall = Aufzeichnungsintervall * Faktor)	
RO	Aufzeichnung einschalten	
RF	Aufzeichnung ausschalten	
XM	Übertragung auslösen	
ON	Online-Modus aktivieren	
Schwelle	Schwellen für das Auslösen des Triggers	
	größer gleich	Der Messwert muss zum Auslösen des Triggers größer/gleich der Schwelle sein.
	kleiner gleich	Der Messwert muss zum Auslösen des Triggers kleiner/gleich der Schwelle sein.

8.1.5 Interface Kanäle 33-64

8.1.5.1 Basis

Bezeichnung 33-64
frei wählbare Kanalbezeichnung für die Interface Kanäle

Die Bedeutung der restlichen Konfigurationsparameter in diesem Konfigurationsabschnitt entspricht dem Konfigurationsabschnitt "Interface Kanäle 1-32" (siehe "Interface Kanäle 1-32" auf Seite 56).

8.1.6 Interface Ausgabekanäle 1-32

Bezeichnung 1-32
frei wählbare Kanalbezeichnung für den Ausgabekanal

Modus

Basiseinstellung für den Ausgabekanal

aus	*Ausgabekanal ist deaktiviert*	
ein	*Ausgabekanal ist aktiviert*	
	Stellwert	*Stellwert, der ausgegeben werden soll*

Hinweis:

Sind ein oder mehrere Ausgabekanäle aktiv, gibt das rapidM2M Developer Paket M11 , nachdem es die ASCII-Datenmessage des digitalen Sensors eingelesen und verarbeitet hat, eine ASCII-Datenmessage mit den Stellwerten aus. Diese wurden entweder über die Eingabemaske am myDataweb -Server eingegebenen oder mittels "Instruction List" (siehe "Instruction List" auf Seite 71) vom Gerät selbst berechnet. Für die Ausgabe wird das selbe Frameformat verwendet, das das rapidM2M Developer Paket M11 auch vom digitalen Sensor erwartet. Als Zahlen Trennzeichen wird das erste unter "Serielle Schnittstellen" (siehe "Serielle Schnittstelle" auf Seite 61) konfigurierte Zahlen Trennzeichen verwendet.

8.1.7 Interface Ausgabekanäle 33-64

Bezeichnung 33-64

frei wählbare Kanalbezeichnung für den Ausgabekanal

Die Bedeutung der restlichen Konfigurationsparameter in diesem Konfigurationsabschnitt entspricht dem Konfigurationsabschnitt "Interface Ausgabekanäle 1-32" (siehe "Interface Ausgabekanäle 1-32" auf Seite 57).

8.1.8 Berechnete Kanäle

Hinweis: *Die Werte der berechneten Kanäle werden jedes Mal direkt bei der Datenausgabe (Anzeige am myDataweb-Server oder Download vom myDataweb-Server) berechnet. Sie sind nicht in der Datenbank des Servers gespeichert.*

8.1.8.1 Basis

Bezeichnung 1-5

frei wählbare Kanalbezeichnung für die berechneten Kanäle

Modus

mögliche Berechnungsmodi für die berechneten Kanäle

aus	*---*	*berechneter Kanal deaktiviert*
Tabelle	*Min*	*Definiert das untere Skalenende der Zeigerinstrumente*
	Max	*Definiert das obere Skalenende der Zeigerinstrumente*
	Einheit	*String, der als Messwerteinheit von allen Anzeigeelementen des Servers verwendet wird*
	Nachkomma	*Anzahl der Nachkommastellen, die von allen Anzeigeelementen des Servers verwendet wird*
Digital	*Invertieren*	*invertiert das Eingangssignal*
+, -, x, /	*Min*	*Definiert das untere Skalenende der Zeigerinstrumente*
	Max	*Definiert das obere Skalenende der Zeigerinstrumente*
	Einheit	*String, der als Messwerteinheit von allen Anzeigeelementen des Servers verwendet wird*
	Nachkomma	*Anzahl der Nachkommastellen, die von allen Anzeigeelementen des Servers verwendet wird*



Element nach unten verschieben



Element nach oben verschieben

8.1.8.2 Berechnung

aus	---	berechneter Kanal deaktiviert
Tabelle	Quellkanal	Auswahl des Kanals, aus dem die Eingangsdaten herangezogen werden
		öffnet die Maske zur Eingabe der Wertetabelle (Zwischen den Tabellenzeilen wird linear interpoliert, für Werte außerhalb der definierten Tabelle wird linear extrapoliert.)
Digital	Quellkanal	Auswahl des Kanals, aus dem die Eingangsdaten herangezogen werden
	High Level	Schwelle für Signalerkennung
+, -, x, /	Quellkanal	Auswahl des Kanals, aus dem die Eingangsdaten herangezogen werden
	+, -, x, /	
	Quellkanal	Auswahl des Kanals, aus dem die Eingangsdaten herangezogen werden

8.1.8.3 Alarme

Hinweis: *Die Überprüfung der Alarmschwellen kann bei berechneten Kanälen erst erfolgen, wenn das Gerät die Messdaten an den myDataweb-Server übermittelt hat.*

Alarm niedrig	Wenn der Messwert auf oder unter diesen Wert fällt, wird ein Alarm ausgelöst.
Alarm hoch	Wenn der Messwert auf oder über diesen Wert steigt, wird ein Alarm ausgelöst.
Hyst %	Hysterese für Entwarnung (z.B. Hyst=5%, Alarm od. Warnung bei 100 -› Entwarnung bei 95)

8.1.9 Interne Kanäle

8.1.9.1 Basis

Bezeichnung GSM Stärke	frei wählbare Kanalbezeichnung für die GSM-Feldstärke	
	Einheit	String, der als Messwerteinheit von allen Anzeigeelementen des Servers verwendet wird

8.1.9.2 Alarme

Warnung	Wert niedrig	Wenn der Messwert auf oder unter diesen Wert fällt, wird eine Warnung ausgelöst.
	Wert hoch	Wenn der Messwert auf oder über diesen Wert steigt, wird eine Warnung ausgelöst.
Alarm	Wert niedrig	Wenn der Messwert auf oder unter diesen Wert fällt, wird ein Alarm ausgelöst.
	Wert hoch	Wenn der Messwert auf oder über diesen Wert steigt, wird ein Alarm ausgelöst.
Hyst %	Hysterese für Entwarnung (z.B. Hyst=5%, Alarm od. Warnung bei 100 -› Entwarnung bei 95)	

8.1.9.3 Trigger

QU	Schnelle Aufzeichnung (Aufzeichnungsintervall = Aufzeichnungsintervall / Faktor)	
SL	Langsame Aufzeichnung (Aufzeichnungsintervall = Aufzeichnungsintervall * Faktor)	
RO	Aufzeichnung einschalten	
RF	Aufzeichnung ausschalten	
XM	Übertragung auslösen	
ON	Online-Modus aktivieren	
Schwelle	Schwellen für das Auslösen des Triggers	
	größer gleich	Der Messwert muss zum Auslösen des Triggers größer/gleich der Schwelle sein.
	kleiner gleich	Der Messwert muss zum Auslösen des Triggers kleiner/gleich der Schwelle sein.

8.1.10 Serielle Schnittstelle

Profil

Basiseinstellung für die serielle Schnittstelle

Standard (1/3)	*Baudrate*	*300*	*Auswahl der benötigten Baudrate*
		600	
		1200	
		2400	
		4800	
		9600	
		19200	
		38400	
		57600	
		115200	
	Stoppbits	*1*	*Auswahl der Anzahl der benötigten Stoppbits*
		2	
	Parität	*keine*	*Auswahl der benötigten Parität*
		ungerade	
		gerade	
	Datenbits	*7*	*Anzahl der zu verwendenden Datenbits*
		8	

Standard (2/3)	*Retry*	*Bei einem Kommunikationsfehler wird das entsprechende Kommando ein Mal wiederholt. Der Fehler wird erst aufgezeigt, wenn auch dieser zweite Versuch fehlschlägt.*	
	Datensatz Struktur	*...[CR][LF]*	*Jeder Frame muss mit den beiden Zeichen „[CR][LF]“ beendet werden.*
		...[ETX]	*Jeder Frame muss mit dem Zeichen „[ETX]“ beendet werden.*
		[STX]...[CR][LF]	*Jeder Frame muss mit dem Zeichen „[STX]“ beginnen und mit den beiden Zeichen „[CR][LF]“ beendet werden.*
		[STX]...[ETX]	*Jeder Frame muss mit dem Zeichen „[STX]“ beginnen und mit dem Zeichen „[ETX]“ beendet werden.*
	Datensatz Timeout	*0...Kein Timeout* *Zeit innerhalb der der angeschlossene digitale Sensor die komplette Antwort auf eine Anfrage senden muss. Bei Überschreiten der Zeit wird ein Kommunikationsfehler erkannt.*	
	Zeichen Timeout	*0...Kein Timeout* *Maximal zulässiger Zeitabstand zwischen dem vollständigen Empfang zweier Zeichen. Bei Überschreiten der Zeit wird ein Kommunikationsfehler erkannt. Die Überwachung beginnt nach dem Empfang des ersten Zeichens der Antwort des digitalen Sensors auf eine Anfrage.*	
	Zahlen Trennzeichen	*Unterteilt die ASCII-Datenmessage des digitalen Sensors in einzelne Messwerte. Über das Eingabefeld können bis zu 10 verschiedene Trennzeichen eingegeben werden. Als Trennzeichen werden auch folgende Escape-Codes unterstützt:* *\a \b \f \n \r \t \v \\ \? \' \" \xhh* *Bei der Verwendung von Escape-Codes reduziert sich die max. Anzahl an Trennzeichen.*	
	Zahlenformat	*1,000.00*	*"," wird als Tausendertrennzeichen verwendet.* *"." wird als Dezimaltrennzeichen verwendet.*
		1.000,00	*"." wird als Tausendertrennzeichen verwendet.* *"," wird als Dezimaltrennzeichen verwendet.*
	Aufwärmzeit	*Zeit vom Einschalten des rapidM2M Developer Paket M11 bis zum Senden der Init Sequenz.*	

Standard (3/3)	*Halten*	*Halten des letztgültigen Messwerts für x Messzyklen*	
		aus	*Funktion deaktiviert*
		1-5	*Anzahl der Messzyklen, für die der Messwert gehalten wird, bevor der Fehlerwert ausgegeben wird*
		ein	*Im Fehlerfall wird der zuletzt gültige Messwert so lange gehalten, bis wieder ein neuer gültiger Messwert vorliegt.*
	Init Sequenz	*Initialisierungsbefehl, der nach dem Einschalten des rapidM2M Developer Paket M11 nach Ablauf der Aufwärmzeit an den digitalen Sensor gesendet wird.* *Folgende Escape Codes unterstützt:* *\a \b \f \n \r \t \v \\ \? \' \" \xhh*	
	Messdaten Sequenz	*Befehl, der zu jedem Messzeitpunkt an den digitalen Sensor gesendet wird.* *Folgende Escape Codes werden unterstützt:* *\a \b \f \n \r \t \v \\ \? \' \" \xhh*	

YSI 6 Series	*Voreinstellung für die YSI 6-Serie Multiparameter Wasserqualitäts-Messsonden*		
	Baudrate	*300*	*Auswahl der benötigten Baudrate*
		600	
		1200	
		2400	
		4800	
		9600	
		19200	
		38400	
		57600	
		115200	
	Retry	*Bei einem Kommunikationsfehler wird das entsprechende Kommando ein Mal wiederholt. Der Fehler wird erst aufgezeigt, wenn auch dieser zweite Versuch fehlschlägt.*	
	Halten	*Halten des letztgültigen Messwerts für x Messzyklen*	
		aus	*Funktion deaktiviert*
		1-5	*Anzahl der Messzyklen, für die der Messwert gehalten wird, bevor der Fehlerwert ausgegeben wird*
		ein	*Im Fehlerfall wird der zuletzt gültige Messwert so lange gehalten, bis wieder ein neuer gültiger Messwert vorliegt.*

<table>
<tr><td rowspan="14">ISCO Periodic Update</td><td colspan="3">Voreinstellung für einen ISCO Probennehmer</td></tr>
<tr><td rowspan="10">Baudrate</td><td>300</td><td rowspan="10">Auswahl der benötigten Baudrate</td></tr>
<tr><td>600</td></tr>
<tr><td>1200</td></tr>
<tr><td>2400</td></tr>
<tr><td>4800</td></tr>
<tr><td>9600</td></tr>
<tr><td>19200</td></tr>
<tr><td>38400</td></tr>
<tr><td>57600</td></tr>
<tr><td>115200</td></tr>
<tr><td>Retry</td><td colspan="2">Bei einem Kommunikationsfehler wird das entsprechende Kommando ein Mal wiederholt. Der Fehler wird erst aufgezeigt, wenn auch dieser zweite Versuch fehlschlägt.</td></tr>
<tr><td>Datensatz Timeout</td><td colspan="2">0…Kein Timeout

Zeit innerhalb der der angeschlossene digitale Sensor die komplette Antwort auf eine Anfrage senden muss. Bei Überschreiten der Zeit wird ein Kommunikationsfehler erkannt.</td></tr>
<tr><td rowspan="3">Halten</td><td colspan="2">Halten des letztgültigen Messwerts für x Messzyklen</td></tr>
<tr><td>aus</td><td>Funktion deaktiviert</td></tr>
<tr><td>1-5</td><td>Anzahl der Messzyklen, für die der Messwert gehalten wird, bevor der Fehlerwert ausgegeben wird</td></tr>
<tr><td>ein</td><td>Im Fehlerfall wird der zuletzt gültige Messwert so lange gehalten, bis wieder ein neuer gültiger Messwert vorliegt.</td></tr>
</table>

8.1.11 Alarmierung

Quittierung	Standard	Für die Entscheidung, ob die Alarme automatisch oder manuell quittiert werden müssen, wird die globale Servereinstellung herangezogen (siehe "Benutzerhandbuch für myDataweb-Server" 206.886).
	automatisch	Alarme werden automatisch quittiert, sobald alle Benachrichtigungen versendet wurden. Wurden auch SMS versendet, welche einen Tarif mit Sendebestätigungsfunktion haben, so wird mit der Quittierung auf die Sendebestätigung gewartet.
	manuell	Alarme müssen durch den Anwender quittiert werden.
Übertragungsausfall Alarm	Alarmierung, falls sich das Instrument länger als die eingestellten Übertragungszyklen nicht meldet	
Transfervolumen	Standard	Die Einstellung für den Transfervolumenalarm wird von der globalen Servereinstellung übernommen (siehe "Benutzerhandbuch für myDataweb-Server" 206.886).
	aus	Der Transfervolumenalarm ist deaktiviert.
	individuell	Die Schwelle, bei der der Transfervolumenalarm ausgelöst werden soll, kann in das nebenstehende Feld in KB eingegeben werden.
Bei Alarm	A	Der Alarm wird in der Alarmliste aufgezeichnet.
	Ü	Eine sofortige Übertragung wird ausgelöst.
Bei Warnung	A	Der Alarm wird in der Alarmliste aufgezeichnet.
	Ü	Eine sofortige Übertragung wird ausgelöst.
Bei Störung-Alarm	A	Der Alarm wird in der Alarmliste aufgezeichnet.
	Ü	Eine sofortige Übertragung wird ausgelöst.
Bei Störung-Warnung	A	Der Alarm wird in der Alarmliste aufgezeichnet.
	Ü	Eine sofortige Übertragung wird ausgelöst.
Testalarm	Es wird an alle Empfänger (Alarmrufplan Typ "parallel"), bzw. den ersten Empfänger (Alarmrufplan Typ "seriell"), eine Testnachricht geschickt, sobald diese Seite gespeichert wurde. Die Checkbox wird danach wieder automatisch rückgesetzt.	

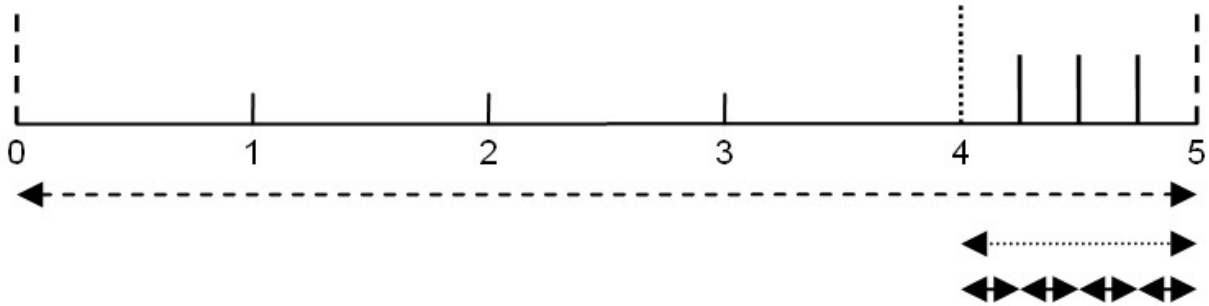
8.1.12 Grundeinstellung

Verbindungsart	Intervall	Das Gerät meldet sich im Übertragungsintervall.
	Intervall & Wakeup	Das Gerät meldet sich im Übertragungsintervall und kann über den Server in den ALOHA-Übertragungsmodus versetzt werden. 
	online	Das Gerät trennt die Verbindung nicht und übermittelt kontinuierlich die Messdaten.
Aloha/Wakeup Dauer	Dauer der Aloha/Wakeup Verbindung	
Übertragungsintervall	zeitlicher Abstand der Übertragungen	
Aufzeichnungsintervall	zeitlicher Abstand der Messdatenaufzeichnungen	
sofortige Messung bei Wakeup	Es wird auch eine Messung gestartet, wenn das Gerät über den Server in den ALOHA-Übertragungsmodus versetzt wird.	
Divisor schnelle Aufzeichnung	Aufzeichnungsintervall = Aufzeichnungsintervall / Faktor (ab Triggerung)	
Faktor langsame Aufzeichnung	Aufzeichnungsintervall = Aufzeichnungsintervall * Faktor (ab Triggerung)	
Messintervall	zeitlicher Abstand der Messungen (00:00 gleich wie Aufzeichnungsintervall)	
Burst Intervall	Zeitspanne, in der im Messintervall gemessen wird, bevor aufgezeichnet wird. (00:00 Messintervall ist durchgehend aktiviert)	
Zeitzone	Regionseinstellungen (nicht relevant für Rohmessdaten, da diese in UTC gespeichert werden)	
Sommerzeit	Konfiguration für automatische Zeitumstellung	
	standard	Die Konfiguration für die Zeitumstellung wird von der globalen Servereinstellung übernommen (siehe "Benutzerhandbuch für myDataweb-Server" 206.886).
	aus	automatische Zeitumstellung deaktiviert
	USA	vordefinierte Einstellung für den amerikanischen Raum
	EU	vordefinierte Einstellung für den europäischen Raum
Positionsintervall	zeitlicher Abstand der Positionsaktualisierung	
Standard Auswertung	Auswertung, welche durch einen Klick auf den Gerätelink in den Karten geladen wird	

Hinweis:

Beispiel zur Erklärung Aufzeichnungs-, Mess-, Burstintervall

Grundeinstellung		*Aufzeichnungsintervall*	*5 min.*
		Burst Intervall	*1 min.*
		Messintervall	*15 sec.*



Hinweis: *Ergänzende Erklärung zu den Verbindungsarten*

Verbindungsart	*Energieverbrauch*	*Datenvolumen*	*Reaktionszeit*
online			
Intervall & Wakeup			
Intervall			

8.1.13 FTP Export Einstellungen

Hinweis: *Dieser Konfigurationsabschnitt ist nur sichtbar, wenn die Lizenz "FTP Agent Extended" für den myDataweb-Server freigeschaltet wurde.*

FTP Export Profil	aus	FTP Export deaktiviert
	"Name eines FTP Export Profils"	Liste mit den FTP-Export-Profilen, die am myDataweb-Server angelegt wurden (zum Anlegen eines FTP-Export-Profils siehe "Benutzerhandbuch für myDataweb-Server" 206.886).
Einstellungen des gewählten Profils	zeigt eine Übersicht der wichtigsten Parameter des ausgewählten FTP-Export-Profils an	
FTP Verzeichnis	ermöglicht es, das Standardverzeichnis des ausgewählten FTP-Export-Profils zu überschreiben	

8.2 Gerätekonfiguration

> ***Hinweis:*** *Abhängig vom jeweiligen Benutzerlevel sind einige der in den folgenden Unterkapiteln erwähnten Konfigurationsfelder unter Umständen ausgeblendet. Wenden Sie sich in diesem Fall an den Administrator des myDataweb-Servers.*

Die Eingabemaske zur Konfiguration des Geräts erreichen Sie durch Klicken auf die Seriennummer in der Messstellenliste (siehe "Benutzerhandbuch für myDataweb-Server" 206.886) oder durch Klicken auf den Gerätenamen in der Messgeräteliste (siehe "Benutzerhandbuch für myDataweb-Server" 206.886).

8.2.1 Kommentar

Kommentar

freies Kommentarfeld (wird auch unterhalb des Messstellennamens in der Messgeräteliste angezeigt)

8.2.2 Messgerät

Kunde	Name des Kunden, dem das Messgerät zugeordnet ist
IMSI	IMSI des verwendeten SIM-Chips bzw. der SIM-Karte
Seriennummer	Seriennummer des Geräts
Geräteklasse	Damit ein Gerät mit einer Messstelle verbunden werden kann, müssen die Geräteklasse der Messstelle und die des Geräts übereinstimmen.
Telefonnummer	Telefonnummer der SIM-Karte. An diese Nummer werden die Steuer-SMS (z.B. Wakeup) gesendet. Format: +43555837465
Geräte Flags	zusätzliche Information zur Geräteklasse (für interne Verwendung)
Firmware Version	aktuell installierte Softwareversion des Messcontrollers
Modem Version	aktuell installierte Softwareversion des Modemcontrollers
OS Version	OS-Version des Modems
Letzter Verbindungsaufbau	jeweils der letzte Zeitstempel der betreffenden Operation
Letzter Wakeup	
Letzter Verbindungsabbau	
Letzter Übertragungsfehler	
Letzte Aloha Verbindung	

Firmware Update	aus	Firmware Update ist deaktiviert.
	ein	Sobald eine neue Version des ausgewählten Firmware-Typs vorhanden ist, wird diese sofort installiert.
	auch wenn tag nicht vorhanden	Firmware wird auch ans Gerät übertragen, wenn das Gerät den aktuellen Firmwarestand nicht an den Server übermittelt hat. (NICHT EMPFOHLEN!)
	Downgrade erlauben	ermöglicht es, eine ältere Firmwareversion als die im Gerät vorhandene zu installieren. (NICHT EMPFOHLEN!)
	einmalig	Führt einmalig ein Firmware Update durch. Ist keine neue Firmware verfügbar oder wurde die Firmware erfolgreich installiert, wird das Firmware Update automatisch auf „aus“ geschaltet.
Firmware Typ	Released	Nur Firmwareversionen bei denen sowohl interner Test als auch Feldtest erfolgreich waren, werden installiert (Fehlfunktionen nahezu ausgeschlossen).
	Release Candidate	Nur Firmwareversionen bei denen der interne Test erfolgreich war, werden installiert (Fehlfunktionen nicht ausgeschlossen).
	Beta Release	Auch Firmwareversionen bei denen noch nicht alle internen Tests erfolgreich abgeschlossen sind, werden installiert (Fehlfunktionen durchaus möglich).

8.2.3 Gerätespezifische Einstellungen

Betriebsart	HOLD	Messung: aus, Übertragung: ein
	RUN	Messung: ein, Übertragung: ein
	Aloha	Für interne Verwendung reserviert

8.2.4 GPRS

SIM Tarif

ausgewählter SIM-Tarif

Kapitel 9 Instruction List

9.1 Allgemein

Das folgende Kapitel beschreibt die Funktionalität des Instruction List Parsers nach IEC 61131-3:2003.

Es gibt zwei Möglichkeiten, um eine Instruction List für das rapidM2M Developer Paket M11 zu erstellen:

- Direkte Eingabe in das Eingabefenster „Script“ im Konfigurationsabschnitt „Steuerung“
- Verwenden eines Script-Templates, das am myDataweb–Server angelegt wurde

9.1.1 Direkte Eingabe einer Instruction List

Die Eingabe der Instruction List erfolgt über den Konfigurationsabschnitt „Steuerung“ (siehe "Steuerung" auf Seite 56) der Eingabemaske zur Konfiguration der Messstelle. Als „Script Type“ muss „IL“ ausgewählt werden, damit das rapidM2M Developer Paket M11 die unter „Script“ eingegebenen Befehle als Instruction List interpretiert.

9.1.2 Verwendung eines Script-Templates

Über die Listenauswahl „Steuerprogramm“ im Konfigurationsabschnitt „Steuerung“ (siehe "Steuerung" auf Seite 56) der Eingabemaske zur Konfiguration der Messstelle können vordefinierte Script-Templates ausgewählt werden (zum Anlegen eines Templates siehe "Benutzerhandbuch für myDataweb-Server" 206.886). Nach der Auswahl des Steuerprogramms wird das Template in das Eingabefenster „Script“ kopiert und kann dort auch angepasst werden. Als „Script Type“ muss auch bei dieser Methode „IL“ ausgewählt werden, damit das rapidM2M Developer Paket M11 die Befehle als Instruction List interpretiert.

> ***Hinweis:*** *Wird mittels Instruction List direkt auf einen Ausgang geschrieben, überschreibt der ermittelte Wert den über die Eingabemaske am myDataweb-Server eingegebenen Stellwert.*

9.2 Implementierung-Details

9.2.1 Gemeinsame Elemente

Nr. der Tabelle	Attribut	Beschreibung	Beispiel/Erklärung
1	2	Kleinbuchstaben	
	3a	Nummernzeichen (#)	
2	1	Großbuchstaben und Zahlen	IW215 IW215Z QX75 IDENT
3	1	Kommentare	(**********) (* Ein eingerahmter Kommentar *) (**********)
4	1	Ganzzahlige (integer) Literale	-12 0 123_456 +986
	2	Reelle (real) Literale	-12.0 0.0 0.456 3.14159_26
	7	Boolesche Null und Eins	0 1
7	1a	Zeitdauer ohne Unterstriche: kurzes Präfix	T#14ms T#-14ms T#14.7s T#14.7m T#14.7h t#14.7d t#25h15m t#5d14h12m18s3.5ms
8	6	Literale für Datum und Zeit (kurzes Präfix)	DT#1984-06-25-15:36:55 dt#1984-06-25-15:36:55
10	1	BOOL	Datentyp: boolesche, 1Bit
	4	DINT	Datentyp: doppelte ganze Zahl (double integer), 32Bit
	8	UDINT	Datentyp: vorzeichenlose, doppelte, ganze Zahl, 32Bit
	10	REAL	Datentyp: reelle Zahl, 32Bit
	12	TIME	Datentyp: Zeitdauer
	15	DATE_AND_TIME oder DT	Datentyp: Datum und Uhrzeit
	19	DWORD	Datentyp: Bit-Folge 32, 32Bit
15	1	I (Speicherort Eingang)	%IW215 %IX2.5
	2	Q (Speicherort Ausgang)	%QX75 %QR1
	3	M (Speicherort Merker)	%MD48 %MR1000
	4	X ((Einzel-) Bit-Größe)	
	5	kein ((Einzel-) Bit-Größe)	
	6	B (Byte- (8 Bits) Größe)	
	7	W (Wort- (16 Bits) Größe)	
	8	D (Doppelwort- (32 Bits) Größe)	

Nr. der Tabelle	Attribut	Beschreibung	Beispiel/Erklärung
23	1	ABS	absolut
	2	SQRT	Quadratwurzel LD %IRo SQRT
24	12	ADD	OUT := IN1 + IN2
	13	MUL	OUT := IN1 * IN2
	14	SUB	OUT := IN1 - IN2
	15	DIV	OUT := IN1 / IN2
	16	MOD	OUT := IN1 modulo IN2
25	1	SHL	OUT := IN1 « IN2
	2	SHR	OUT := IN1 » IN2
	3	ROR	OUT := IN1 rotate right IN2
	4	ROL	OUT := IN1 rotate left IN2
26	5	AND	OUT := IN1 & IN2
	6	OR	OUT := IN1 or IN2
	7	XOR	OUT := IN1 xor IN2
	8	NOT	OUT := not IN1
27	2a	MAX	Vergleicht den Wert im Accu und den Operanden und liefert den Größeren der beiden
	2b	MIN	Vergleicht den Wert im Accu und den Operanden und liefert den Kleineren der beiden
28	5	GT	OUT := IN1 > IN2
	6	GE	OUT := IN1 >= IN2
	7	EQ	OUT := IN1 = IN2
	8	LE	OUT := IN1 <= IN2
	9	LT	OUT := IN1 < IN2
	10	NE	OUT := IN1 <> IN2
30	1a	ADD	TIME := TIME + TIME
	3a	ADD	DT := DT + TIME
	4a	SUB	TIME := TIME - TIME
	8a	SUB	DT := DT - TIME
	9a	SUB	TIME := DT - DT

9.2.2 IL-Syntaxelemente

Nr. der Tabelle	Attribut	Beschreibung	Beispiel/Erklärung
52 (1/3)	1	LD	setzt aktuelles Ergebnis dem Operanden gleich LD %IR0 LDN %MX4.0
	2	ST	speichert aktuelles Ergebnis auf die Operanden-Adresse ST %QR0 STN %MX1000.2
	4	AND	Logisches UND AND 8 ANDN 12
	6	OR	Logisches ODER OR 8 ORN 12
	7	XOR	Logisches Exklusiv-ODER XOR 8 XORN 12
	7a	NOT	Logische Negation (Einer-Komplement) NOT %MX0:1
	8	ADD	Addition LD 3 ADD 4 LD T#12s ADD T#1h23m LD DT#1984-06-25-15:36:55 ADD T#1h23m

Nr. der Tabelle	Attribut	Beschreibung	Beispiel/Erklärung
52 (2/3)	9	SUB	Subtraktion LD 3 SUB 2 LD T#1d3h12s SUB T#1h23m LD DT#1984-06-25-15:36:55 SUB T#1h23m LD DT#1988-06-25-15:36:55 SUB DT#1984-06-25-15:36:55
	10	MUL	Multiplikation LD 2 MUL 3
	11	DIV	Division LD 12 DIV 6
	11a	MOD	Modulo-Division LD 12 MOD 5
	12	GT	Vergleich: › LD 12 GT 10
	13	GE	Vergleich: ›= LD 12 GE 10
	14	EQ	Vergleich: = LD 12 EQ 12
	15	NE	Vergleich: ‹› LD 12 NE 10

Nr. der Tabelle	Attribut	Beschreibung	Beispiel/Erklärung
52 (3/3)	16	LE	Vergleich: <= LD 12 LE 13
	17	LT	Vergleich: < LD 12 LT 10
	18	JMP	Sprung zur Marke JMP {label} JMPC {label} conditional JMPCN {label} conditional not LD %IDo LT 10 JMPCN label1 ADD 3 label1: SUB 2

9.3 Instruction List Fehlercodes

Zu jedem Messzeitpunkt, nachdem alle Messwerte erzeugt wurden, wird das Script der Instruction List zeilenweise durchlaufen. Sollte dabei ein Fehler auftreten, stoppt die Scriptausführung und trägt den entsprechenden Fehlercode in das Gerätelog ein. Als Parameter zum Fehlercode wird die Nummer der Zeile eingetragen, in der der Fehler aufgetreten ist. Eine Anleitung zum Auswerten des Gerätelogs finden Sie im Kapitel „Auswerten des Gerätelogs“ (siehe "Auswerten des Gerätelogs" auf Seite 138).

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
3001	UNKNOWN ERROR	## [1)]	---	unbekannter Fehler
3002	DECODING ERROR	## [1)]	---	Fehler beim Dekodieren der Script-Zeile
3003	OPERATOR ERROR	## [1)]	---	ungültiger Operator (siehe "IL-Syntaxelemente" auf Seite 74 und "Erweiterungen" auf Seite 78)
3004	TYPE MISMATCH	## [1)]	---	Datentyp des Operanden und des Accu sind inkompatibel.

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
3005	PARSE ERROR	## [1]	---	Kein Label, Kommentar, Operator oder Operand in der Script-Zeile gefunden
3006	UNKNOWN OPERAND	## [1]	---	ungültiger Operand (siehe "Gemeinsame Elemente" auf Seite 72, Tabelle Nr. 4, Nr. 7, Nr. 8, Nr. 15)
3007	INVALID DIRECT ADDRESS	## [1]	---	keine gültige Adresse
3008	INVALID NUMERIC OPERAND	## [1]	---	ungültiger numerischer Operand
3009	UNKNOWN LABEL	## [1]	---	Das Label, zu dem gesprungen werden soll, existiert nicht.
3010	INVALID LOCATION	## [1]	---	ungültige Angabe des Speicherortes (siehe "Gemeinsame Elemente" auf Seite 72 Tabelle Nr. 15)
3011	SIZE NOT SUPPORTED	## [1]	---	ungültige Angabe des Typs beim Lesen bzw. Schreiben auf einen Speicherort (siehe "Gemeinsame Elemente" auf Seite 72 und "Erweiterungen" auf Seite 78, jeweils Tabelle Nr. 15)
3012	NOT ABLE TO WRITE TO OPERAND	## [1]	---	Es wird versucht, auf einen read-only Operanden zu schreiben.
3013	MAX NUMBER OF LABELS REACHED	## [1]	---	Die Anzahl der definierten Labels überschreitet die maximal zulässige Anzahl für diese Geräteklasse.
3014	LABEL ALREADY DEFINED	## [1]	---	Es wurde bereits ein Label mit exakt derselben Bezeichnung in den vorherigen Script-Zeilen definiert.
3015	MAX INSTRUCTIONS PER RUN EXCEEDED	## [1]	---	Die Anzahl der Befehle des Scripts überschreitet die maximal zulässige Anzahl für diese Geräteklasse.
3016	UNKNOWN TYPE	## [1]	---	Datentyp des Operanden unbekannt
3017	INVALID TIME OPERAND	## [1]	---	ungültige Angabe eines Operanden vom Typ „Zeitdauer“
3018	INVALID DATE AND TIME OPERAND	## [1]	---	ungültige Angabe eines Operanden vom Typ „Datum und Uhrzeit“

[1] Nummer der Script-Zeile in der der Fehler aufgetreten ist

9.4 Erweiterungen

Der Standard wurde durch folgende Ergänzungen erweitert:

Nr. der Tabelle	Beschreibung	Beispiel/Erklärung
15	R (Reelle Zahl, (32 Bits) Größe)	%IR0 %QR2 %MR1000
	T (Zeitdauer - Größe)	LD T#1d23h45m ST %MT0
	S (Datum und Uhrzeit - Größe)	LD DT#2010-08-25-15:36:55 ST %MS0
52 (1/2)	DOW Day of Week	ermittelt den Wochentag aus einer Variablen vom Typ "Datum und Uhrzeit" (* 0 = Monday *) DOW DT#2010-08-25-15:36:55
	LDS [1)] Load Status	(* Status vom %IR1 laden*) LDS %IR1
	STS [1)] Store Status	(* Status von %QR1 auf 2 setzen *) LD 2 STS %QR1
	POW Power of	Mit dem Operator "POW" kann eine Variable potenziert werden. LD %IR0 POW 2.0

Nr. der Tabelle	Beschreibung	Beispiel/Erklärung
52 (2/2)	INC Increment	erhöht eine Variable LD 5 INC
	DEC Dencrement	reduziert eine Variable LD 5 DEC
	JGTZ	Sprung zur Marke, wenn der Wert im Accu › 0 JGTZ {label} LD 5 JGTZ label1 ADD 3 label1: SUB 2
	JGEZ	Sprung zur Marke, wenn der Wert im Accu >= 0
	JZ	Sprung zur Marke, wenn der Wert im Accu = 0
	JNZ	Sprung zur Marke, wenn der Wert im Accu ‹› 0
	JLEZ	Sprung zur Marke, wenn der Wert im Accu ‹= 0
	JLTZ	Sprung zur Marke, wenn der Wert im Accu ‹ 0
	CEIL	Wert im Accu aufrunden
	FLOOR	Wert im Accu abrunden
	ROUND	Wert im Accu kaufmännisch runden

[1)]Das rapidM2M Developer Paket M11 verwendet eine spezielle Kodierung, um verschiedene Fehlerzustände in den Messkanälen anzuzeigen (z.B. OL = open loop, SC = short circuit). Dieser in den normalen Messwert kodierte Fehlercode darf nicht für Berechnungen verwendet werden. Mit den beiden Operatoren „LDS“ und „STS“ kann dieser Fehlercode gelesen bzw. geschrieben werden. Der Operator "LDS" liefert immer ein Ergebnis vom Typ "UINT". Der Operator "STS" erwartet einen Parameter vom Typ "UINT".

Mögliche Fehlercodes

Status	Beschreibung
0	Wert gültig
1	Undefinierter Fehler
2	Overflow
3	Underflow
4	Open Loop (Unterbrechung)
5	Short Circuit (Kurzschluss)

9.5 Geräteabhängige Implementierung

Maximal zulässige Anzahl der Scripts-Befehle	100
Maximale Zeichenanzahl für ein Syntaxelement	25
Maximale Zeichenanzahl für eine Script-Zeile	100
Maximal zulässige Anzahl an Labels	20
Adressen der Eingänge	%Ix0 .. %Ix64 • %Ix0: Interner Messkanal "GSM Stärke" • %Ix1...%Ix64 oder %Ix1000...%Ix1063: Interface Kanäle 1-64
Adressen der Ausgänge	%Qx1000 .. %Qx1063 • %Qx1000...%Qx1063: Ausgabekanäle 1-64
Adressen der Memory Register	%Mx0 .. %Mx31

Adressen der globalen Register	%Mx1000 .. %Mx1001
	• %Mx1000: Globale Trigger Maske
	  • Bit0: Warnung
	  • Bit1: Alarm
	  • Bit2: Störung-Warnung
	  • Bit3: Störung-Alarm
	  • Bit4-7: reserviert
	  • Bit8: schnelle Aufzeichnung
	  • Bit9: langsame Aufzeichnung
	  • Bit10: reserviert
	  • Bit11: Aufzeichnung einschalten
	  • Bit12: Aufzeichnung ausschalten
	  • Bit13: Übertragung auslösen
	  • Bit14: 0...Trigger wird bei "größer gleich" dem Schwellwert ausgelöst, 1...Trigger wird bei "kleiner gleich" dem Schwellwert ausgelöst.
	  • Bit15: Online-Modus aktivieren
	  • Bit16: Aufwärmzeit aktiv
	  • Bit17: Messung aktiv
	  • Bit18-19: reserviert
	• %Mx1001: aktueller Zeitstempel

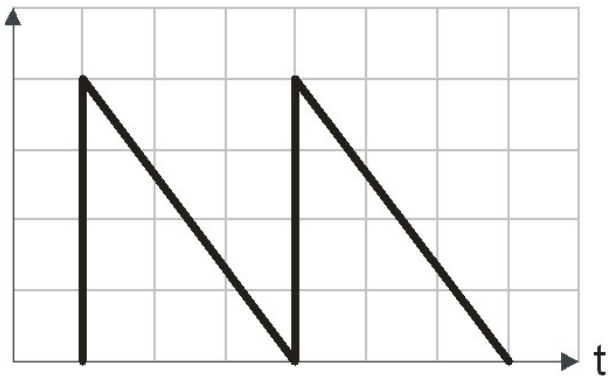
Adressen der Register, die für den jeweiligen Eingang angeben, welche Trigger aktiv oder inaktiv sind.	%Mx2000 .. %Mx2064 • Bit0-7: reserviert • Bit8: schnelle Aufzeichnung • Bit9: langsame Aufzeichnung • Bit10: reserviert • Bit11: Aufzeichnung einschalten • Bit12: Aufzeichnung ausschalten • Bit13: Übertragung auslösen • Bit14: 0...Trigger wird bei "größer gleich" dem Schwellwert ausgelöst, 1...Trigger wird bei "kleiner gleich" dem Schwellwert ausgelöst. • Bit15: Online-Modus aktivieren
Adressen der Register, die für den jeweiligen Eingang angeben, welche Alarme aktiv oder inaktiv sind.	%Mx3000 .. %Mx3064 • Bit0: Warnung • Bit1: Alarm • Bit2: Störung-Warnung • Bit3: Störung-Alarm
Adressen der Display Register	%Mx4000 .. %Mx4007

9.6 Beispiele

9.6.1 Sägezahngenerator

Dieses Beispiel erzeugt einen Sägezahn am Ausgang eines Geräts.

```
LD %MR0          (* Lädt den Wert des Memory Registers 0 *)
LE 0.0           (* Prüft auf <= 0.0 *)
JMPCN ende       (* Wenn die Prüfung der vorhergehenden Zeile nicht 0
                 ergibt, zum Label "ende" springen *)
LD 4.0           (* Den Wert 4 in den Accu laden *)
ST %MR0          (* Den Wert 4 vom Accu in das Memory Register 0 speichern *)
ende:            (* Label "ende" *)
LD %MR0          (* Lädt den Wert des Memory Registers 0 *)
SUB 0.1          (* 0.1 vom Wert im Accu subtrahieren *)
ST %QR0          (* Den Wert vom Accu auf den Ausgang 0 schreiben *)
ST %MR0          (* Den Wert vom Accu ins Memory Register 0 scheiben *)
```

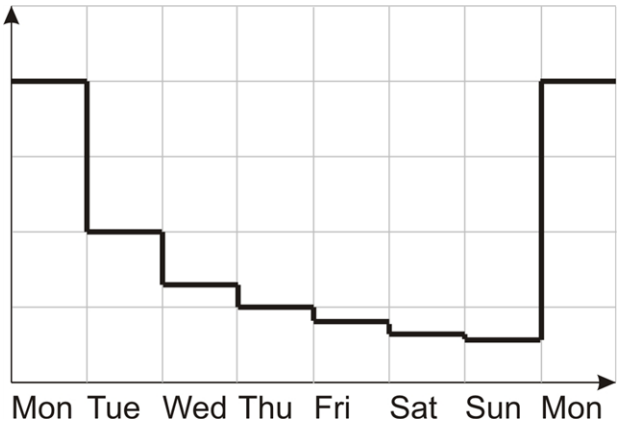


Mittels IL erzeugter Sägezahn

9.6.2 Wochentagbestimmung

Dieses Beispiel ermittelt den Wochentag aus dem aktuellen Zeitstempel eines Geräts. Danach wird ein Zwischenwert berechnet, indem die Zeitdifferenz zwischen den Durchläufen des Scripts durch 6 dividiert wird. Bei 2min Messintervall ergibt sich so ein Wert von 20. Dieser Zwischenwert wird dann noch durch den ermittelten Wochentag dividiert und das Ergebnis dann am Ausgang 0 ausgegeben.

```
LD %MS0                 (* Lädt den Zeitstempel aus dem Memory Register 0 *)
EQ  DT#0-00-0-0:0:0     (* Prüft auf = 0-00-0-0:0:0 (Noch kein Zeitstempel vorhanden) *)
JMPC END                (* Wenn die Prüfung der vorhergehenden Zeile 1
                        ergibt, zum Label "END" springen *)
DOW %MS1001             (* Wochentag aus dem akt. Zeitstempel ermitteln *)
ADD 1                   (* +1 damit bei der späteren Division nicht durch 0 (Montag)
                        dividiert wird*)
ST %MD2                 (* Ermittelten Wochentag im Memory Register 2 speichern)
LD %MS1001              (* Akt. Zeitstempel in den Accu laden *)
SUB %MS0                (* Zeitstempel, der im Memory Register 0 gespeichert ist,
                        vom Accu abziehn *)
DIV T#6s                (* Division durch 6 *)
ST %MT1                 (* Typenkonvertierung von "Zeitdauer" auf "reelle Zahl" durch
LD %MR1                 Speichern in und Laden aus Memory Register 1*)
DIV %MR2                (* Durch den Wochentag dividieren *)
ST %QR0                 (* Den Wert vom Accu auf den Ausgang 0 schreiben *)
END:                    (* Label "END" *)
LD %MS1001              (* Akt. Zeitstempel laden *)
ST %MS0                 (* Akt. Zeitstempel ins Memory Register 0 schreiben *)
```



Mittels IL erzeugtes Ausgangssignal

Kapitel 10 Pawn Script

10.1 Allgemein

Das folgende Kapitel beschreibt die Funktionalität des Pawn Scripts.

Zusätzliche Informationen finden Sie auf der Homepage der Entwickler:
http://www.compuphase.com/pawn/pawn.htm

Es gibt zwei Möglichkeiten, um ein Pawn Script für das rapidM2M Developer Paket M11 zu erstellen:

- Direkte Eingabe in das Eingabefenster „Script“ im Konfigurationsabschnitt „Steuerung“
- Verwenden eines Script-Templates, das am myDataweb–Server angelegt wurde

10.1.1 Direkte Eingabe eines Pawn Scripts

Die Eingabe des Pawn Scripts erfolgt über den Konfigurationsabschnitt „Steuerung“ (siehe "Steuerung" auf Seite 56) der Eingabemaske zur Konfiguration der Messstelle. Als „Script Type“ muss „Pawn“ ausgewählt werden, damit das rapidM2M Developer Paket M11 die unter „Script“ eingegebenen Befehle als Pawn Script interpretiert.

10.1.2 Verwendung eines Script-Templates

Über die Listenauswahl „Script Vorlage“ im Konfigurationsabschnitt „Steuerung“ (siehe "Steuerung" auf Seite 56) der Eingabemaske zur Konfiguration der Messstelle können vordefinierte Script-Templates ausgewählt werden (zum Anlegen eines Templates siehe "Benutzerhandbuch für myDataweb-Server" 206.886). Nach der Auswahl des Steuerprogramms wird das Template in das Eingabefenster „Script“ kopiert und kann dort angepasst werden. Als „Script Type“ muss auch bei dieser Methode „Pawn“ ausgewählt werden, damit das rapidM2M Developer Paket M11 die Befehle als Pawn Script interpretiert.

> ***Hinweis:*** *Wird mittels Pawn Script direkt auf einen Ausgang geschrieben, überschreibt der ermittelte Wert den über die Eingabemaske am myDataweb-Server eingegebenen Stellwert.*

10.2 Allgemeine Syntax

10.2.1 Format

Bezeichner, Zahlen und Zeichen werden durch Leerzeichen, Tabulatoren, Zeilenumbrüche und "Form Feed" getrennt. Eine Serie von einer oder mehreren dieser Separatoren wird als Leerraum erkannt.

10.2.2 Optionale Semikolons

Semikolons (um ein Statement zu beenden) sind optional, wenn sie am Ende einer Zeile auftreten. Semikolons sind notwendig, um mehrere Statements in einer Zeile zu trennen. Ein Ausdruck kann auf mehrere Zeilen aufgeteilt werden, jedoch müssen Postfix-Operatoren in derselben Zeile wie der Operand stehen.

10.2.3 Kommentare

Text zwischen den Symbolen /* und */ (beide Symbole können auf derselben oder auf unterschiedlichen Zeilen stehen) und Text nach // (bis zum Ende einer Zeile) sind Kommentare. Kommentare dürfen nicht verschachtelt werden. Der Compiler betrachtet Kommentare als Leerzeichen. Ein Kommentar, der mit "/** " (zwei Sterne und ein Leerzeichen nach dem zweiten Stern) beginnt und mit einem "*/" endet, ist ein Dokumentationskommentar. Ein Kommentar, der mit "/// " (drei Schrägstriche und ein Leerzeichen nach dem dritten Schrägstrich) beginnt, ist ebenfalls ein Dokumentationskommentar. Der Parser kann den Dokumentationskommentar in unterschiedlicher Weise unterstützen, zum Beispiel könnte er eine Online-Hilfe daraus generieren.

10.2.4 Bezeichner

Namen von Variablen, Funktionen und Konstanten. Bezeichner bestehen aus den Zeichen a...z, A...Z, 0...9, _ oder @. Das erste Zeichen darf keine Ziffer sein. Die Zeichen @ und _ alleine sind keine gültigen Bezeichner, z.B. "_Up" ist ein gültiger Bezeichner, aber "_" ist es nicht. Pawn unterscheidet zwischen Groß- und Kleinschreibung. Der Parser schneidet Bezeichner ab einer bestimmten Länge ab. Es werden standardmäßig nur die ersten 16 Zeichen für die Unterscheidung herangezogen.

10.2.5 Reservierte Schlüsselworte

Statements	Operator	Direktiven	Andere
assert	defined	defined	defined
break	sizeof	sizeof	sizeof
case	state	state	state
continue	tagof	tagof	tagof
default			
do			
else			
exit			
for			
goto			
if			
return			
sleep			
state			
switch			
while			

10.2.6 Numerische Konstanten

10.2.6.1 Numerische Integer Konstanten

Binär
ob gefolgt von einer Serie von 0 und 1

Dezimal:
eine Serie von Ziffern zwischen 0 und 9

Hexadezimal
0x gefolgt von einer Serie von Ziffern zwischen 0 und 9 und den Buchstaben a bis f

10.2.6.2 Numerische Gleitkomma-Konstanten

Eine Gleitkommazahl ist eine Zahl mit einem Nachkommateil. Eine Gleitkommazahl beginnt mit einer oder mehreren Ziffern, beinhaltet einen Dezimaltrennpunkt und hat zumindest eine Ziffer nach dem Dezimaltrennpunkt. z.B. "12.0" und "0.75" sind gültige Gleitkommazahlen. Optional kann noch ein Exponent angehängt werden. Die Notation ist der Buchstabe "e" (Kleinbuchstabe), gefolgt von einer ganzzahligen numerischen Konstante. Z.B. "3.12e4" oder "12.3e-3" sind gültige Gleitkommazahlen mit Exponent.

10.3 Variablen

10.3.1 Deklaration

Das Schlüsselwort "new" deklariert eine neue Variable. Für spezielle Deklarationen wird das Schlüsselwort "new" durch "static" ersetzt (siehe "Statische lokale Deklaration" auf Seite 88). Sofern sie nicht explizit initialisiert wird, ist der Wert der neuen Variablen Null.

Eine Variablendeklaration kann auftreten

- an jeder Position, an der ein Ausdruck gültig ist - lokale Variable
- an jeder Position, an der eine Funktionsdeklaration oder eine Implementation der Funktion gültig ist - globale Variablen;
- im ersten Ausdruck einer "for" Schleife (siehe "for (Ausdruck 1 ; Ausdruck 2 ; Ausdruck 3) Statement " auf Seite 100) - lokale Variable

Beispiel:

```
new a;     // ohne Initialisierung (Wert ist 0)
new b = 3; // mit Initialisierung (Wert ist 3)
```

10.3.2 Lokale Deklaration

Eine lokale Deklaration erscheint innerhalb eines Anweisungs-Blocks. Auf eine Variable kann nur innerhalb dieses Blocks und der darin enthaltenen Blöcke zugegriffen werden. Eine Deklaration innerhalb des ersten Ausdrucks einer Schleifenanweisung ist ebenfalls eine lokale Deklaration.

10.3.3 Globale Deklaration

Eine globale Deklaration erscheint außerhalb einer Funktion und eine globale Variable kann in jeder Funktion verwendet werden. Globale Variablen können nur mit konstanten Ausdrücken initialisiert werden.

10.3.4 Statische lokale Deklaration

Eine lokale Variable wird zerstört, wenn die Ausführung den Block verlässt, in dem die Variable geschaffen wurde. Lokalen Variablen in einer Funktion existieren nur während der Laufzeit der genannten Funktion. Jede neuer Aufruf der Funktion erstellt und initialisiert neue lokale Variablen. Wenn eine lokale Variable mit dem Schlüsselwort "static" anstatt "new" deklariert ist, bleibt die Variable auch nach dem Ende einer Funktion im Speicher. Dies bedeutet, dass statische lokale Variablen eine private, dauerhafte Speicherung bereitstellen, die nur in einer einzigen Funktion (oder einem Block) zugänglich sind. Wie globale Variablen, können statische lokale Variablen nur mit konstanten Ausdrücken initialisiert werden.

10.3.5 Statische globale Deklaration

Eine statische globale Variable verhält sich wie eine globale Variable, mit dem Unterschied, dass die Variable nur in der Datei gültig ist, in der sie deklariert wurde. Um eine globale Variable statisch zu deklarieren, ersetzen Sie das Schlüsselwort "new" mit "static".

10.3.6 Gleitkommawerte

Pawn unterstützt Gleitkommawerte. Diese können an jeder Stelle eingesetzt werden, an der eine Variablendeklaration gültig ist.

Beispiel:

```
new Float:a;       // ohne Initialisierung (Wert ist 0.0)
new Float:b = 3.0; // mit Initialisierung (Wert ist 3.0)
```

10.4 Konstante Variablen

Es ist manchmal notwendig eine Variable zu erstellen, die einmal initialisiert wird und dann nicht mehr verändert werden soll. Eine solche Variable verhält sich ähnlich wie eine symbolische Konstante, aber sie ist dennoch eine Variable. Um eine konstante Variable zu deklarieren, legen Sie das Schlüsselwort "const" zwischen das Schlüsselwort, das die Variablendeklaration ("new", "static") startet und den Namen der Variablen.

Beispiel:

```
new const address[4] = { 192, 0, 168, 66 }
static const status /* initialized to zero */
```

Drei typische Situationen, in denen man eine konstante Variable nutzen könnte, sind:

- Um eine "array"-Konstante zu erstellen. Auf symbolische Konstanten kann nicht per Index zugegriffen werden.
- Ein besonderer Fall ist, wenn die Array-Argumente in einer Funktion als "const" markiert werden. Array-Argumente werden immer per Referenz übergeben. Wenn sie als "const" deklariert werden, schützt sie das vor ungewollten Änderungen. Siehe Beispiele von "const-Funktionsargumenten" im Kapitel "Funktionsargumente ("call-by-value" versus "call-by-reference")" auf Seite 104.

10.5 Array Variablen

10.5.1 Eindimensionales Array

Die Syntax name[constant] deklariert "name" als ein Array aus “constant” Elementen, wobei jedes Element ein Eintrag ist. "name" ist ein Platzhalter für den Namen der Variable und "constant" ist ein positiver Wert ungleich Null. "constant" ist optional und kann weggelassen werden. Wenn kein Wert zwischen den Klammern steht, ist die Anzahl von Elementen gleich der Anzahl der Initialwerte. Der Array-Index-Bereich ist "Null-basierend", das bedeutet, dass das erste Element "name[0]" und das letzte Element "name[constant-1]" ist.

10.5.2 Initialisierung

Datenobjekte können bei ihrer Deklaration initialisiert werden. Der initialisierte Wert von globalen Datenobjekten muss ein konstanter Wert sein. Arrays, global oder lokal, müssen ebenfalls mit konstanten Werten initialisiert werden. Nicht initialisierte Daten sind standardmäßig Null.

Beispiele:

Auflistung: gültige Deklaration

```
new i = 1
new j                           /* j ist 0 */
new k = 'a'                     /* k hat den Zeichencode von 'a' */
new a[] = [1,4,9,16,25]         /* a hat 5 Elemente */
new s1[20] = ['a','b']          /* die restlichen 18 Elemente sind 0 */
new s2[] = ''Hello world...''   /* ein unpacked string */
```

Auflistung: gültige Deklaration

```
new c[3] = 4                    /* Ein Array kann nicht auf einen einzelnen
                                   Wert gesetzt werden */
new i = "Good-bye"              /* Nur ein Array kann einen String halten. */
new q[]                         /* Unbekannte Größe für ein Array */
new p[2] = { i + j, k - 3 }     /* Arrayinitialisierer müssen Konstanten sein. */
```

10.5.3 Progressive Initialisierung für Arrays

Der Punkte-Operator führt die Initialisierung des Arrays aufgrund der letzten beiden initialisierten Werte weiter. Der Punkte-Operator (drei Punkte, "...") initialisiert das Array bis zur Arraygrenze.

Beispiel: Auflistung: Arrayinitialisierer

```
new a[10] = { 1, ... }              // setzt alle Elemente auf 1
new b[10] = { 1, 2, ... }           // b = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
new c[8]  = { 1, 2, 40, 50, ... } // c = 1, 2, 40, 50, 60, 70, 80, 90
new d[10] = { 10, 9, ... }          // d = 10, 9, 8, 7, 6, 5, 4, 3, 2, 1
```

10.5.4 Mehrdimensionale Arrays

(Es werden nur Arrays mit bis zu 3 Dimensionen unterstützt)

Mehrdimensionale Arrays sind Arrays, die Referenzen zu weiteren Sub-Arrays enthalten. Zum Beispiel ist ein zweidimensionales Array ein "Array auf Eindimensionale Arrays".

Beispiele für die Deklaration von zweidimensionalen Arrays:

```
new a[4][3]
new b[3][2] = [ [ 1, 2 ], [ 3, 4 ], [ 5, 6 ] ]
new c[3][3] = [ [ 1 ], [ 2, ...], [ 3, 4, ... ] ]
new d[2]{10} = [ "agreement", "dispute" ]
new e[2][] = [ ''OK'', ''Cancel'' ]
new f[][] = [ ''OK'', ''Cancel'' ]
```

Wie die beiden letzten Deklarationen (Variablen "e" und "f") zeigen, hat die letzte Dimension eine nicht spezifizierte Länge. In diesem Fall wird die Länge des Sub-Arrays aus dem dazugehörigen Initialisierer erkannt. Jedes Sub-Array hat eine unterschiedliche Länge. In diesem speziellen Beispiel enthält "e[1][5]" den Buchstaben "l" des Wortes "Cancel", aber "e[0][5]" ist ungültig, da das Sub-Array e[0] nur drei Einträge (die Buchstaben "O", "K", und den Null-Terminator) beinhaltet. Der Unterschied zwischen den Deklarationen der Arrays "e" und "f" ist, dass wir bei "f" den Compiler die Anzahl der höheren Dimension ermitteln lassen. "sizeof f" ist 2 genauso wie "sizeof e" (siehe "Arrays und der "sizeof"-Operator" auf Seite 90).

10.5.5 Arrays und der "sizeof"-Operator

Der "sizeof"-Operator gibt die Anzahl der Elemente einer Variable zurück. Für eine einfache (nicht Array) Variable ist das Ergebnis von "sizeof" immer 1.

Ein Array mit einer Dimension enthält eine Anzahl von Elementen und der "sizeof"-Operator gibt diese Anzahl zurück. Der Codeausschnitt unterhalb würde deshalb "5" ausgeben, da das Array 4 Zeichen und den Null-Terminator enthält.

```
new msg[] = ''Help''
printf(''%d'', sizeof msg);
```

Der "sizeof"-Operator gibt immer die Anzahl der Einträge, auch für ein "packed" Array, zurück. Der Codeausschnitt unterhalb gibt auch "5" aus, da die Variable 5 Einträge enthält, auch wenn diese im Speicher weniger Platz benötigt.

```
new msg{} = "Help"
printf(''%d'', sizeof msg);
```

Bei mehrdimensionalen Arrays kann der "sizeof"-Operator die Anzahl der Elemente jeder Dimension zurückgeben. Für die letzte (niedrigste) Dimension ist ein Element ein einzelner Eintrag, jedoch für die höchste Dimension ist es ein Sub-Array. Beachten Sie, dass im nachfolgenden Codeausschnitt die Syntax "sizeof matrix" die Anzahl der Elemnete der höhere Dimension zurückgibt, und dass die Syntax "sizeof matrix[]" die niedrigere Dimension des zweidimensionalen Arrays ausgibt. Der Codeausschnitt gibt 3 (höhere Dimension) und 2 (niedrigere Dimension) aus.

```
new matrix[3][2] = { { 1, 2 }, { 3, 4 }, { 5, 6 } }
printf(''%d %d'', sizeof matrix, sizeof matrix[]);
```

Die Anwendung des "sizeof"-Operators auf mehrdimensionale Arrays ist besonders praktisch, wenn er als Standardwert für Funktionsargumente verwendet wird.

10.6 Operatoren und Ausdrücke

10.6.1 Zeichenerklärung

Die Anwendung von einigen Operatoren hängt von der jeweiligen Art des Operanden ab. Aus diesem Grund wird in diesem Kapitel folgende Notation angewendet:

e
: *beliebiger Ausdruck (eng. expression)*

v
: *beliebiger Ausdruck, dem ein Wert zugewiesen werden kann (“lvalue” Ausdruck - Variable)*

a
: *ein Array*

f
: *eine Funktion*

s
: *ein Symbol - dies kann eine Variable, eine Konstante oder eine Funktion sein*

10.6.2 Ausdrücke

Ein Ausdruck besteht aus ein oder mehreren Operanten mit einem Operator. Der Operand kann eine Variable, eine Konstante oder ein anderer Ausdruck sein. Ein Ausdruck gefolgt von einem Semikolon ist ein Statement.

Beispiele für Ausdrücke:

```
v++ f(a1, a2)
v = (ia1 * ia2) / ia3
```

10.6.3 Arithmetik

Operator	Beispiel	Erklärung
+	e1 + e2	Ergebnis der Addition von e1 und e2
-	e1 - e2	Ergebnis der Subtraktion von e1 und e2
	-e	Ergebnis der arithmetischen Negation von e (Zweierkomplement)
*	e1 * e2	Ergebnis der Multiplikation von e1 und e2
/	e1 / e2	Ergebnis der Division e1 durch e2. Das Ergebnis wird zum nächstgelegenen ganzzahligen Wert, der kleiner oder gleich dem Quotienten ist, abgeschnitten. Sowohl positive als auch negative Werte werden abgerundet (Richtung unendlich).
%	e1 % e2	Ergebnis ist der Rest der Division e1 durch e2. Das Vorzeichen ist dasselbe wie bei e2
++	v++	erhöht v um 1. Das Ergebnis des Ausdrucks ist der Wert vor der Erhöhung.
	++v	erhöht v um 1. Das Ergebnis des Ausdrucks ist der Wert nach der Erhöhung.
--	v--	verringert v um 1. Das Ergebnis des Ausdrucks ist der Wert vor der Verringerung.
	--v	verringert v um 1. Das Ergebnis des Ausdrucks ist der Wert nach der Verringerung.

Hinweis: *Das unäre + ist in Pawn nicht definiert. Die Operatoren ++ und -- ändern den Operanden. Der Operand muss ein "lvalue" sein.*

10.6.4 Bit-Manipulation

Operator	Beispiel	Erklärung
~	~e	Ergebnis ist das Einerkomplement von e.
››	e1 ›› e2	Ergebnis der arithmetischen Verschiebung nach rechts von e1 durch e2 Bits. Die Verschiebung ist vorzeichenbehaftet: Das Bit ganz links wird auf die freien Bits des Ergebnisses kopiert.
›››	e1 ››› e2	Ergebnis der logischen Verschiebung nach rechts von e1 durch e2 Bits. Die Verschiebung ist vorzeichenlos. Die freien Bits des Ergebnisses werden mit 0 aufgefüllt.

Operator	Beispiel	Erklärung
<<	e1 << e2	Ergebnis: Verschiebung nach links von e1 durch e2 Bits. Die freien Bits des Ergebnisses werden mit 0 aufgefüllt. Es gibt keinen Unterschied zwischen einer arithmetischen und einer logischen Verschiebung nach links.
&	e1 & e2	Ergebnis ist das bitweise logische "und" von e1 und e2.
|	e1 | e2	Ergebnis ist das bitweise logische "oder" von e1 und e2.
^	e1 ^ e2	Ergebnis ist das bitweise "exklusiv oder" von e1 und e2.

10.6.5 Zuweisung

Das Ergebnis eines Zuweisungsausdrucks ist der Wert des Operanden nach der Zuweisung.

Operator	Beispiel	Erklärung
=	v = e	weist den Wert von e der Variable v zu
	v = a	weist das Array a der Variable v zu. v muss ein Array mit derselben Größe und denselben Dimensionen sein wie a. a kann eine Zeichenkette oder ein Array sein.

Hinweis: Die folgenden Operatoren kombinieren eine Zuweisung mit einer arithmetischen oder bitweisen Operation. Das Ergebnis des Ausdrucks ist der Wert des linken Operanten nach der arithmetischen oder bitweisen Operation.

Operator	Beispiel	Erklärung
+=	v += e	erhöht v um e
-=	v -= e	vermindert v um e
*=	v *= e	multipliziert v mit e
/=	v /= e	dividiert v mit e
%=	v %= e	weist v den Rest der Division von v und e zu
>>=	v >>= e	verschiebt v arithmetisch um e Bits nach rechts
>>>=	v >>>= e	verschiebt v logisch um e Bits nach rechts
<<=	v <<= e	verschiebt v um e Bits nach links
&=	v &= e	führt ein bitweises "und" von v und e aus und weist das Ergebnis v zu
|=	v |= e	führt ein bitweises "oder" von v und e aus und weist das Ergebnis v zu
^=	v ^= e	führt ein bitweises "exklusiv oder" von v und e aus und weist das Ergebnis v zu

10.6.6 Vergleichsoperatoren

Ein logisches "false" wird durch einen Integer-Wert von 0 repräsentiert; ein logisches "true" durch einen Wert, der nicht 0 ist. Ergebnisse eines Vergleichs-Ausdrucks sind entweder 0 oder 1 und ihr "tag" wird auf "bool" gesetzt.

Operator	Beispiel	Erklärung
==	e1 == e2	Das Ergebnis ist "true", wenn e1 und e2 gleich sind.
!=	e1 != e2	Das Ergebnis ist "true", wenn e1 nicht gleich e2 ist.

> ***Hinweis:*** *Die folgenden Operatoren können verkettet werden, wie im Ausdruck "e1 <= e2 <= e3". Dies bedeutet, dass das Ergebnis "1" ist, wenn jeder einzelne Vergleich zutrifft und "0", wenn zumindest ein Vergleich nicht zutrifft.*

Operator	Beispiel	Erklärung
<	e1 < e2	Das Ergebnis ist ein logisches "true", wenn e1 kleiner ist als e2.
<=	e1 <= e2	Das Ergebnis ist ein logisches "true", wenn e1 kleiner oder gleich e2 ist.
>	e1 > e2	Das Ergebnis ist ein logisches "true", wenn e1 größer ist als e2.
>=	e1 >= e2	Das Ergebnis ist ein logisches "true", wenn e1 größer oder gleich e2 ist.

10.6.7 Boolean

Ein logisches "false" wird durch einen Integer-Wert von 0 repräsentiert, ein logisches "true" durch einen Wert, der nicht 0 ist. Ergebnisse eines Vergleichs-Ausdrucks sind entweder 0 oder 1 und ihr "tag" wird auf "bool" gesetzt.

Operator	Beispiel	Erklärung
!	!e	Das Ergebnis ist eine logisches "true", wenn e logisch "false" ist.
||	e1 || e2	Das Ergebnis ist "true", wenn entweder e1 oder e2 (oder beide) logisch "true" sind. Der Ausdruck e2 wird nur ausgewertet, wenn e1 logisch "false" ist.
&&	e1 && e2	Das Ergebnis ist "true", wenn e1 und e2 logisch "true" sind. Der Ausdruck e2 wird nur ausgewertet, wenn e1 logisch "true" ist.

10.6.8 Sonstiges

Operator	Beispiel	Erklärung
[ ]	a[e]	Array Index: Das Ergebnis ist der Eintrag an der Position e des Arrays a.
{ }	a{e}	Array Index: Das Ergebnis ist das Zeichen an der Position e des "packed" Arrays a.
( )	f(e1, e2, ... eN)	Das Ergebnis ist der Wert, der von der Funktion f zurückgegeben wird. Die Funktion wird mit den Parametern e1, e2, ... eN aufgerufen. Die Reihenfolge der Auswertung der Parameter ist nicht definiert. (Die Implementation der Script-Engine wertet die Parameter möglicherweise in umgekehrter Reihenfolge aus.)
? :	e1 ? e2 : e3	Das Ergebnis ist entweder e2 oder e3, abhängig vom Wert von e1. Der bedingte Ausdruck ist ein zusammengesetzter Ausdruck mit einem zweiteiligen Operator, "?" und ":". Der Ausdruck e2 wird ausgewertet, wenn e1 logisch "true" ist, e3 wird ausgewertet, wenn e1 logisch "false" ist.
:	tagname: e	"tag" Überschreibung: Der Wert des Ausdrucks ändert sich nicht, jedoch ändert sich der "tag".
defined	defined s	Ergebnis ist "1", wenn das Symbol definiert wurde. Das Symbol kann eine Konstante oder eine globale oder lokale Variable sein. Der "tag" des Ausdrucks ist "bool".
sizeof	sizeof s	Das Ergebnis ist die Anzahl der Elemente der angegeben Variable. Für einfache Variablen und für eindimensionale Arrays ist ein Element ein Eintrag. Für mehrdimensionale Arrays ist das Ergebnis die Anzahl der Elemente (Sub Arrays) in der höchsten Dimension. Fügen Sie [] zum Namen des Arrays hinzu, um eine niedrigere Dimension anzugeben. Wenn die Größe der Variable nicht bekannt ist, dann ist das Ergebnis 0. Wenn dieser Operator in einem "default"-Wert einer Funktion verwendet wird, dann wird der Ausdruck zum Zeitpunkt des Aufrufs der Funktion und nicht zum Zeitpunkt der Definition ausgeführt.
tagof	tagof s	Das Ergebnis ist eine eindeutige Zahl, die den "tag" der Variablen, der Konstanten, des Rückgabewerts einer Funktion oder des Names der "tag"-Bezeichnung repräsentiert. Wenn dieser Operator in einem "default"-Wert einer Funktion verwendet wird, dann wird der Ausdruck zum Zeitpunkt des Aufrufs der Funktion und nicht zum Zeitpunkt der Definition ausgeführt.

10.6.9 Priorität der Operatoren

Die nachfolgende Tabelle gruppiert Operatoren mit derselben Priorität, beginnend mit der höchsten Priorität.

Wenn die Auswertung eines Ausdrucks nicht explizit durch Klammern begründet wird, wird sie von den Assoziationsregeln bestimmt. Zum Beispiel: a*b/c ist gleich (a*b)/c auf Grund der links zu rechts Assoziation, und a=b=c ist gleichzusetzen mit a=(b=c).

Operator	Erklärung	Lesefolge
( ) [ ] { }	Funktionsaufruf Array Index (Element) Array Index (Zeichen)	links-nach-rechts
! ~ - ++ -- : defined sizeof tagof	logisches Nicht Einerkomplement Zweierkomplement (unäres Minus) Erhöhung Verringerung "tag"-Überschreibung Symbol Definitions-Status Symbol Größe in "Elementen" eindeutige Zahl des "tags"	rechts-nach-links
* / %	Multiplikation Division Modulo	links-nach-rechts
+ -	Addition Subtraktion	links-nach-rechts
›› ››› ‹‹	arithemtische Verschiebung nach rechts logische Verschiebung nach rechts Verschiebung nach links	links-nach-rechts
&	bitweises "und"	links-nach-rechts
^	bitweises "exklusiv oder"	links-nach-rechts
|	bitweises "oder"	links-nach-rechts
‹ ‹= › ›=	kleiner als kleiner oder gleich als größer als größer oder gleich als	links-nach-rechts
== !=	gleich ungleich	links-nach-rechts
&&	logisches "und"	links-nach-rechts
||	logisches "oder"	links-nach-rechts
? :	bedingte Ausführung	rechts-nach-links
=	Zuweisung *= /= %= += -= ››= ›››= ‹‹= &= ^= |=	rechts-nach-links
,	Komma	links-nach-rechts

10.7 Anweisungen

Ein Statement kann aus einer oder mehreren Zeilen bestehen. Eine Zeile kann zwei oder mehrere Statements enthalten.

Statements zur Ablaufsteuerung (if, if-else, for, while, do-while und switch) können geschachtelt werden.

10.7.1 Statement-Etikett

Ein Etikett besteht aus einem Identifizierer gefolgt von einem ":". Ein Etikett ist ein "Sprung-Ziel" eines "goto" Statements.

Jede Anweisung kann mit einem Etikett versehen werden. Es muss dem Etikett ein Statement folgen, dies kann auch ein "leeres Statement" sein.

Der Gültigkeitsbereich eines Etiketts ist die Funktion in der es deklariert wurde d.h. ein "goto"-Statement kann nicht aus der aktuellen Funktion in eine andere Funktion springen.

10.7.2 Zusammengesetzte Anweisungen

Eine zusammengesetzte Anweisung (auch Block genannt) ist eine Serie von Null oder mehreren Anweisungen, welche durch Klammern ("{" und "}") umgeben ist. Die schließende Klammer ("}") darf nicht mit einem Semikolon abgeschlossen werden. Jede Anweisung kann durch einen Block ersetzt werden. Eine zusammengesetzte Anweisung, die keine Anweisungen enthält, ist ein Spezialfall und wird "leeres Statement" genannt.

10.7.3 Ausdrucksanweisung

Jeder Ausdruck wird zu einem Statement, wenn ein Semikolon (";") angehängt wird. Ein Ausdruck wird auch zu einem Statement, wenn dem Ausdruck bis zum Ende der Zeile nur Leerzeichen folgen und der Ausdruck nicht in der nächsten Zeile weitergeführt werden kann.

10.7.4 Leeres Statement

Eine leeres Statement führt keine Anweisungen aus und besteht aus einem Block-Statement ohne Anweisungen, d.h. es besteht aus dem Symbol "{ }". Leere Statements werden in Kontrollflussanweisungen ohne Aktionen eingesetzt (z.B. "while (!iskey()) {}") oder, wenn ein Etikett genau vor einer schließenden Klammer eines Block-Statements definiert wird. Ein leeres Statement endet nicht mit einem Semikolon.

10.7.5 assert Ausdruck

bricht das Programm mit einem Laufzeitfehler ab, wenn der Ausdruck logisch "false" ergibt

Hinweis: *Dieser Ausdruck schützt vor "unmöglich" oder ungültigen Bedingungen. Im folgenden Beispiel ist eine negative Fibonacci-Zahl ungültig. Die assert-Anweisung markiert diesen Fehler als Programmierer-Fehler. assert-Anweisungen sollten nur Programmierer-Fehler kennzeichnen und niemals Benutzereingaben.*

Beispiel:

```
fibonacci(n)
{
  assert n > 0

  new a = 0, b = 1
  for (new i = 2; i < n; i++)
  {
    new c = a + b
    a = b
    b = c
  }
  return a + b
}
```

10.7.6 break

beendet und verlässt das kleinste, umschließende "do"-, "for"- oder "while"-Statement an jedem beliebigen Punkt in der Schleife. Das "break"-Statement bewegt den Programmfluss zum nächsten Statement außerhalb der Schleife.

Beispiel:

```
example(n)
{
  new a = 0

  for(new i = 0; i < n ; i++ )
  {
    a += i

    if(i>10)
      break

    a += 1
  }
  return a
}
```

10.7.7 continue

beendet die aktuelle Iteration der kleinsten umschließenden "do"-, "for"- oder "while"-Anweisung und bewegt die Programmsteuerung an den Bedingungsteil der Schleife.

Beispiel

```
example(n)
{
  new a = 0

  for(new i = 0; i < n ; i++ )
  {
    a += i

    if(i>10)
      continue

    a += 1
  }
  return a
}
```

10.7.8 do Statement while (Ausdruck)

führt ein Statement aus, bevor der Bedingungsteil (die "while"-Bedingung) evaluiert wird. Das Statement wird wiederholt, solange die Bedingung logisch "true" ist. Das Statement wird zumindest einmal ausgeführt.

Beispiel:

```
example(n)
{
  new a = 0

  do
  {
    a++
  }
  while(n >= 0)

  return a
}
```

10.7.9 exit Ausdruck

bricht das Programm ab. Der Ausdruck ist optional, aber wenn er vorhanden ist, muss er in der selben Zeile wie das "exit"-Statement beginnen und enden. Die "exit"-Anweisung gibt den Wert des Ausdrucks zurück an die Hauptanwendung oder gibt Null zurück, wenn kein Ausdruck angegeben wird.

10.7.10 for (Ausdruck 1 ; Ausdruck 2 ; Ausdruck 3) Statement

Alle drei Ausdrucke sind optional.

Ausdruck 1:

wird nur einmal ausgewertet, vor Eintritt in die Schleife. Dieser Ausdruck kann zum Initialisieren einer Variablen genutzt werden. Dieser Ausdruck hält auch die Variablendeklaration mittels der "new"-Syntax. Eine Variable, die an dieser Stelle deklariert wird, ist nur innerhalb der Schleife gültig. Es ist nicht möglich einen Ausdruck (mit bereits vorhandenen Variablen) und eine Deklaration von neuen Variablen in diesem Feld zu kombinieren. Es müssen entweder alle Variablen in diesem Feld bereits vorhanden sein, oder sie müssen alle in diesem Bereich deklariert werden.

Ausdruck 2:

Dieser Ausdruck wird vor jedem Durchlauf der Schleife ausgeführt und beendet die Schleife, wenn der Ausdruck logisch "false" zurückgibt. Wenn dieser Ausdruck weggelassen wird, wird das Ergebnis des Ausdrucks 2 als logisch "true" angenommen.

Ausdruck 3:

Dieser Ausdruck wird nach jeder Ausführung des Statements ausgeführt. Die Programmsteuerung bewegt sich von Ausdruck 3 zum Ausdruck 2 für die nächste (bedingte) Iteration der Schleife.

Beispiel:

```
example(n)
{
  new a = 0

  for(new i = 0; i < n; i++)
  {
    a++
  }

  return a
}
```

Das Statement "for (; ;)" ist gleich dem Statement "while (true)".

10.7.11 goto Etikett

bewegt die Programmsteuerung (unbedingt) zu der Anweisung, die dem angegebenen Etikett folgt. Das Etikett muss innerhalb der gleichen Funktion wie die "goto"-Anweisung sein. Eine "goto"-Anweisung kann nicht aus einer Funktion springen.

10.7.12 if (Ausdruck) Statement 1 else Statement 2

führt das Statement 1 aus, wenn das Ergebnis des Ausdrucks logisch "true" ergibt. Die "else"-Klausel des "if"-Statements ist optional. Wenn das Ergebnis des Ausdruck logisch "false" ergibt und eine "else"-Klausel existiert, dann wird das Statement, das mit der "else"-Klausel assoziiert ist, (Statement 2) ausgeführt.

Beispiel:

```
example(n)
{
  if(n < 0)
    return -1
  else if (n == 0)
    return 0
  else
    return 1
}
```

10.7.13 return Ausdruck

beendet die aktuelle Funktion und bewegt die Programmsteuerung zum nächsten Statement nach dem Funktionsaufruf. Der Wert des Ausdrucks wird als Funktionsergebnis zurückgeliefert. Der Ausdruck kann ein Array oder eine Zeichenfolge sein. Der Ausdruck ist optional, wenn er jedoch vorhanden ist, muss er an der selben Zeile beginnen, wie das "return"-Statement. Wenn kein Ausdruck angegeben ist, dann wird Null zurückgeliefert.

10.7.14 switch (Ausdruck) { case Liste }

überträgt die Ablaufsteuerung an unterschiedliche Statements innerhalb des "switch" in Abhängigkeit vom Wert des "switch"-Ausdrucks. Der Hauptteil der "switch"-Anweisung ist eine zusammengesetzte Anweisung, die eine Reihe von "case"-Klauseln enthält. Jede "case"-Klausel beginnt mit dem Schlüsselwort "case", gefolgt von einer Liste von Konstanten und einem Statement. Die Liste der Konstanten ist eine Serie von Ausdrücken, getrennt durch Kommas, die jeweils zu einem konstanten Wert ausgewertet werden. Diese Liste endet mit einem Doppelpunkt. Um einen Bereich in dieser Liste anzugeben, trennen Sie die untere und obere Grenze des Bereichs mit einem doppelten Punkt (".."). Ein Beispiel für einen Bereich ist: "case 1..9:".

Das "switch"-Statement bewegt die Ablaufsteuerung zu einer "case"-Klausel, wenn ein Wert der Liste dem Wert des "switch"-Ausdrucks entspricht.

Die "default"-Klausel besteht aus dem Schlüsselwort "default" und einem Doppelpunkt. Die "default"-Klausel ist optional, aber wenn sie angegeben wird, muss sie als letzter Eintrag in der "case"-Liste eingetragen sein. Das "switch"-Statement bewegt die Ablaufsteuerung zur "default"-Klausel, wenn keine der "case"-Klauseln mit dem "switch"-Ausdruck übereinstimmt.

Beispiel:

```
example(n)
{
  new a = 0

  switch (n)
  {
    case 0..3:
      a = 0
    case 4,6,8,10:
      a = 1
    case 5,7:
      a = 2
    case 9:
      a = 3
    default:
      a = -1
  }

  return a
}
```

10.7.15 while (Ausdruck) Statement

wertet den Ausdruck aus und führt das Statement aus, wenn das Ergebnis des Ausdrucks logisch "true" ergibt. Nachdem die Anweisung ausgeführt wurde, kehrt die Programmsteuerung erneut zu dem Ausdruck zurück. Das Statement wird daher ausgeführt, solange der Ausdruck logisch "true" ist.

Beispiel:

```
example(n)
{
  new a = 0

  while(n >= 0)
  {
    a++
  }

  return a
}
```

10.8 Funktionen

Eine Funktionsdeklaration spezifiziert den Namen der Funktion und die formalen Parameter, die in Klammern eingeschlossen sind. Eine Funktion kann auch einen Wert zurückliefern. Eine Funktion muss global definiert, d.h. außerhalb einer anderen Funktion deklariert werden und ist global verfügbar.

Wenn ein Semikolon der Funktionsdeklaration folgt (anstatt einer Anweisung), dann ist dies eine Vorwärtsdeklaration einer Funktion.

Die "return"-Anweisung setzt den Rückgabewert der Funktion. Zum Beispiel, hat die Funktion "sum" (siehe unten) als Rückgabewert die Summe der beiden Parameter. Der "return"-Ausdruck ist optional.

```
sum(a, b)
{
  return a + b
}
```

Argumente einer Funktion sind (implizit deklarierte) lokale Variablen für diese Funktion. Der Funktionsaufruf bestimmt die Werte der Argumente. Ein weiteres Beispiel für eine vollständige Definition einer Funktion ist "leapyear", die "true" für ein Schaltjahr und "false" für kein Schaltjahr zurückgibt.

```
leapyear(y)
{
  return y % 4 == 0 && y % 100 != 0 || y % 400 == 0
}
```

Die Anweisungen, die in diesem Beispiel verwendet wurden, werden im Kapitel "Operatoren und Ausdrücke" auf Seite 91 behandelt.

Normalerweise beinhalten Funktionen lokale Variablendeklarationen und bestehen aus einer Block-Anweisung.

Hinweis: *Im nächsten Beispiel verhindert die "assert"-Anweisung negative Werte für den Exponenten*

```
power(x, y)
{
  /* gibt x hoch y zurück*/
  assert y >= 0

  new r = 1
  for (new i = 0; i < y; i++)
    r *= x

  return r
}
```

Eine Funktion kann mehrere "return"-Anweisungen enthalten, eine wird z.B. benutzt um schnell eine Funktion zu beenden, wenn ungültige Parameter übergeben werden, oder wenn sich herausstellt, dass die Funktion nichts zu tun hat. Wenn eine Funktion ein Array zurückgibt, müssen alle "return" Anweisungen ein Array mit derselben Anzahl von Einträgen zurückgeben.

10.8.1 Funktionsargumente ("call-by-value" versus "call-by-reference")

Die "faculty"-Funktion im nächsten Beispiel hat einen Parameter, der in der Schleife benutzt wird um die Fakultät dieser Zahl zu berechnen. Was Aufmerksamkeit verdient ist, dass die Funktion das Argument modifiziert.

```
main()
{
  new v = 5
  new f = faculty(v)
}

faculty(n)
{
  assert n >= 0

  new result = 1
  while (n > 0)
    result *= n--

  return result
}
```

Egal welchen (positiven) Wert die Variable "n" am Beginn der "while"-Schleife hatte, am Ende der Funktion wird "n" Null sein. Am Beispiel der Funktion "faculty" wird der Parameter als Wert ("by value") übergeben, damit ist eine Änderung der Variable "n" nur lokal in der Funktion "faculty" gültig. Mit anderen Worten, die Variable "v" in der Funktion "main()" hat vor und nach dem Aufruf der Funktion denselben Wert.

Argumente können als Wert ("by value") oder als Referenz ("by reference") übergeben werden. Einem Funktionsargument, das als Referenz übergeben werden soll, muss als Präfix "&" dem Namen vorangestellt werden. Als Standard werden der Funktion die Argumente als Wert übergeben.

Beispiel:

```
swap(&a, &b)
{
  new temp = b
  b = a
  a = temp
}
```

Um eine Array einer Funktion zu übergeben, fügen Sie ein Klammernpaar ("[]") dem Namen des Arguments an. Es kann auch zusätzlich die Anzahl der Einträge angegeben werden. Dies verbessert die Fehlererkennung des Parsers des Compilers.

Beispiel:

```
addvector(a[], const b[], size)
{
  for (new i = 0; i < size; i++)
    a[i] += b[i]
}
```

Arrays werden immer als Referenz übergeben.

> ***Hinweis:*** *Das Array "b" im oben gezeigten Beispiel wird in der Funktion nicht verändert. Dieses Funktionsargument wurde als "const" deklariert, um dies explizit zu machen. Zusätzlich zur verbesserten Fehlererkennung, erlaubt es dem Compiler einen effizenteren Code zu generieren.*

Das folgende Codebeispiel ruft die Funktion "addvector" auf und addiert zu jedem Element der Variablen "vect" den Wert 5:

```
new vect[3] = [ 1, 2, 3 ]

addvector(vect, [5, 5, 5], 3)

/* vect[] beinhaltet nun die Werte 6, 7 und 8 */
```

10.8.2 Benannte Parameter versus positionsgebundene Parameter

In den vorangegangenen Beispielen war die Reihenfolge der Parameter bei einem Funktionsaufruf wichtig, da jeder Parameter zu dem Funktionsparameter mit derselben Position kopiert wurde. Zum Beispiel bei der "weekday" Funktion, die unterhalb definiert wird, würde der Aufruf "weekday(12, 31, 1999)" lauten, um den Wochentag des letzten Tages des letzten Jahrhunderts zu erhalten.

```
weekday(month, day, year)
{
  /* gibt den Tag der Woche zurück: 0=Samstag, 1=Sonntag, etc. */
  if (month <= 2)
    month += 12, --year

  new j = year % 100
  new e = year / 100
  return (day + (month+1)*26/10 + j + j/4 + e/4 - 2*e) % 7
}
```

Das Datumsformat unterscheidet sich je nach Kultur und Nation, während in den Vereinten Staaten von Amerika das Format Monat/Tag/Jahr verbreitet ist, verwenden europäische Staaten oft das Format Tag/Monat/Jahr und in technischen Publikationen wird Jahr/Monat/Tag (ISO/IEC 8824) verwendet. Mit anderen Worten, keine Reihenfolge der Parameter ist "standardisiert" oder "normal". Aus diesem Grund gibt es eine alternative Möglichkeit, um Parameter an eine Funktion zu übergeben: die "benannten Parameter". Diese wird im nächsten Beispiel gezeigt (die Funktion wurde genau wie im vorherigen Beispiel deklariert).

```
new wkday1 = weekday( .month = 12, .day = 31, .year = 1999)
new wkday2 = weekday( .day = 31, .month = 12, .year = 1999)
new wkday3 = weekday( .year = 1999, .month = 12, .day = 31)
```

Bei "benannten Parametern" wird ein Punkt (".") dem Namen des Arguments vorangestellt. Das Argument der Funktion kann auf einen beliebigen Ausdruck gesetzt werden, der gültig für das Argument ist. Das Gleichheitszeichen ("=") hat im Falle eines benannten Parameters nicht die Bedeutung einer Zuordnung, sondern verknüpft den Ausdruck mit einem Funktionsargument.

Es können positionsgebundene und benannte Parameter vermischt werden, jedoch müssen die positionsgebundenen vor den benannten Parametern angegeben werden.

10.8.3 Standardwerte von Funktionsargumenten

Ein Funktionsargument kann einen Standardwert haben. Der Standardwert eines Funktionsarguments muss eine Konstante sein. Um einen Standardwert anzugeben, fügen Sie an den Namen des Parameters ein Gleichheitszeichen ("=") und den Wert an.

Wenn bei einem Funktionsaufruf ein Platzhalter anstelle eines gültigen Funktionsparameters angegeben wird, wird der Standardwert übernommen. Der Platzhalter ist das Unterstrichzeichen ("_"). Der Argumentplatzhalter ist nur für Parameter mit einem Standardwert gültig.

Die rechten Argumentplatzhalter können von der Argumentenliste entfernt werden.

Zum Beispiel, wenn die Funktion "increment" wie folgt definiert ist:

```
increment(&value, incr=1)
{
  value += incr
}
```

sind die folgenden Funktinosaufrufe alle gleich:

```
increment(a)
increment(a, _)
increment(a, 1)
```

Standardwerte für Argumente, die als Referenz übergeben werden, sind hilfreich um diese Paramter optional zu machen. Zum Beispiel, wenn die Funktion "divmod" geschrieben wurde, um sowohl den Quotienten als auch den Rest als Parameter zu übergeben.

```
divmod(a, b, &quotient=0, &remainder=0)
{
  quotient = a / b
  remainder = a % b
}
```

Mit der vorangegangenen Definition der Funktion "divmod" sind die folgenden Funktionsaufrufe alle gültig:

```
new p, q

divmod(10, 3, p, q)
divmod(10, 3, p, _)
divmod(10, 3, _, q)
divmod(10, 3, p)
divmod 10, 3, p, q
```

Das nächste Beispiel addiert die Werte von einem Array zu einem anderen. Wenn nur ein Parameter angegeben wird, dann werden die Werte des Arrays um 1 erhöht:

```
addvector(a[], const b[] = {1, 1, 1}, size = 3)
{
  for (new i = 0; i < size; i++)
    a[i] += b[i]
}
```

10.9 Erweiterungen

10.9.1 Konstanten

Struktur eines TablePoint Eintrags

```
Mdn_TablePoint[Float:.key, Float:.value]
```

Returncodes für allgemeine Zwecke

```
OK    = 0
ERROR = -1
```

Fehler Codes der Funktion Mdn_CalcTable

```
MDN_TAB_ERR_FLOOR = -1
MDN_TAB_ERR_CEIL  = -2
```

Status des Messwertkanals

```
const Mdn_ValueStatus:
{
  MDN_STATUS_OK  = 0,      // Wert ist o.k.
  MDN_STATUS_NAN = 1,      // Wert ist NAN
  MDN_STATUS_OF  = 2,      // Kanal ist Overflow
  MDN_STATUS_UF  = 3,      // Kanal ist Underflow
  MDN_STATUS_OL  = 4,      /* Kabelbruch wurde erkannt bzw. kein Sensor
                              angeschlossen */
  MDN_STATUS_SC  = 5,      // Kurzschluss wurde erkannt
};
```

Alarm Flags

```
const
{
  MDN_FLG_WARNING   = 0b00000001,  // Warnung
  MDN_FLG_ALARM     = 0b00000010,  // Alarm
  MDN_FLG_FAULTLOW  = 0b00000100,  // Technischer Fehler niedriger Priorität
  MDN_FLG_FAULTHIGH = 0b00001000,  // Technischer Fehler hoher Priorität
  MDN_FLG_UNDERFLOW = 0b10000000,  // Unterschreitung der Alarmschwelle
};                                 /* Muss gesetzt werden, wenn der Wert
                                      unterhalb des Schwellenwertes ist.
                                      Wenn dieses Flag nicht gesetzt ist, dann
                                      wird eine Überschreitung der Schwelle
                                      angenommen. */
```

Trigger Flags

```
const
{
  // Gerätespezifisch
  MDN_TRG_INTERN1    = 0b0000000000010000,
  // Gerätespezifisch
  MDN_TRG_INTERN2    = 0b0000000000100000,
  // Gerätespezifisch
  MDN_TRG_INTERN3    = 0b0000000001000000,
  // Gerätespezifisch
  MDN_TRG_INTERN4    = 0b0000000010000000,
  // Schnelle Aufzeichnung anfordern
  MDN_TRG_RECQUICK   = 0b0000000100000000,
  // Langsame Aufzeichnung anfordern
  MDN_TRG_RECSLOW    = 0b0000001000000000,
  // Starte Aufzeichnung der aktuellen Messung
  MDN_TRG_RECORD_ON  = 0b0000100000000000,
  // Verhindere Aufzeichnung der aktuellen Messung
  MDN_TRG_RECORD_OFF = 0b0001000000000000,
  // Fordere eine einzelne Verbindung mit dem Server an
  MDN_TRG_TXSTART    = 0b0010000000000000,
  // Fordere eine Online Verbindung mit dem Server an
  MDN_TRG_CONTINOUS  = 0b1000000000000000,
};
```

10.9.2 Geräteabhängige Kanalnummern

Konstantendefinition	Erklärung
MDN_CH_GSM	Interner Messkanal "GSM Stärke"
MDN_CH_AIR232_IN1 : MDN_CH_AIR232_IN64	Interface Kanäle 1-64
MDN_CH_AIR232_OUT1 : MDN_CH_AIR232_OUT64	Interface Ausgabekanäle 1-64
MDN_NUM_CHANNELS	Gesamtanzahl aller Kanäle (alle Eingänge und Ausgänge)

10.9.3 Funktionen

10.9.3.1 Core-Funktionen

heapspace();

liefert den freien Speicherplatz auf dem Heap

	Erklärung
Rückgabewert	*Der freie Speicherplatz auf dem Heap. Der Stack und der Heap besetzen einen gemeinsamen Speicherbereich, so dass dieser Wert die Anzahl der Bytes angibt, die entweder für den Stack oder den Heap übrig sind.*

funcidx(const name[]);

liefert den Index einer öffentlichen Funktion

Parameter	*Erklärung*
name	*Name der öffentlichen Funktion*

	Erklärung
Rückgabewert	• *-1, wenn keine Funktion mit dem übergebenen Namen existiert* • *Index der öffentlichen Funktion*

numargs();

liefert die Anzahl der an eine Funktion übergebenen Argumente

	Erklärung
Rückgabewert	*Die Anzahl der Argumente, die an eine Funktion übergeben wurden. numargs ist nützlich innerhalb von Funktionen mit einer variablen Argumentenliste.*

getarg(arg, index=0);

liefert den Wert des Arguments

Parameter	*Erklärung*
arg	*Die Sequenznummer des Arguments. Verwenden Sie 0 für das erste Argument.*
index	*Index, falls sich "arg" auf ein Array bezieht*

	Erklärung
Rückgabewert	*Diese Funktion liefert ein Argument aus einer variablen Argumentenliste. Wenn das Argument ein Array ist, gibt "index" den Index des gewünschten Arrayelements an. Der Rückgabewert ist der Wert des Arguments.*

setarg(arg, index=0, value);

setzt den Wert des Arguments

Parameter	*Erklärung*
arg	*Die Sequenznummer des Arguments. Verwenden Sie 0 für das erste Argument.*
index	*Index, falls sich "arg" auf ein Array bezieht*
value	*Wert auf den das Argument gesetzt werden soll*

	Erklärung
Rückgabewert	• *true, wenn der Wert gesetzt werden konnte* • *false, wenn das Argument oder der Index ungültig sind* *Diese Funktion setzt ein Argument in einer variablen Argumentenliste. Wenn das Argument ein Array ist, gibt "index" den Index des gewünschten Arrayelements an.*

tolower(c);

wandelt ein Zeichen in einen Kleinbuchstaben um

Parameter	*Erklärung*
c	*Zeichen, das in einen Kleinbuchstaben umgewandelt werden soll*

	Erklärung
Rückgabewert	*Die Kleinbuchstaben-Variante des übergebenen Zeichens, falls vorhanden, oder der unveränderte Zeichencode von "c", wenn der Buchstabe "c" kein Kleinbuchstaben-Äquivalent hat.*

toupper(c);

wandelt ein Zeichen in einen Großbuchstaben um

Parameter	*Erklärung*
c	*Zeichen, das in einen Großbuchstaben umgewandelt werden soll*

	Erklärung
Rückgabewert	*Die Großbuchstaben-Variante des übergebenen Zeichens, falls vorhanden, oder der unveränderte Zeichencode von "c", wenn der Buchstabe "c" kein Großbuchstaben-Äquivalent hat.*

swapchars(c);

vertauscht die Reihenfolge der Bytes

Parameter	*Erklärung*
c	*Wert für den die Bytes vertauscht werden sollen*

	Erklärung
Rückgabewert	*Wert, bei dem die Bytes im Parameter "c" vertauscht sind (das niedrigste Byte wird das höchste Byte)*

min(value1, value2);

liefert den kleineren der beiden übergebenen Wert

Parameter	*Erklärung*
value1	*zwei Werte, von denen der kleinere ermittelt werden soll*
value2	

	Erklärung
Rückgabewert	*der kleinere der beiden übergebenen Werte*

max(value1, value2);

liefert den größeren der beiden übergebenen Wert

Parameter	*Erklärung*
value1	*zwei Werte, von denen der größere ermittelt werden soll*
value1	

	Erklärung
Rückgabewert	*der größere der beiden übergebenen Werte*

clamp(value, min=cellmin, max=cellmax);

prüft, ob der übergebene Wert zwischen "min" und "max" liegt

Parameter	*Erklärung*
value	*Wert, der geprüft werden soll*
min	*untere Bereichsgrenze*
max	*obere Bereichsgrenze*

	Erklärung
Rückgabewert	• *"value", wenn der Wert zwischen "min" und "max" liegt* • *"min", wenn der Wert kleiner "min" ist* • *"max", wenn der Wert größer "max" ist*

10.9.3.2 Consolen-Funktionen

print(const string[] value);

druckt den angegebenen Wert auf die Standardausgabe

Parameter	*Erklärung*
value	*die auszugebende Zeichenfolge*

printi(value);

druckt den angegebenen Wert auf die Standardausgabe

Parameter	*Erklärung*
value	*der auszugebende Wert vom Typ Integer*

printf(Float:value);

druckt den angegebenen Wert auf die Standardausgabe

Parameter	*Erklärung*
value	*der auszugebende Wert vom Typ Float*

10.9.3.3 Mdn-Funktionen

getapilevel();

gibt das umgesetzte API-Level der Skript-Engine aus

Mdn_GetCh(ch, &Float:value, &Mdn_ValueStatus:status = MDN_STATUS_OK);

liest den aktuellen Wert und Status eines Kanals aus

Parameter	*Erklärung*
ch	*Nummer des Kanals (siehe "Geräteabhängige Kanalnummern" auf Seite 109)*
value	*Wert des Kanals*
status	*Status des Kanals (siehe "Status des Messwertkanals" im Kapitel "Konstanten" auf Seite 108) - OPTIONAL*

Mdn_SetCh(ch, Float:value, Mdn_ValueStatus:status = MDN_STATUS_OK);

setzt den Wert und den Status eines Kanals

Parameter	*Erklärung*
ch	*Nummer des Kanals (siehe "Geräteabhängige Kanalnummern" auf Seite 109)*
value	*Wert des Kanals*
status	*Status des Kanals (siehe "Status des Messwertkanals" im Kapitel "Konstanten" auf Seite 108) - OPTIONAL*

Mdn_GetTime(&hour=0, &minute=0, &second=0);

liest die aktuelle Zeit und speichert sie in drei separate Felder

Parameter	*Erklärung*
hour	*Stunde des aktuellen Zeitpunktes - OPTIONAL*
minute	*Minute des aktuellen Zeitpunktes - OPTIONAL*
second	*Sekunde des aktuellen Zeitpunktes - OPTIONAL*

	Erklärung
Rückgabewert	*Sekunden seit 1.1.2000*

Mdn_GetDate(&year=0, &month=0, &day=0);

liest das aktuelle Datum und speichert es in drei separate Felder

Parameter	*Erklärung*
year	*Jahr des aktuellen Zeitpunktes - OPTIONAL*
month	*Monat des aktuellen Zeitpunktes - OPTIONAL*
day	*Tag des aktuellen Zeitpunktes - OPTIONAL*

	Erklärung
Rückgabewert	*Sekunden seit 1.1.2000*

Mdn_DoW(timestamp);

berechnet den Wochentag aus einem gegebenen Timestamp

Parameter	*Erklärung*
timestamp	*Zeitstempel des zu berechnenden Tages*

	Erklärung
Rückgabewert	*Wochentag, 0=Montag ... 6=Sonntag*

Mdn_GetAlarm(ch);

gibt den Alarmstatus eines Kanals zurück

Parameter	*Erklärung*
ch	*Nummer des Kanals (siehe "Geräteabhängige Kanalnummern" auf Seite 109)*

	Erklärung
Rückgabewert	*positiv, wenn erfolgreich (siehe "Alarm Flags" im Kapitel "Konstanten" auf Seite 108), ERROR, falls kein gültiger Kanal angegeben wurde*

Mdn_SetAlarm(ch, alarm, Float:value, Float:level = 0.0);

setzt den Alarmstatus eines Kanals

Parameter	*Erklärung*
ch	*Nummer des Kanals (siehe "Geräteabhängige Kanalnummern" auf Seite 109)*
value	*Wert des Kanals, der für die Alarmberechnung herangezogen wurde*
level	*enthält den Schwellenwert des Alarms - OPTIONAL*

Mdn_SetTrigger(trigger);

setzt globale Triggerereignisse

Parameter	*Erklärung*
trigger	*Trigger-Flags (siehe "Trigger Flags" im Kapitel "Konstanten" auf Seite 108).*

	Erklärung
Rückgabewert	*OK, wenn erfolgreich*

Mdn_CalcTable(Float:key, &Float:value, const table[][Mdn_TablePoint], size = sizeof table);

sucht einen bestimmten Schlüssel in der übergebenen Tabelle und gibt den entsprechenden Wert zurück. Wenn der Schlüssel zwischen zwei Tabellenwerten liegt, wird zwischen den zwei nächsten Punkten interpoliert.

Parameter	*Erklärung*
key	*Wert, der für die Suche herangezogen wird*
value	*enthält das Ergebnis der Berechnung durch die Funktion*
table	*Die Tabelle, die durchsucht wird, muss vom Typ Mdn_TablePoint sein.*
size	*Anzahl der Elemente in der Tabelle*

	Erklärung
Rückgabewert	• *OK, wenn der entsprechende Wert gefunden wurde* • *MDN_TAB_ERR_FLOOR, wenn der Schlüssel kleiner als der erste Tabelleneintrag ist. "value" beinhaltet den ersten Tabelleneintrag.* • *MDN_TAB_ERR_CEIL, wenn der Schlüssel größer als der letzte Tabelleneintrag ist. "value" beinhaltet den letzten Tabelleneintrag.*

10.9.3.4 Mathematische Funktionen

Hilfreiche Konstanten

Definintition	Wert	Beschreibung
M_E	2.7182818284590452354	e
M_LOG2E	1.4426950408889634074	$\log_2 e$
M_LOG10E	0.43429448190325182765	$\log_{10} e$
M_LN2	0.69314718055994530942	ln 2
M_LN10	2.30258509299404568402	ln 10
M_PI	3.14159265358979323846	π
M_PI_2	1.57079632679489661923	π/2
M_PI_4	0.78539816339744830962	π/4
M_1_PI	0.31830988618379067154	1/π
M_2_PI	0.63661977236758134308	2/π
M_2_SQRTPI	1.12837916709551257390	2/sqrt(π)
M_SQRT2	1.41421356237309504880	sqrt(2)
M_SQRT1_2	0.70710678118654752440	1/sqrt(2)

fround(Float:x);

führt kaufmännisches Runden des übergebenen Floats durch

Parameter	*Erklärung*
x	*Float, der gerundet werden soll*

	Erklärung
Rückgabewert	*kaufmännisch gerundeter ganzzahliger Wert*

Die Arbeitsweise der folgenden Funktionen entspricht jener der Standad ANSI-C Implementierung:

Float:sin(Float:x);

Sinus von x

Float:cos(Float:x);

Kosinus von x

Float:tan(Float:x);

Tangens von x

Float:asin(Float:x);

arcsin(x) im Bereich [-π/2, π/2], x Element von [-1, 1]

Float:acos(Float:x);

arccos(x) im Bereich [0, π], x Element von [-1, 1]

Float:atan(Float:x);

arctan(x) im Bereich [-π/2, π/2]

Float:atan2(Float:y, Float:x);

arctan(y/x) im Bereich [-π, π]

Float:sinh(Float:x);

Sinus Hyperbolicus von x

Float:cosh(Float:x);

Cosinus Hyperbolicus von x

Float:tanh(Float:x);

Tangens Hyperbolicus von x

Float:exp(Float:x);

Exponentialfunktion e^x

Float:log(Float:x);

natürlicher Logarithmus ln(x), x › 0

Float:log10(Float:x);

Logarithmus zur Basis 10 $\log_{10}(x)$, x › 0

Float:pow(Float:x, Float:y);

x^y. Ein Argumentenfehler liegt vor bei x = 0 und y ‹= 0, oder bei x ‹ 0 und y ist nicht ganzzahlig.

Float:sqrt(Float:x);
Wurzel x, x >= 0

Float:ceil(Float:x);
kleinster ganzzahliger Wert, der nicht kleiner als x ist

Float:floor(Float:x);
größter ganzzahliger Wert, der nicht größer als x ist

Float:fabs(Float:x);
absoluter Wert | x |

Float:ldexp(Float:x, n);
$x*2^n$

Float:frexp(Float:x, &n);
zerlegt x in eine normalisierte Mantisse im Bereich [1/2, 1], die als Resultat geliefert wird, und eine Potenz von 2, die in n abgelegt wird. Ist x null, sind beide Teile des Resultats null.

Float:modf(Float:x, &Float:ip);
zerlegt x in einen ganzzahligen Teil und einen Rest, die beide das gleiche Vorzeichen wie x besitzen. Der ganzzahlige Teil wird bei ip abgelegt, der Rest ist das Resultat.

Float:fmod(Float:x, Float:y);
Gleitpunktrest von x/y, mit dem gleichen Vorzeichen wie x. Wenn y null ist, hängt das Resultat von der Implementierung ab.

10.9.4 Pawn Script Fehlercodes

Zu jedem Messzeitpunkt, nachdem alle Messwerte erzeugt wurden, wird das PAWN-Script durchlaufen. Sollte dabei ein Fehler auftreten, wird die Scriptausführung gestoppt und deaktiviert. Zudem wird in das Gerätelog der entsprechende Fehlercode eingetragen. Eine Anleitung zum Auswerten des Gerätelogs finden Sie im Kapitel „Auswerten des Gerätelogs“ (siehe "Auswerten des Gerätelogs" auf Seite 138).

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
3001	AMX_ERR_EXIT	---	---	Abbruch
3002	AMX_ERR_ASSERT	---	---	Assertion fehlgeschlagen
3003	AMX_ERR_ STACKERR	---	---	Stack / Heap Kollision (unzureichende Stack-Größe)"
3004	AMX_ERR_BOUNDS	---	---	Array-Index außerhalb des gültigen Bereichs
3005	AMX_ERR_ MEMACCESS	---	---	ungültiger Speicherzugriff
3006	AMX_ERR_INVINSTR	---	---	ungültige Anweisung
3007	AMX_ERR_ STACKLOW	---	---	Stack-Unterlauf

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
3008	AMX_ERR_ HEAPLOW	---	---	Heap Unterlauf
3009	AMX_ERR_ CALLBACK	---	---	keine (ungültige) native Callback-Funktion
3010	AMX_ERR_NATIVE	---	---	Native-Funktion ist fehlgeschlagen
3011	AMX_ERR_DIVIDE	---	---	Division durch Null
3012	AMX_ERR_SLEEP	---	---	Sleep-Modus
3013	AMX_ERR_ INVSTATE	---	---	ungültiger Zustand
3014	reserviert			
3015	reserviert			
3016	AMX_ERR_MEMORY	---	---	out of memory
3017	AMX_ERR_FORMAT	---	---	ungültiges/nicht unterstütztes P-Code Dateiformat
3018	AMX_ERR_VERSION	---	---	Datei ist für eine neuere Version des AMX
3019	AMX_ERR_ NOTFOUND	---	---	Datei oder Funktion nicht gefunden
3020	AMX_ERR_INDEX	---	---	ungültiger Index-Parameter (ungültiger Einstiegspunkt)
3021	AMX_ERR_DEBUG	---	---	Debugger kann nicht ausgeführt werden
3022	AMX_ERR_INIT	---	---	AMX nicht initialisiert (oder doppelt initialisiert)
3023	AMX_ERR_ USERDATA	---	---	Benutzer-Datenfeld kann nicht gesetzt werden (Tabelle voll)
3024	AMX_ERR_INIT_JIT	---	---	JIT kann nicht initialisiert werden.
3025	AMX_ERR_PARAMS	---	---	fehlerhafter Parameter
3026	AMX_ERR_DOMAIN	---	---	Domain-Fehler. Das Ergebnis des Ausdrucks ist nicht im gültigen Bereich.
3027	AMX_ERR_GENERAL	---	---	allgemeiner Fehler (unbekannter oder nicht spezifizierter Fehler)
3028	AMX_ERR_OVERLAY	---	---	Overlays werden nicht unterstützt (JIT) oder sind nicht initialisiert.

10.10 Beispiele

10.10.1 Sägezahngenerator

Dieses Beispiel erzeugt einen Sägezahn am Ausgang eines Geräts.

```
#include <mdlAIR232>                    /* Definitionen für das rapidM2M Developer Paket M11
                                           laden */

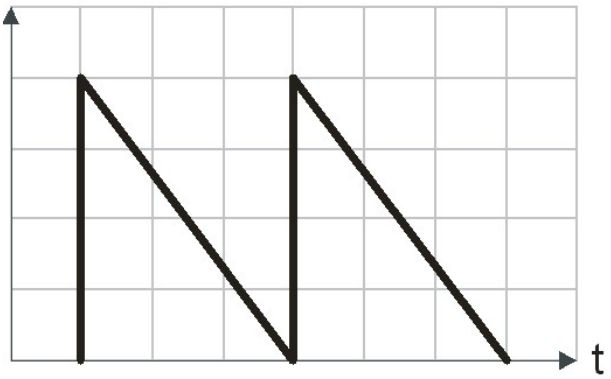
forward public Mdn_CtrlFinish();        /* Deklaration der Funktion, die am Ende des Messzyklus
                                           aufgerufen wird. */

static Float:fCurrentValue;             // Statische Deklaration der Variable fCurrentValue vom Typ Float
                                        // Diese Variable wird zwischen 2 Messzyklen nicht zurückgesetzt
main()
{
  fCurrentValue = 4.0;                  // fCurrentValue mit dem Wert 4.0 initialisieren
}

public Mdn_CtrlFinish()
{
  if(fCurrentValue <= 0.0)              // Wenn die Variable kleiner oder gleich 0.0 ist,
     fCurrentValue = 4.0;               // setze die Variable auf 4.0.

  fCurrentValue -= 0.1;                 // Ziehe 0.1 von der Variable ab.

  Mdn_SetCh(MDN_CH_IOUT1, fCurrentValue); // Setze den Ausgang 1 auf den Wert der Variable fCurrentValue
}
```



Mittels Pawn erzeugter Sägezahn

10.10.2 Wochentagbestimmung

Dieses Beispiel ermittelt den Wochentag aus dem aktuellen Zeitstempel eines Geräts. Danach wird ein Zwischenwert berechnet, indem die Zeitdifferenz zwischen den Durchläufen des Scripts durch 6 dividiert wird. Bei 2min Messintervall ergibt sich so ein Wert von 20. Dieser Zwischenwert wird dann noch durch den ermittelten Wochentag dividiert und das Ergebnis dann am Ausgang 0 ausgegeben.

```
#include <mdlAIR232>                                  /* Definitionen für das rapidM2M Developer Paket M11
                                                         laden */

forward public Mdn_CtrlFinish();                      /* Deklaration der Funktion, die am Ende
                                                         des Messzyklus aufgerufen wird. */

static iLastStamp;                                    /* Statische Deklaration der Variable
                                                         fCurrentValue vom Typ Float
                                                         Diese Variable wird zwischen 2 Messzyklen
                                                         nicht zurückgesetzt*/

main()
{
  iLastStamp = 0;                                     // iLastStamp mit dem Wert 0 initialisieren
}

public Mdn_CtrlFinish()
{
  new iDayOfWeek;                                     /* Deklaration der Variable iDayOfWeek vom
  new iCurrentStamp;                                     Typ Integer */
  new Float:fResult;                                  /* Deklaration der Variable iCurrentStamp vom
                                                         Typ Integer */
                                                      /* Deklaration der Variable fResult vom
                                                         Typ Float
                                                         Diese Variablen werden mit jedem
                                                         Messzyklus zurückgesetzt */

  if(iLastStamp != 0)                                 /* Wenn der Inhalt der Variable iLastStamp
  {                                                      bereits gesetzt wurde, wird der Inhalt der
                                                         Klammern ausgeführt. */

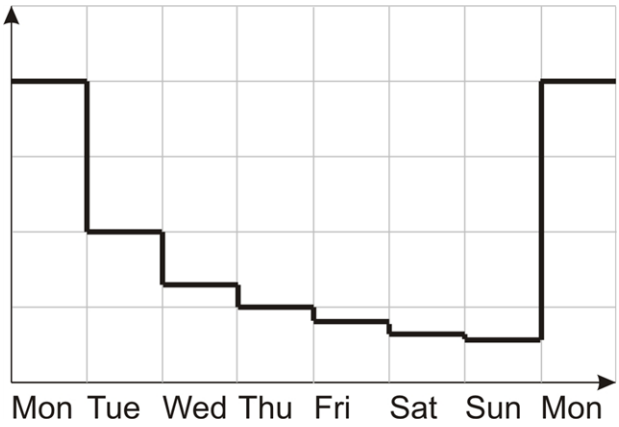
    iDayOfWeek = Mdn_DoW(Mdn_GetDate());              /* Wochentag in die Variable iDayOfWeek
                                                         speichern */
    iDayOfWeek += 1;                                  /* +1 damit bei der späteren Division nicht durch 0
                                                         (Montag) dividiert wird */

    iCurrentStamp = Mdn_GetTime();                    /* die aktuelle Zeit auslesen und in die Variable
                                                         iCurrentStamp ablegen */

    fResult = (iCurrentStamp - iLastStamp) / 6.0;     /* Zeitdifferenz zwischen aktuellem und letztem
                                                         Messzeitpunkt bilden (Messintervall) und durch
                                                         6 dividieren */
    fResult = fResult / iDayOfWeek;                   // Ergebnis durch den Wochentag dividieren

    Mdn_SetCh(MDN_CH_IOUT1, fResult);                 // Ergebnis auf den Ausgang schreiben
  }

  iLastStamp = Mdn_GetTime();                         /* aktuellen Zeitpunkt für die Verarbeitung zum
}                                                        nächsten Messzeitpunkt zwischenspeichern */
```



Mittels Pawn erzeugtes Ausgangssignal

10.10.3 Durchflussberechnung mit Tabelle

Dieses Beispiel ermittelt den Durchfluss eines Abwasserkanals aufgrund des Füllstandes des Abwasserkanals. An Eingang 1 wird der Füllstand gemessen und an Ausgang 1 soll der Durchfluss ausgegeben werden.

```
#include <mdlAIR232>                          /* Definitionen für das rapidM2M Developer Paket M11
                                                 laden */

forward public Mdn_CtrlFinish();              /* Deklaration der Funktion, welche am Ende des
                                                 Messzyklus aufgerufen wird. */
static const Table[][Mdn_TablePoint] =        /* Statische Tabelle für die Umrechnung von Höhe
[                                                zu Durchfluss */
  [ 0.0,  0.0],
  [ 2.0,  1.0],
  [ 5.0,  3.0],
  [10.0, 10.0],
  [50.0, 15.0]
];

main()
{
}

public Mdn_CtrlFinish()
{
  new Float:fHeight;                          // Deklaration der Variable fHeight vom Typ Float
  new Float:fFlow;                            // Deklaration der Variable fFlow vom Typ Float
  new Mdn_ValueStatus:iStatus;                /* Deklaration der Variable iStatus vom
                                                 Typ Mdn_ValueStatus
                                                 Diese Variablen werden mit jedem Messzyklus
                                                 zurückgesetzt */

  Mdn_GetCh(MDN_CH_UI1, fHeight, iStatus);    /* Lese den Kanal 1 aus und speichere den Wert in
                                                 die Variable fHeight und den Status in
                                                 die Variable iStatus. */

  if(iStatus == MDN_STATUS_OK)                /* Wenn der Status des Kanals 1 OK ist, führe den
  {                                              folgenden Code in den {}-Klammern aus */
    new iResult;                              // Deklaration der Variable iResult vom Typ Integer

    iResult = Mdn_CalcTable( fHeight, fFlow,  /* Diese Funktion berechnet den Durchfluss
                             Table, sizeof Table);  (fFlow) aufgrund der Höhe (fHeight)
                                                 und der Tabelle (Table).
                                                 Sollte sich die Höhe zwischen 2 Einträgen
                                                 befinden, wird das Ergebnis linear interpoliert.
                                                 Der Rückgabewert enthält den Fehlerwert
                                                 der Berechnung */

    switch(iResult)
    {
      case MDN_TAB_ERR_FLOOR:                 /* wenn die Höhe kleiner als der erste Eintrag in der
      {                                          Tabelle ist */
        fFlow = 0.0;                          // Setze den Durchfluss auf 0.0.
      }
      case MDN_TAB_ERR_CEIL:                  /* wenn die Höhe größer als der letzte Eintrag in der
      {                                          Tabelle ist */
        fFlow = 0.0;                          // Setze den Durchfluss auf 0.0.
      }
    }
  }

  Mdn_SetCh(MDN_CH_IOUT1, fFlow);             //Schreibe den Durchfluss auf den Ausgang 1.
}
```

10.11 Unterschiede zu C

- Pawn fehlt der Eingabe-Mechanismus von C. Pawn ist eine "integer-only" Variante von C. Es gibt keine Strukturen oder Unions. Floating Point-Unterstützung muss mit benutzerdefinierten Operatoren und der Hilfe von nativen Funktionen implementiert werden.
- Die Syntax für Gleitkommawerte ist strenger als die in C. Werte wie ".5" und "6." sind in C akzeptabel, aber im Pawn muss man "0.5" und "6.0" schreiben. In C ist der Dezimalpunkt optional, wenn ein Exponent enthalten ist, so kann man in C "2E8" schreiben; Pawn akzeptiert den Großbuchstaben "E" nicht. Verwenden Sie den Kleinbuchstaben "e". Es erfordert das Komma: z.B. "2.0e8" (siehe "Numerische Konstanten" auf Seite 87).
- Pawn unterstützt keine "Zeiger". Für die Übergabe von Funktionsparametern als Referenz bietet Pawn ein "Referenz"-Argument (siehe "Funktionsargumente ("call-by-value" versus "call-by-reference")" auf Seite 104). Das "Platzhalter"-Argument ersetzt einige Verwendungen des NULL-Zeigers (siehe "Standardwerte von Funktionsargumenten" auf Seite 106).
- Zahlen können mit Hexadezimal-, Dezimal-, oder Binärbasis angegeben werden. Die Oktale Basis wird nicht unterstützt (siehe "Numerische Konstanten" auf Seite 87). Hexadezimale Zahlen müssen mit "0x" ("x" in Kleinbuchstaben)beginnen. Der Präfix “0X” ist ungültig.
- "Cases" in einem "switch"-Statement sind nicht "durchfallend". Es muss dem "case"-Label zumindest eine Anweisung folgen. Um mehrere Anweisungen auszuführen, müssen Sie ein zusammengesetztes Statement (mit {}) erstellen (siehe "switch (Ausdruck) { case Liste }" auf Seite 101). In C/C++ ist die "switch"-Anweisung ein "bedingtes goto". In Pawn ist die "switch"-Anweisung ein strukturiertes "if".
- Eine "break"-Anweisung beendet nur Schleifen. In C/C++ beendet die "break"-Anweisung auch ein "case" in einer "switch"-Anweisung.
- Pawn unterstützt "array Zuweisungen", mit der Limitation, dass beide Arrays die gleiche Länge haben müssen. Zum Beispiel, wenn "a" und "b" Arrays mit 6 Zeilen sind, dann ist der Ausdruck "a=b" gültig. Neben Zeichenketten, unterstützt Pawn auch literale Arrays und somit Ausdrücke wie "a = {0,1,2,3,4,5}", wobei "a" eine Array Variable mit 6 Elementen ist.

- "defined" ist ein Operator und keine Präprozessor-Direktive. Der "defined" Operator in Pawn arbeitet mit Konstanten (deklariert mit "const"), globalen Variablen, lokalen Variablen und Funktionen.

- Der "sizeof"-Operator gibt die Größe von Variablen in "Elementen" zurück und nicht in "Bytes". Ein Element ist ein Eintrag oder ein Sub-Array. Weitere Details finden Sie im Kapitel "Sonstiges" auf Seite 95.

- Eine leere Anweisung ist ein leerer Block (mit {}), nicht ein Semikolon (siehe "Zusammengesetzte Anweisungen" auf Seite 97). Diese Änderung verhindert häufige Fehler.

- Eine Division erfolgt in der Weise, dass der Rest der Division das gleiche Vorzeichen hat (oder hätte) wie der Nenner. Für positive Nenner bedeutet dies, dass die Richtung für die Kürzung für den Operator "/" stets in Richtung des kleineren Werts (wobei -2 kleiner ist als -1) passiert. Der "%"-Operator ergibt immer ein positives Ergebnis unabhängig vom Vorzeichen des Zählers (siehe "Operatoren und Ausdrücke" auf Seite 91).

- Es gibt keinen unären Operator "+", da dieser sowieso ein "no-operation"-Operator ist ("a = +1" ist nicht gültig; korrekt: "a = 1").

- Drei der bitweisen Operatoren haben andere Prioritäten als in C. Die Prioritätsstufe des "&", "^" und "|" Operators ist höher als die relationalen Operatoren. Dennis Ritchie erklärte, dass diese Operatoren in C ihre niedrigen Prioritätsstufen bekamen, weil frühe C-Compiler noch nicht über die logischen Operatoren "&&" und "||" verfügten, so dass stattdessen bitweise "&" und "|" verwendet wurden.

- Das Schlüsselwort "const" in Pawn implementiert die "enum" Funktionalität von C.

- In den meisten Fällen sind Vorwärts-Deklarationen von Funktionen (d.h., Prototypen) nicht notwendig. Pawn ist ein 2-Pass-Compiler. Er erkennt alle Funktionen beim ersten Durchlauf und verwendet diese beim zweiten Durchlauf. Benutzerdefinierte Operatoren müssen jedoch vor der Benutzung deklariert werden. Falls vorhanden, müssen Vorwärts-Deklarationen genau mit der Definition der Funktion übereinstimmen. Die Parameternamen in den Prototypen und den Definitionen der Funktionen müssen ident sein. Pawn kümmert sich um Parameter-Namen im Prototyp auf Grund der "benannte Parameter"-Funktion. Pawn verwendet Prototypen, um vorwärtsdeklariete Funktionen aufzurufen. Um diese dabei mit benannten Parametern zu verwenden, muss der Compiler bereits die Namen der Parameter (und ihre Position in der Parameterliste) kennen. Aus diesem Grund müssen die Parameternamen in den Prototypen mit jenen in den Definitionen übereinstimmen.

Kapitel 11 Fehlersuche und Behebung

11.1 Allgemeine Probleme

Problem	Ursache/Lösung
Modul zeigt keine Reaktion (Status-LED nach Drücken der Taste zum Einschalten des Wireless GSM Controllers immer aus, keine Ausgabe von Debug-Informationen an Com 2)	• Kabelverbindungen überprüfen (siehe "Anschluss des Sensors und der Versorgung" auf Seite 39) • Prüfen Sie , ob der Jumper zum Aktivieren der Energieversorgung des Moduls (J104) korrekt gesetzt ist (siehe "Jumper zum Aktivieren der Energieversorgung des M11 Moduls (J104) " auf Seite 28). • Überprüfen Sie, ob das rapidM2M Modul M11 korrekt in die Aufnahme am rapidM2M Developer Board eingesetzt ist (siehe "Einsetzen des M11 Moduls" auf Seite 38).
Kommunikationsprobleme	• Laden Sie das Gerätelog vom myDataweb-Server und benutzen Sie myDatanetDeviceConfig für die Auswertung (siehe "Auswerten des Gerätelogs" auf Seite 138).
Es sind nicht alle/keine Daten am Server vorhanden	• Es kam zu einem Verbindungsabbruch während der Übertragung, erkennbar an einem Timeout-Eintrag in der Verbindungsliste (siehe "Benutzerhandbuch für myDataweb-Server" 206.886). Lösung: ALOHA-Übertragung auslösen oder auf die nächste zyklische Übertragung warten. • Die Zuweisung von Gerät und Messstelle ist nicht korrekt (siehe "Messstelle" auf Seite 55).
Daten der Interface Kanäle sind nicht plausibel	• Prüfen Sie, ob bei der Konfiguration der Interface Kanäle die Spaltenauswahl korrekt ist (siehe "Interface Kanäle 1-32" auf Seite 56). • Überprüfen Sie die Konfiguration der seriellen Schnittstelle (siehe "Serielle Schnittstelle" auf Seite 61). Es könnten das Zahlenformat oder das/die Zahlen-Trennzeichen falsch konfiguriert sein. • Prüfen, Sie ob der Jumper zur Versorgung der Level Shifter (J111) geschlossen ist (siehe "Jumper zur Versorgung der Level Shifter (J111)" auf Seite 24). • Sind weniger Spalten in der Datenmessage vorhanden als bei der Konfiguration der Interface Kanäle (siehe "Interface Kanäle 1-32" auf Seite 56) angegeben, wird das als ungültiger Frame erkannt, keine Daten aufgezeichnet und ein entsprechender Eintrag im Gerätelog erzeugt (siehe "Fehlercodes der UART2-Schnittstelle " auf Seite 51).

Problem	Ursache/Lösung
Keine Ausgabe von Debug-Informationen an COM1	• Kabelverbindungen überprüfen (siehe "Anschluss des Sensors und der Versorgung" auf Seite 39) • Überprüfen Sie, ob die Konfiguration Ihres Terminalprogramms mit den im Kapitel "RS232-Schnittstelle COM1 (J400)" auf Seite 23 beschriebenen Einstellungen übereinstimmt. • Prüfen Sie, ob der Jumper zur Versorgung der Level Shifter (J111) geschlossen ist (siehe "Jumper zur Versorgung der Level Shifter (J111)" auf Seite 24).
Alarmzustand eines Interface Kanals wurde nicht erkannt	Messintervall erhöhen
Alarmzustand wurde nicht übertragen, obwohl die Daten vorhanden sind	• Alarmeinstellungen des Messkanals überprüfen • Es kam zu einem Verbindungsabbruch während der Übertragung, erkennbar an einem Timeout-Eintrag in der Verbindungsliste (siehe "Benutzerhandbuch für myDataweb-Server" 206.886). Lösung: ALOHA-Übertragung auslösen oder auf die nächste zyklische Übertragung warten.
Alarmnachricht wurde nicht zugestellt, obwohl der Alarm signalisiert wird	• Einstellungen des Alarmrufplans prüfen (siehe "Benutzerhandbuch für myDataweb-Server" 206.886). • Adressdaten des Alarmrufplans prüfen (siehe "Benutzerhandbuch für myDataweb-Server" 206.886).
Am Ausgabekanal wird nicht der über die Eingabemaske am myDataweb-Server festgelegte Wert ausgegeben.	• Die Messstelleneinstellungen wurden von einer Server-PLC überschrieben. • Prüfen Sie, ob mittels Script auf den Ausgabekanal geschrieben wird. Ein von einem Script ermittelter Wert überschreibt den über die Eingabemaske am myDataweb-Server festgelegten Wert.
Die Instruction List wird nicht korrekt ausgeführt.	• Prüfen Sie, ob bei der Konfiguration der Steuerung der korrekte Script-Typ ausgewählt wurde (siehe "Steuerung" auf Seite 56). • Laden Sie das Gerätelog vom myDataweb-Server und benutzen Sie myDatanetDeviceConfig für die Auswertung (siehe "Auswerten des Gerätelogs" auf Seite 138). Eine Auflistung aller möglichen Fehlercodes der Instruction List finden Sie im Kapitel „Instruction List Fehlercodes“ (siehe "Instruction List Fehlercodes" auf Seite 76).

Problem	Ursache/Lösung
Das Pawn Script wird nicht korrekt ausgeführt.	• Prüfen Sie, ob bei der Konfiguration der Steuerung der korrekte Script-Typ ausgewählt wurde (siehe "Steuerung" auf Seite 56). • Laden Sie das Gerätelog vom myDataweb-Server und benutzen Sie myDatanetDeviceConfig für die Auswertung (siehe "Auswerten des Gerätelogs" auf Seite 138). Eine Auflistung aller möglichen Fehlercodes des Pawn Scripts finden Sie im Kapitel „Pawn Script Fehlercodes“ (siehe "Pawn Script Fehlercodes" auf Seite 119).

11.2 Log-Eingräge und Fehlercodes

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
1020	ERROR SOD	##	---	Interner System-Fehler Sollte dieser Fehler mehrmals mit demselben Parameter-Code im Gerätelog enthalten sein, liegt unter Umständen ein Hardwareproblem vor. Kontaktieren Sie in diesem Fall den Hersteller (siehe "Kontaktinformationen" auf Seite 149).
1200	MODEM ERROR			Modemfehler (siehe "Modemfehler" auf Seite 135)
1212	ERROR MODEM IRREGULAR OFF	##	---	Zeigt eine fehlerhafte Verbindung an. Der Parameter enthält dabei einen Zähler, der angibt wie viele Verbindungen hintereinander nicht funktioniert haben.
1252	MODEM TO CON	##	---	Timeout während des Verbindungsaufbaus. Der Parameter enthält dabei einen Zähler, der angibt wie viele Verbindungsaufbauversuche hintereinander nicht funktioniert haben.
1281	ZLIB STREAMPROCESS ERR	##	---	interner Fehler Sollte dieser Fehler mehrmals im Gerätelog enthalten sein, kontaktieren Sie den Hersteller (siehe "Kontaktinformationen" auf Seite 149).
1282	ZLIB STREAMFINISH ERR	##	---	interner Fehler Sollte dieser Fehler mehrmals im Gerätelog enthalten sein, kontaktieren Sie den Hersteller (siehe "Kontaktinformationen" auf Seite 149).
1440	DECAY MEM ERR	---	---	reserviert für Erweiterungen

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
1600 (1/2)	STATE (1/2)	1	OK	letzte Verbindung o.k.
		2	TRANSMISSION ERROR	letzte Übertragung fehlerhaft später erneut versuchen
		3	MEASUREMENT ERROR	letzte Messung fehlerhaft • Überprüfen Sie die Konfiguration • Überprüfen Sie die Verbindung zwischen Gerät uns Sensor
		4	HOLD	Standby (GPRS an, Messung aus) Wird der ALOHA–Übertragungsmodus am Gerät ausgelöst, schaltet das rapidM2M Developer Paket M11 wieder in den Modus "RUN" (GPRS an, Messung an).
		5	TRANSPORT	Transportsperre (GPRS aus, Messung aus) siehe "Standby"
		6	OFFLINE	Offline (GPRS aus, Messung an) siehe "Standby"
		7	NO NET (NET LOCK)	Netzsperre/kein passender Provider [1)] • Versuchen Sie die Antennenposition zu verbessern. • Überprüfen Sie, ob sich das Gerät im Versorgungsbereich befindet (www.microtronics.at/footprint). • Netzsperre aufheben (siehe "Benutzerhandbuch für myDatanetDeviceConfig" 206.887)
		8	NO NET	kein GSM Netzwerk • Antennenpositionierung verbessern • später erneut versuchen
		10	GPRS ERROR	keine GPRS Verbindung Versuchen Sie die Antennenposition zu verbessern.

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
1600 (1/2)	STATE (1/2)	11	IP ERROR	kein myDataweb-Server erreichbar • Überprüfen Sie, ob am myDataweb-Server der Port 51241 frei geschaltet ist. • später erneut versuchen
		12	SIM CARD ERROR	fehlerhafter SIM-Chip Kontaktieren Sie den Support.
2000	MODULE ERR			Modul-spezifischer Fehler (siehe "Fehlercodes der UART2-Schnittstelle " auf Seite 51).
3000 - 3099				Fehlercodes der Scriptausführung. Abhängig vom ausgewählten Script Typ gelten folgende Fehlerbeschreibungen: • "Instruction List Fehlercodes" auf Seite 76 • "Pawn Script Fehlercodes" auf Seite 119

[1)] Eine Netzsperre lässt sich nur bei Geräten mit USB-Anschluss in Verbindung mit dem Konfigurationsprogramm myDatanetDeviceConfig aktivieren.

11.2.1 Modemfehler

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
GPRS-Fehler (1/2)				
1200	BEARER LINE BUSY	-993	---	Leitung belegt später erneut versuchen
	BEARER NO ANSWER	-992	---	Keine Antwort • später erneut versuchen • Überprüfen Sie, ob sich das Gerät im Versorgungsbereich befindet (www.microtronics.at/footprint).
	BEARER NO CARRIER	-991	---	Kein Träger • später erneut versuchen • Überprüfen Sie, ob sich das Gerät im Versorgungsbereich befindet (www.microtronics.at/footprint).
	BEARER GPRS FAILED	-988	---	GPRS Setup-Fehler • Versuchen Sie die Antennenposition zu verbessern. • Überprüfen Sie, ob sich das Gerät im Versorgungsbereich befindet (www.microtronics.at/footprint).
	BEARER PPP LCP FAILED	-987	---	Fehler beim Aushandeln des LCP interner Fehler
	BEARER PPP AUTH FAILED	-986	---	Fehler bei der PPP Authentifizierung interner Fehler
	BEARER PPP IPCP FAILED	-985	---	Fehler beim Aushandeln des IPCP interner Fehler
	BEARER PPP LINK FAILED	-984	---	PPP Gegenstelle reagiert nicht auf Echo-Request interner Fehler
	BEARER PPP TERM REQ	-983	---	PPP Session wurde von der Gegenstelle beendet. interner Fehler

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
GPRS-Fehler (2/2)				
	BEARER CALL REFUSED	-982	---	eingehender Anruf abgelehnt später erneut versuchen
	BEARER UNKNOWN	-981	---	später erneut versuchen
Fehler beim Aufbauen der GPRS-Verbindung				
1200	BEARER BAD HDL	-980	---	invalid handle interner Fehler
	BEARER BAD STATE	-979	---	Der Träger wurde nicht gestoppt. interner Fehler
	BEARER DEV	-978	---	Fehler bei der Link-Layer Initialisierung interner Fehler
	BEARER GPRS START UNKNOWN	-977	---	interner Fehler
Fehler beim Abbauen der GPRS-Verbindung				
1200	BEARER GPRS STOP BAD HDL	-976	---	invalid handle interner Fehler
	BEARER GPRS STOP UNKNOWN	-975	---	interner Fehler
1200	NETLOCK FAILURE	-970	---	Fehler bei der Netzauswahl • Überprüfen Sie die Einstellungen der Netzsperre bzw. entfernen Sie die Netzsprerre[1)]. • Überprüfen Sie, ob sich das Gerät im Versorgungsbereich befindet (www.microtronics.at/footprint).

[1)] Eine Netzsperre lässt sich nur bei Geräten mit USB-Anschluss in Verbindung mit dem Konfigurationsprogramm myDatanetDeviceConfig aktivieren.

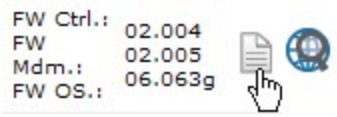
Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
1200	BAND SEL FAILED	-969	---	Es konnte weder auf dem GSM900/1800 noch auf dem GSM85071900 Band ein Netzwerk gefunden werden. • Versuchen Sie die Antennenposition zu verbessern. • Überprüfen Sie, ob sich das Gerät im Versorgungsbereich befindet (www.microtronics.at/footprint).
TCP Channel Fehler (1/2)				
1200	CHANNEL ABORTED	-965	---	Es wird versucht auf einen nicht mehr verfügbaren TCP-Client zu schreiben/lesen. später erneut versuchen
	CHANNEL CSTATE	-964	---	Der Channel ist nicht im "WIP_CSTATE_ READY" Zustand. später erneut versuchen
	CHANNEL NOT SUPPORTED	-963	---	Die Option wird vom Channel nicht unterstützt. später erneut versuchen
	CHANNEL OUT OF RANGE	-962	---	Der Options-Wert liegt außerhalb des zulässigen Bereichs. später erneut versuchen
	CHANNEL MEMORY	-961	---	interner Fehler
	CHANNEL INTERNAL	-960	---	interner Fehler
	CHANNEL INVALID	-959	---	ungültige Option oder Parameter interner Fehler
	TCP DNS FAILURE	-958	---	Der Name konnte nicht in eine IP-Adresse aufgelöst werden. interner Fehler
	CHANNEL RESOURCES	-957	---	Es stehen keine weiteren TCP-Puffer mehr zur Verfügung später erneut versuchen

Log-Eintrag		Parameter		Beschreibung
Code	Klartext	Code	Klartext	
TCP Channel Fehler (2/2)				
1200	CHANNEL PORT IN USE	-956	---	TCP-Server-Port wird bereits verwendet. später erneut versuchen
	CHANNEL REFUSED	-955	---	Die TCP-Verbindung wurde vom Server abgelehnt. später erneut versuchen
	CHANNEL HOST UNREACHABLE	-954	---	keine Route zum Host. später erneut versuchen
	CHANNEL NETWORK UNREACHABLE	-953	---	kein Netz erreichbar später erneut versuchen
	CHANNEL PIPE BROKEN	-952	---	TCP-Verbindung unterbrochen später erneut versuchen
	CHANNEL TIMEOUT	-951	---	Timeout (DNS-Request, TCP-Verbindung, Ping-Response,..) später erneut versuchen
	CHANNEL UNKNOWN	-950	---	interner Fehler

11.3 Auswerten des Gerätelogs

11.3.1 Auswerten des Gerätelogs am myDataweb-Server

Am myDataweb-Server sind die letzten 300 Logeinträge über den unten abgebildeten Button, der sich in der Messgeräteliste befindet, abrufbar. Da die Logeinträge genau wie die Messdaten im Übertragungsintervall zum Server gesendet werden, sind immer nur die Logeinträge bis zur letzten Serververbindung verfügbar.



Eine genauere Beschreibung zur Auswertung des Gerätelogs am myDataweb-Server finden Sie im Handbuch des Servers ("Benutzerhandbuch für myDataweb-Server" 206.886).

11.3.2 Auswerten des Gerätelogs mittels myDatanetDeviceConfig

Eine genaue Beschreibung zur Auswertung des Gerätelogs mittels myDatanetDeviceConfig finden Sie im Handbuch zum myDatanetDeviceConfig ("Benutzerhandbuch für myDatanetDeviceConfig" 206.887).

Kapitel 12 Ersatzteile und Zubehör

12.1 rapid M2M-Module

Beschreibung	Menge	Bestellnummer
rapidM2M Modul M11	1	300068

12.2 Antennen

Beschreibung	Menge	Bestellnummer
Standantenne Quad Band SMA	1	300085
Verlängerung für Antenne 900 SMA	1	206.807

12.3 Versorgung

Beschreibung	Menge	Bestellnummer
Steckernetzteil 24V (U: 24V, I max.: 1A, P max.: 24W)	1	213.814
MDN Netzgerät (U: 12V, I max.: 0,5A, P max.: 6W)	1	206.623
Primärstecker EU für Ladegerät/Netzgeräte [1)]	1	300027
Primärstecker UK für Ladegerät/Netzgeräte [1)]	1	300028
Primärstecker US für Ladegerät/Netzgeräte [1)]	1	300029
Primärstecker AUS für Ladegerät/Netzgeräte [1)]	1	300030

[1)] Nicht geeignet für das im Standardlieferumfang des Developer Pakets(300070) enthaltene Steckernetzgerät.

12.4 Adapter

Beschreibung	Menge	Bestellnummer
Nullmodemadapter 9pol. D-Sub female/male	1	206.686

12.5 Sonstiges Zubehör

Beschreibung	Menge	Bestellnummer
myDatanet Tool Pen	1	206.646

Kapitel 13 Applikationsbeispiele

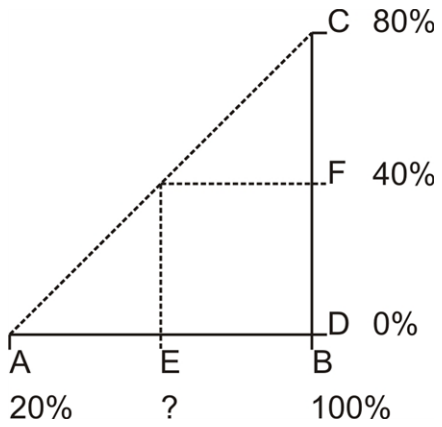
13.1 Laternensteuerung

13.1.1 Anforderung

Der Laternenkontroller misst die Umgebungshelligkeit und den Stromverbrauch der Laterne. Die Helligkeit der Lampe kann dadurch eingestellt werden, dass dem Laternenkontroller ein Stellwert für die PWM zum Ansteuern des Schalters übergeben wird. Das rapidM2M Developer Paket M11 liest den Status über die RS232 vom Laternenkontroller und berechnet mittels Instruction List den Stellwert für die PWM, der dann wieder über die RS232 Schnittstelle an den Laternenkontroller übermittelt wird. In diesem vereinfachten Beispiel soll die Helligkeit der Lampe nur abhängig von der Umgebungshelligkeit gesteuert werden. Mittels Instruction List könnten aber noch eine Vielzahl an anderen Faktoren und Parametern in die Berechnung des Stellwerts miteinbezogen werden.

Regelungsvorgaben:

Umgebungshelligkeit [%]	PWM [%]
100	0
80	100
von 80...0 linear interpolieren	von 100...20 linear interpolieren
0	20



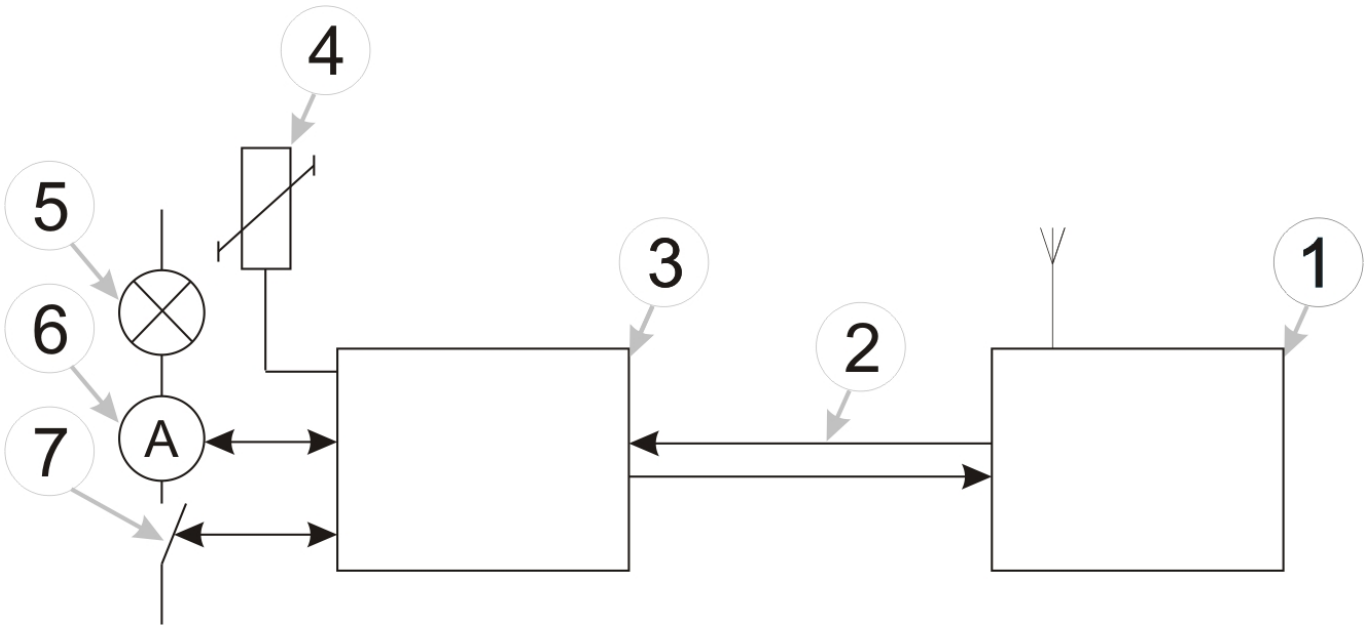
A, B, E...PWM
C, D, F...Umgebungshelligkeit

Steigung = (B-A)/(C-D)
Steigung = (100%-20%)/(80%-0%)
Offset = 20%

E = ((B-A)/(C-D)) * F + A
E = ((100%-20%)/(80%-0%)) * 40% + 20% = 60%

Annahme:

- Befehl zum Initialisieren des Laternenkontrollers nach dem PowerOn: INIT
- Befehl zum Auslesen des Status des Laternenkontroller: RS
- Aufbau des Statusframe:[STX]‹Lampe[ein/aus]›;‹Stromverbrauch [A]›;‹Umgebungshelligkeit [%]›[ETX]
- Erwarteter Steuerframe: [STX]‹Lampe[ein/aus]›;‹PWM Einstellung [%]›[ETX]



Prinzipschaltbild der Laternensteuerung

1 rapidM2M Developer Paket M11	5 Lampe
2 RS232 Schnittstlle	6 Strommesssensor (0-10A)
3 Laternenkontroller	7 Schalter für die Helligkeitsregelung (PWM 0-100%)
4 Umgebungshelligkeitssensor (0-100%)	

13.1.2 Erforderliche Konfiguration

An dieser Stelle wird nicht die komplette Konfiguration für das Applikationsbeispiel aufgeführt sondern nur die wichtigsten Einstellungen, um die Anforderungen des beschriebenen Beispiels zu erfüllen.

Steuerung	Script-Typ	IL
	Script	siehe "Instruction List" auf Seite 145

Interface Kanäle1-32 -› Basis	Bezeichnung 1	"Zustand der Lampe"
	Interface	RS232
	Spalte	1
	Bezeichnung 2	"Stromverbrauch"
	Interface	RS232
	Spalte	2
	Einheit	"A"
	Bezeichnung 3	"Umgebungshelligkeit"
	Interface	RS232
	Spalte	3
	Einheit	"%"
Interface Ausgabekanäle 1-32	Bezeichnung 1	"Zustand der Lampe"
	Modus	ein
	Bezeichnung 2	PWM Einstellung
	Modus	ein
Serielle Schnittstelle	Datensatz Struktur	[STX]...[ETX]
	Zahlen Trennzeichen	;
	Zahlenformat	1,000.00
	Init Sequenz	INIT
	Messdaten Sequenz	RS

13.1.3 Instruction List

```
LD %IR3     (* Lädt die aktuelle Umgebungshelligkeit *)
LE  80.0    (* Prüfen, ob die Umgebungshelligkeit <= 80% *)
JMPCN OFF   (* Wenn Prüfung der vorherigen Zeile nicht 0 ergibt, zum Label "OFF" springen *)
LD %IR3     (* Lädt die aktuelle Umgebungshelligkeit *)
MUL 1.0     (* Mit der Steigung ( (100-20)*(80-0) multiplizieren*)
ADD 20.0    (* Offset (20) addieren *)
ST %QR1     (* Berechneten Wert auf den Ausgang "PWM Einstellung" schreiben *)
LD 1        (* Wert zum Einschalten der Lampe in den Acuu kopieren *)
ST %QD0     (* Wert in den Ausgang "Zustand der Lampe" schreiben *)
JMP END     (* Sprung zum Label "END" (Block zum Abschalten der Lampe überspringen *)
OFF:        (* Label "OFF" *)
LD 0        (* Wert zum Ausschalten der Lampe in den Acuu kopieren *)
ST %QD0     (* Wert in den Ausgang "Zustand der Lampe" schreiben *)
END:        (* Label "END" *)
```

13.1.4 Kommunikation auf der Schnittstelle

[STX]INIT[ETX]	Befehl zum Initialisieren des Laternenkontrollers (Nur ein einziges Mal bei PowerOn)	M11 -› Ctrl
[STX]RS[ETX]	Befehl zum Auslesen des Status (im Messintervall)	M11 -› Ctrl
[STX]0;0.00;80.0[ETX]	Status: Lampe "0", Stromverbrauch "0.00"A, Umgebungshelligkeit "0.0"%	Ctrl -› M11 .
[STX]1;100.0[ETX]	Steuerframe: Lampe "1", PWM Einstellung "100.0"%	M11 -› Ctrl

Kapitel 14 Dokumentenhistorie

Rev.	Datum	Änderungen
02	05.12.2012	**Kapitel "Technische Daten" auf Seite 11** *Anzahl der verfügbaren "Interface Kanäle" und "Interface Ausgabekanäle" hinzugefügt* **Kapitel "Auswahl des GSM-Netzes" auf Seite 18** *Kapitel, die die Funktionsweise der GSM-Netzauswahl beschreiben, hinzugefügt* **Kapitel "Script-Abarbeitung" auf Seite 19** *Kapitel hinzugefügt* **Kapitel "Fehlercodes der UART2-Schnittstelle " auf Seite 51** *Aufschlüsselung der Fehlercodes hinzugefügt* **Kapitel "Steuerung" auf Seite 56** *Auswahl des Pawn Scripts* *Anpassung der Erklärung an modifizierte Eingabemaske* **Kapitel "Pawn Script" auf Seite 85** *Kapitel, die die Funktionsweise des Pawn Scripts beschreiben, hinzugefügt.* **Kapitel "Fehlersuche und Behebung" auf Seite 129** *Fehlerbeschreibungen und Lösungsvorschläge ergänzt* **Kapitel "Log-Eingräge und Fehlercodes " auf Seite 132** *Aufschlüsselung der Fehlercodes hinzugefügt* **Kapitel "Ersatzteile und Zubehör" auf Seite 141** *MDN Netzgerät (206.623) hinzugefügt*

Kapitel 15 Kontaktinformationen

Support & Service:
Microtronics Engineering GmbH
Hauptstrasse 7
3244 Ruprechtshofen
Austria, Europe
Tel. +43 (0)2756 7718023
support@microtronics.at
www.microtronics.at

Microtronics Engineering GmbH (Headquarter)
Hauptstrasse 7
3244 Ruprechtshofen
Austria, Europe
Tel. +43 (0)2756 77180
Fax. +43 (0)2756 7718033
office@microtronics.at
www.microtronics.at

Microtronics Engineering GmbH
Windthorststrasse 5
63179 Obertshausen
Germany, Europe
Tel. +49 (0)6104 971057
Fax. +49 (0)6104 971058
office@microtronics.de
www.microtronics.de

Microtronics-UK Ldt.
9b The Shade, Soham, Ely
Cambs, CB7 5DE
United Kingdom, Europe
Tel. +44 (0)1353 727017
office@microtronics-uk.co.uk
www.microtronics-uk.co.uk