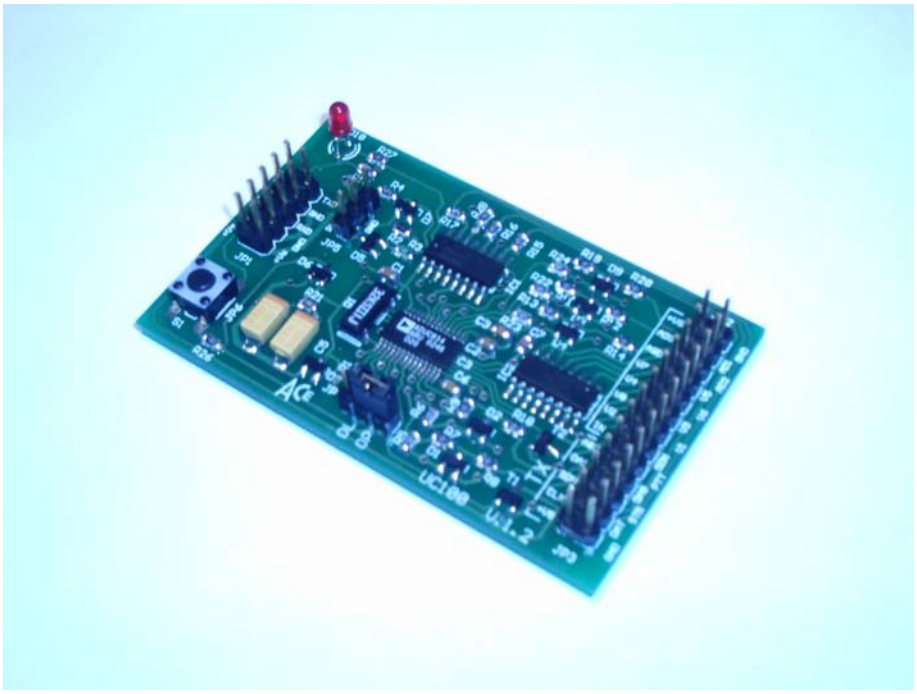
Universal Controller UC100

Benutzerhandbuch

HW Vers.1.2 SW Vers.1.01



Kurzbeschreibung
Inbetriebnahme
Spezifikationen
Anschlussbeschreibung
Anschaltbeispiele
Programmierbefehle

Inhalt

Kurzbeschreibung

Der Universal Controller UC100 ist eine freiprogrammierbare Steuerungseinheit, optimiert für die Informationsübertragung in Funksystemen. Er enthält einen 8051 kompatiblen Prozessorkern mit 2x 12 Bit DAC zur Erzeugung der Modulationssignale sowie insgesamt 11x ADC Eingänge zur Verarbeitung analoger Eingangssignale. Über die RS-232 Schnittstelle kann ein anwender-spezifisches Programm in das prozessorinterne EEPROM geladen werden. Unter Verwendung eines Standard - Terminalprogramms korrespondiert der PC mit dem UC100, wobei eine einfache Basic-ähnliche Programmiersprache zum Einsatz kommt. Der Controller läuft danach eigenständig mit dem gespeicherten Programm. Durch die geringe Leistungsaufnahme ist das Modul ideal für batteriebetriebene Anwendungen geeignet.

Es stehen 4 Schnittstellen nach außen zur Verfügung:

- Spannungsversorgung (3 … 5V DC, max. ca. 10mA)
- Computerschnittstelle (RS-232 kompatibel)
- Verbindung zu Sensorik (11x Analog In, Shutdown, +V)
- Anschluss eines Senders (Mod., PTT, ser.Bus, +V)

Programmiermöglichkeiten:

- Ausgabe von Fixtexten in Morsetelegrafie (A1, F1, F2)
- Ausgabe von Fixtexten in PSK31
- Ausgabe von Fixtexten im AX-25 Protokoll (AFSK, FSK)
- Ausgabe von Ziffern in DTMF Codierung
- Ausgabe von Tönen (Fixfrequenzen)
- Ausgabe der Temperatur des On - Board Temp.- Sensors in Morsetelegraphie, PSK31, DTMF, AX-25, an LCD´s oder seriell
- Ausgabe jeglicher Analogspannungswerte wie oben
- Einlesen von Analogspannungen, arithmetische Bearbeitung und Ausgabe dieser Werte wie oben
- Umwandlung von Analogspannungen in Audiofrequenzen
- Ausgabe von Kombinationen aus Fixtexten und Messwerten
- Senden von Meldungen bei Überschreiten von Grenzwerten
- Kombinationen aus oben aufgeführten Punkten
- Arbeiten in einer Programmschleife oder Aktivierung durch ein externes DC oder NF Signal
- Vielfältige weitere Anwendungen durch freie Programmier-Möglichkeit

Weiters können die wichtigsten Parameter wie Modulationspegel, DC – Anteil, Morsetempo, Baudraten bei AX-25, AFSK Frequenzen etc. durch einfache Befehle beliebig konfiguriert und dadurch sehr einfach an bestehende Sender angepasst werden.

Der UC100 Universalcontroller erlaubt außerhalb der aktiven Phasen einen Stromsparmodus, in welchem die Stromaufnahme auf wenige µA reduziert wird und dadurch die Batterie-Lebensdauer extrem verlängert. Das laufende Programm wird dadurch nicht beeinflusst. Während der Stromsparperiode steht ein Shutdown – Signal zur Verfügung, um eventuelle zusätzliche Hardware (Sensorik etc.) ebenfalls in der Stromaufnahme zu reduzieren.

Inbetriebnahme

1. Verbindung zum PC

TXD (Pin 2), RXD (Pin 3) und GND (Pin 5) der 9-poligen SubD - Buchse des seriellen Schnittstellenkabels mit den gleichnamigen Pins des UC100 verbinden.

2. PC – Software

Sollte mit dem UC100 eine passende Terminalsoftware mitgeliefert worden sein, kann diese am PC installiert und damit die gesamte Kommunikation und Programmierung vorgenommen werden.
Die für Installation und Betrieb notwendigen Informationen können der entsprechenden Readme - Datei entnommen werden.
Es können auch andere Programme zur Datenkommunikation verwendet werden. Notwendig ist dafür ein Standard Terminal Programm zur Kommunikation auf einem seriellen Com – Port.
Einstellung: 1200 Bps, 8 Datenbits, 1 Stoppbit, kein Paritybit, kein Echo, kein Hardware Handshake.
Nützlich ist auch ein Standard-Texteditor zum Download von bereits geschriebenen Userprogrammen.

3. Datenaustausch

Wenn die Datenleitung korrekt mit dem Controller verbunden ist, kann die Betriebsspannung angelegt werden. Damit wird ein Reset Impuls erzeugt und ein eventuell bereits im Flash-Speicher (EEPROM) geladenes User - Programm startet.
Durch kurzes Drücken des Tasters S1 (INT) auf dem Board kann der Controller in den Programmier - Mode geschalten werden. Anzeige: **ram mode (list|run|line|ram|test|burn)**
Der Controller steigt damit aus dem im EEPROM abgelegten Userprogramm aus und kann entweder mit der PC-Tastatur Zeile für Zeile oder durch Download eines im Texteditor gespeicherten Programms mit einem Userprogramm geladen werden.

4. Erstellung eines User – Programms

4.1 RAM – Mode

Diese Betriebsart kann zum schnellen Test von kleineren Userprogrammen verwendet werden. Hierbei werden die einzelnen Befehlszeilen im prozessorinternen RAM gespeichert. Durch den begrenzten Speicherplatz erlaubt dieser Betrieb allerdings nur ca. 30% des möglichen Userprogrammumfangs. Außerdem gehen die gespeicherten Befehle beim Abschalten der Betriebsspannung des UC100 verloren.

4.1.1 Zeilenweise Eingabe der Befehle mit der Tastatur

Zu Beginn den Befehl **ram** eingeben oder Taster S1 kurz drücken. Nun kann ein Userprogramm im RAM Mode erstellt werden, indem Befehle aus der Liste der möglichen Instruktionen eingegeben werden. Die notwendige Syntax muss dabei beachtet werden. Wenn nicht, erscheint nach betätigen der Enter-Taste in der folgenden Zeile eine Error Meldung. Die Programmeingabe kann jedoch danach ohne Probleme mit der Eingabe der korrekten Programmzeile fortgesetzt werden.
Sollte der zur Verfügung stehende Speicherplatz im RAM überschritten werden, erscheint die Anzeige „memory full". Wenn dennoch im RAM Betrieb weiter gearbeitet werden soll, muss die Anzahl der Programmzeilen reduziert werden. Andernfalls muss das Userprogramm über den Burn Mode in den Controller geschrieben werden. Hierbei steht der gesamte Speicherplatz von 640 Byte zur Verfügung.
Ein Programm, welches im RAM geschrieben und ausgeführt werden soll, wird mit dem Befehl **run** in der letzten Zeile gestartet. Gestoppt wird es wiederum durch Drücken der Taste S1 (INT). Es erscheint wieder das Grundmenü des Programmiermodus **ram mode (list|run|line|ram|test|burn)**
Der Befehl **list** listet die im RAM abgelegten Userprogramm-Befehle zur Überprüfung auf.

4.1.2 Download des Userprogramms als fertiges File

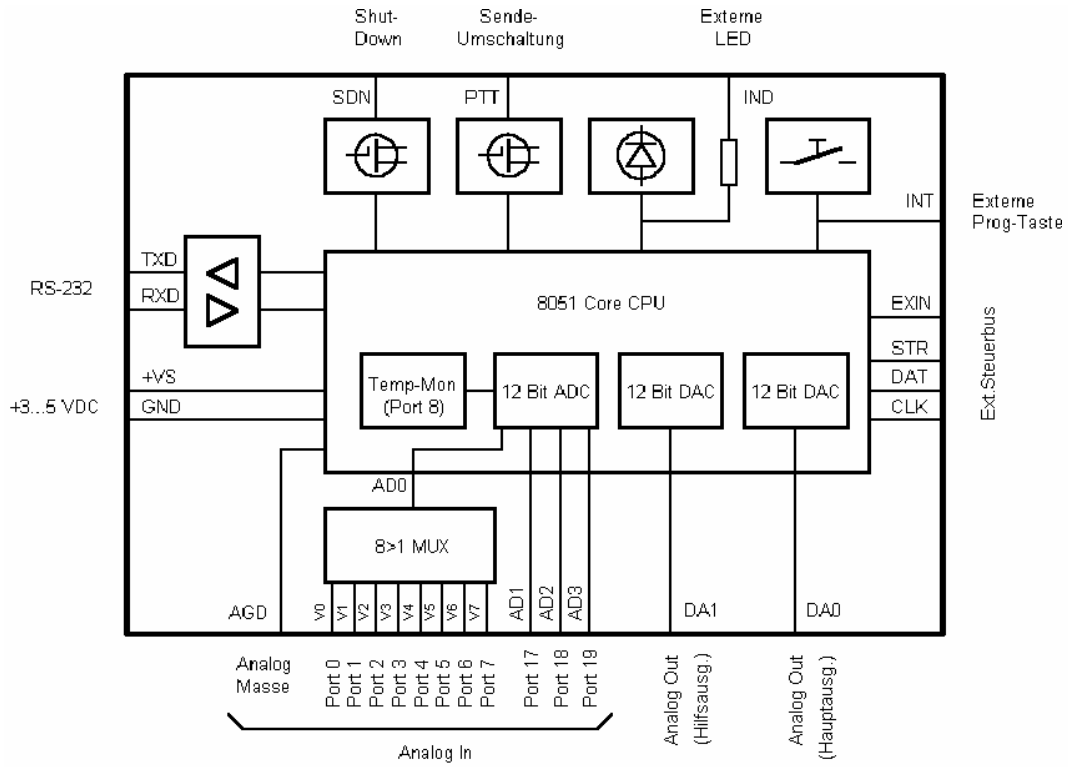
Das Userprogramm kann auch in einem Standard Texteditor geschrieben und als txt - File in den UC100 geladen werden. Damit müssen bei mehrmaligem Umändern des Programms nicht jedes Mal alle Befehlszeilen neu eingegeben werden. Man ändert die jeweilige Zeile im Editor und führt einen File - Download über das Terminalprogramm durch. Durch die schnelle Datenübertragung zum Controller muss jedoch in diesem Fall das sonst in der Übertragung vorgesehene Echo ausgeschaltet werden. Das geschieht einfach durch Verwendung des leicht abgeänderten Befehls **ram&** in der ersten Zeile.

4.2 BURN – Mode

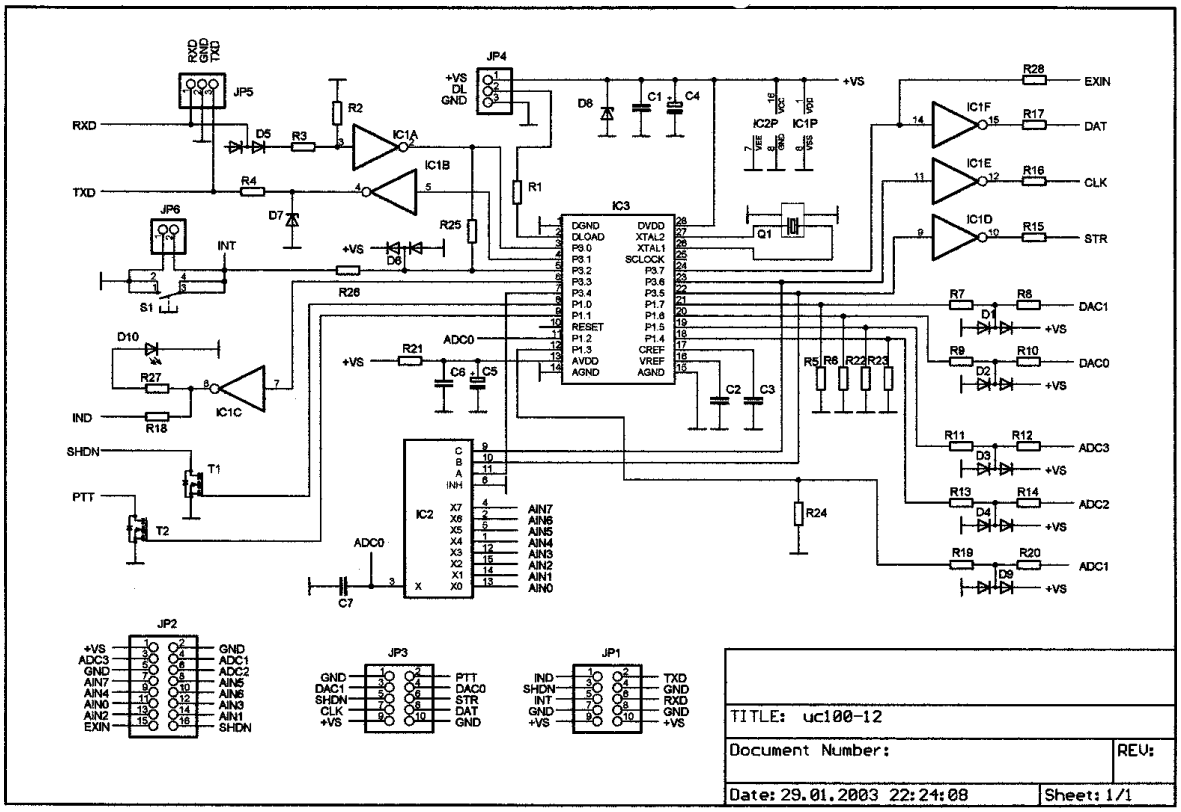
Zum Abspeichern eines Userprogramms im EEPROM verwendet man den Befehl **burn**. Alle nun folgenden Programmzeilen werden Zeichen für Zeichen in das EEPROM gebrannt. Durch diese Betriebsart wird nicht nur das Userprogramm unverlierbar in den UC100 geschrieben, sondern es steht auch der gesamte mögliche Speicherplatz zur Verfügung.
Tritt bei dieser Art der Programmierung ein Syntax Fehler auf, erscheint eine Error Meldung und es wird in den Test Mode geschaltet. So kann problemlos die korrekte Eingabeform getestet und danach weiterprogrammiert werden.
Durch Eingabe von **list** im Burn Mode besteht die Möglichkeit, die Anzahl der Programm Bytes abzufragen und das gesamte Userprogramm aufzulisten. Es stehen insgesamt 640 Byte zur Verfügung, die jedoch auch sehr komplexe Programmabläufe ermöglichen
In diesem Betrieb empfiehlt es sich, hier immer den Download aus dem Texteditor zu verwenden. Dazu schreibt man ähnlich wie im RAM Mode in der ersten Zeile den Befehl **burn&**, danach die Befehle des Userprogramms, und beendet mit dem Befehl **end** in der letzten Zeile. Am Bildschirm erscheint danach die Anzahl der programmierten Zeilen.
Das Programm kann nun entweder durch Eingabe von **run** oder durch Ab- und wiederum Einschalten des UC100 gestartet werden. Es startet ab nun immer nach Anlegen der Betriebsspannung.

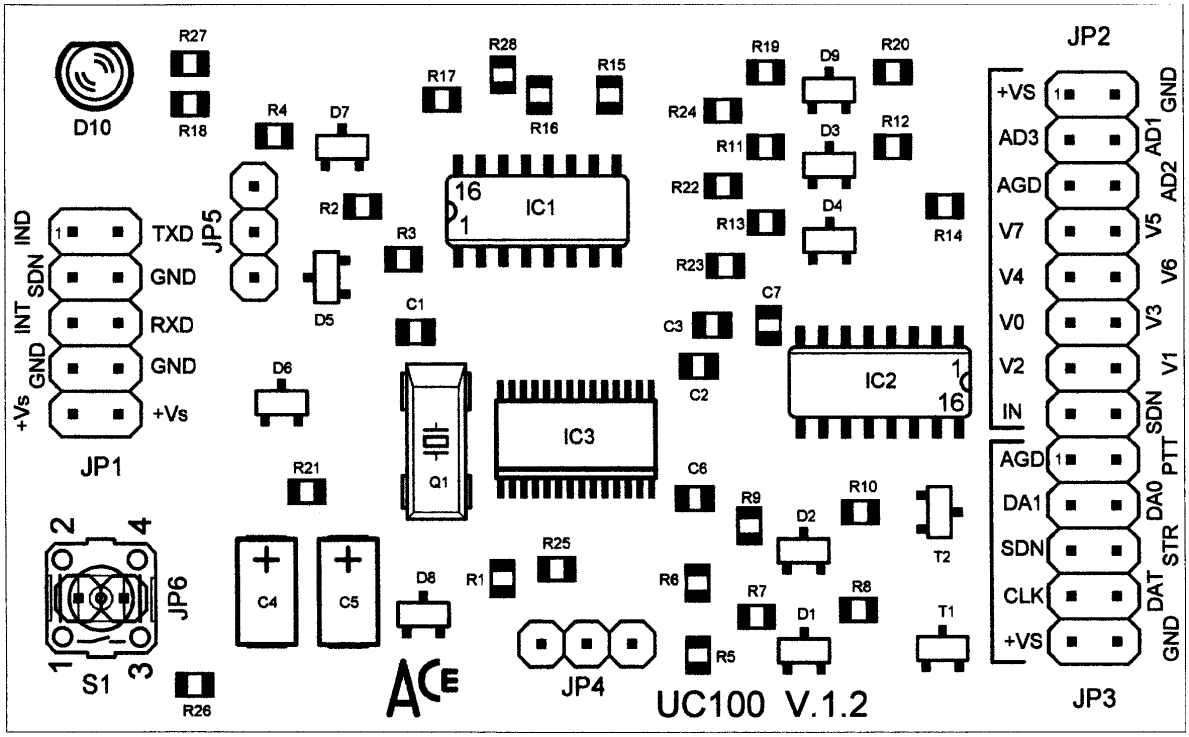
Spezifikationen

Betriebsspannung	3,0 ... 5,5 VDC stabilisiert
Stromaufnahme (ohne ext. Lasten)	ca. 10 mA
Stromaufnahme (Sleep-Mode)	ca. 50 µA
Eingangsimpedanz der AD-Wandler	ca. 1 MOhm
Eingangsspannung der AD-Wandler	max. +2,5 VDC
Ausgangsimpedanz der DA-Wandler	ca. 200 Ohm
Ausgangsspannung der DA-Wandler	max. +2,5 VDC (oder +Vs, schaltbar)
Schaltstrom bei PTT und SHDN	max. 200 mA (gegen GND)
Ausgangsimpedanz der TX-Bus Signale	ca. 470 Ohm

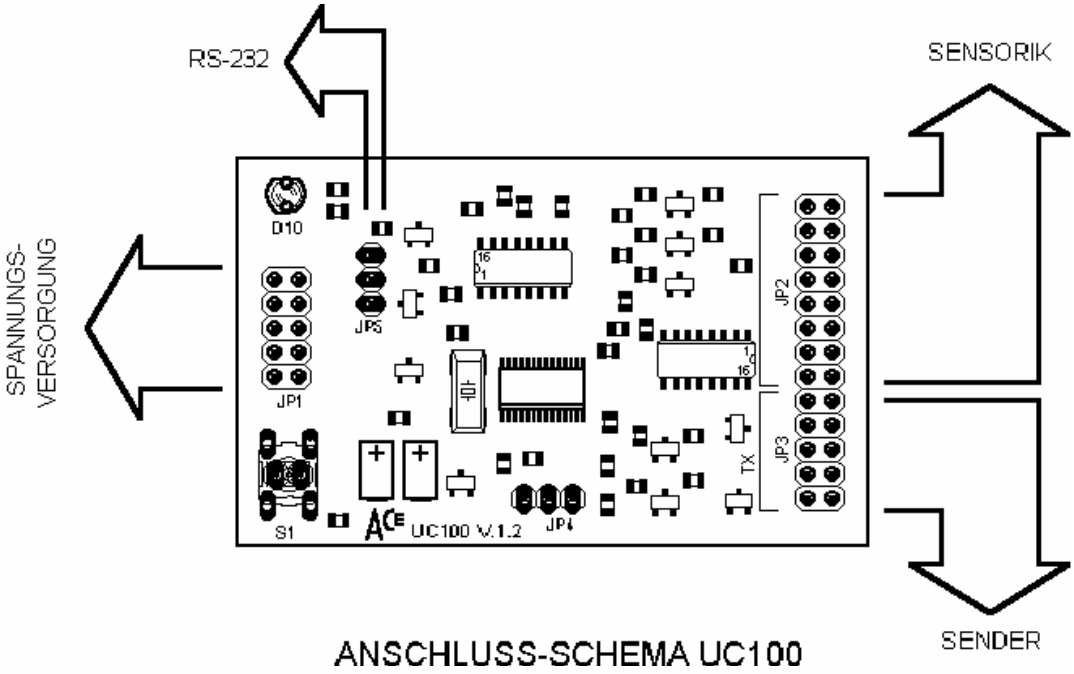


UC100 Blockdiagramm


TITLE: uc100-12
Document Number:
REV:
Date: 29.01.2003 22:24:08
Sheet: 1/1


UC100 V.1.2
JP1
JP2
JP3
JP4
JP5
JP6
S1
IC1
IC2
IC3
Q1
D10
TXD
GND
RXD
+Vs
IN
AGD
DA1
SDN
CLK
+VS

Pinbeschreibung



ANSCHLUSS-SCHEMA UC100

JP1 (Versorgungs-Anschluss):

+VS:
Versorgungsspannung für UC100, stabilisierte Gleichspannung zwischen 3.0V und 5.0V.
Anlegen der Spannung erzeugt einen Reset Impuls. Durch 5,6V Zenerdiode vor kurzzeitigen Überspannungsspitzen geschützt (Achtung: kein Vorwiderstand vor der Zenerdiode, Zerstörungsgefahr bei Überstrom!)

GND:
Allgemeine Versorgungsmasse, wird mit Minuspol der Betriebsspannungsquelle verbunden

SDN:
Shutdown Pin für zusätzliche Elektronik. Open Drain Ausgang zur Reduktion des Stromverbrauchs
während der „Sleep" – Phase (schaltet im Betrieb gegen GND), maximal +24VDC / 200mA

INT:
Umschaltsignal von Normalbetrieb in den Programmiermodus, Taster gegen GND, liegt parallel zum internen Taster S1

IND:
LED Ansteuerung für externe Anzeige. Es kann eine 2mA LED direkt gegen GND angeschlossen werden (1k Wid. intern vorh.)

TXD:
TX-Daten für RS-232 Schnittstelle, PC kann direkt angeschlossen werden, Ausgangswert der Baudrate 1200

RXD:
RX-Daten für RS-232 Schnittstelle, PC kann direkt angeschlossen werden, Ausgangswert der Baudrate 1200

JP2 (Sensorik-Anschluss):

+VS:
Versorgungsspannung des UC100 zur Verfügung als Betriebsspannung für eventuell angeschlossene Sensor-Elektronik (direkte Verbindung mit +VS auf JP1)
GND:
Versorgungsmasse für eventuell angeschlossene Sensorik (direkte Verbindung mit GND auf JP1)
AGD:
Analogmasse für eingespeiste Mess-Spannungen (direkte Verbindung mit GND auf JP1)
AD1:
Eingang in AD-Wandler (Port 17), Eingangsimpedanz statisch ca. 1M // 30pF, Vin > +VS und < 0V vermeiden
AD2:
Eingang in AD-Wandler (Port 18), Eingangsimpedanz statisch ca. 1M // 30pF, Vin > +VS und < 0V vermeiden
AD3:
Eingang in AD-Wandler (Port 19) , Eingangsimpedanz statisch ca. 1M // 30pF, Vin > +VS und < 0V vermeiden
V0:
Eingang in AD-Wandler über Multiplexer (Port 0) , Vin > +VS und < 0V vermeiden
V1:
Eingang in AD-Wandler über Multiplexer (Port 1) , Vin > +VS und < 0V vermeiden
V2:
Eingang in AD-Wandler über Multiplexer (Port 2) , Vin > +VS und < 0V vermeiden
V3:
Eingang in AD-Wandler über Multiplexer (Port 3) , Vin > +VS und < 0V vermeiden
V4:
Eingang in AD-Wandler über Multiplexer (Port 4) , Vin > +VS und < 0V vermeiden
V5:
Eingang in AD-Wandler über Multiplexer (Port 5) , Vin > +VS und < 0V vermeiden
V6:
Eingang in AD-Wandler über Multiplexer (Port 6) , Vin > +VS und < 0V vermeiden
V7:
Eingang in AD-Wandler über Multiplexer (Port 7) , Vin > +VS und < 0V vermeiden
SDN:
Shutdown Pin für zusätzliche Elektronik. Open Drain Ausgang zur Reduktion des Stromverbrauchs während der „Sleep" Phase (schaltet im Betrieb gegen GND), maximal +24VDC / 200mA
IN:
Digitales Eingangs-Steuersignal für bestimmte Software-Varianten (nicht in Version 1.00)

JP3 (TX-Anschluss):

DA0:
Analogausgang für Sendemodulation (Hauptsignal), Ausgangswid. 200 Ohm, Vmax = Vref (2,5V) oder +VS (einstellbar)
DA1:
Analogausgang für Sendemodulation (Hilfssignal), Ausgangswid. 200 Ohm, Vmax = Vref (2,5V) oder +VS (einstellbar)
AGD:
Analogmasse für Sendemodulation (direkte Verbindung mit GND auf JP1)
+VS:
Versorgungsspannung des UC100 zur Verfügung als Betriebsspannung für eventuell angeschlossenen Sender (direkte Verbindung mit +VS auf JP1)
GND:
Versorgungsmasse für eventuell angeschlossenen Sender (direkte Verbindung mit GND auf JP1)
PTT:
TX-ON Signal (Open – Drain), maximal +24VDC / 200mA

SDN:
Shutdown Pin für zusätzliche Elektronik. Open Drain Ausgang zur Reduktion des Stromverbrauchs
während der „Sleep" Phase (schaltet im Betrieb gegen GND) oder anwenderkonfigurierbar, maximal +24VDC / 200mA
DAT:
Daten – Signal des seriellen TX – Steuerbus (kann auch als digitales Ausgangssignal frei programmiert werden)
CLK:
Clock - Signal des seriellen TX – Steuerbus (kann auch als digitales Ausgangssignal frei programmiert werden)
STR:
Strobe - Signal des seriellen TX – Steuerbus (kann auch als digitales Ausgangssignal frei programmiert werden)

JP4 (Jumper für Prog-Download):

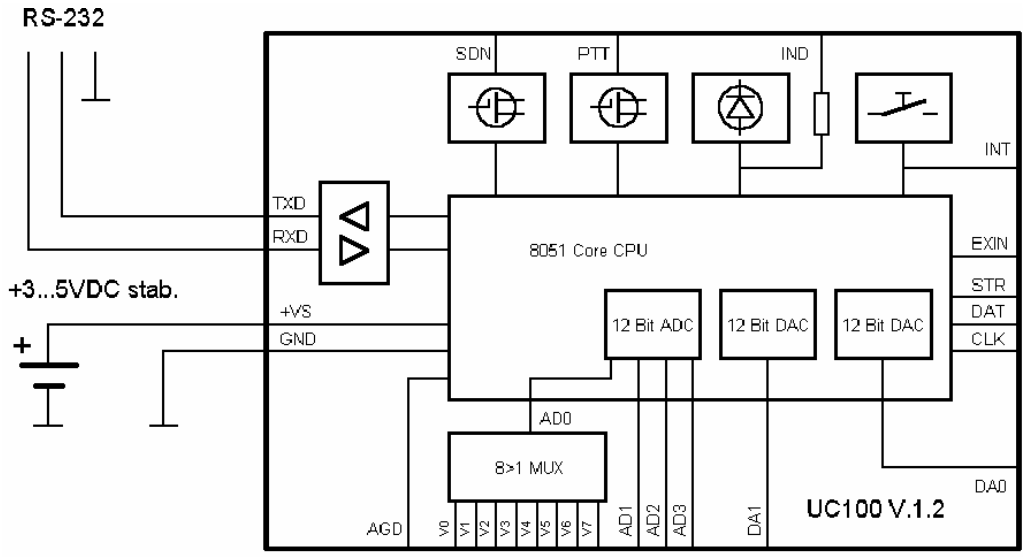
OP:
Jumper auf OP gesteckt ist notwendig für normalen Betrieb
DL:
Jumper auf DL ist zum Download eines Firmware – Updates notwendig

JP5 (zusätzlicher RS-232 Anschluss):

TXD:
TX-Daten für RS-232 Schnittstelle (wie JP1)
RXD:
RX-Daten für RS-232 Schnittstelle (wie JP1)

Anschaltungsvarianten

1. Verbindung mit dem PC

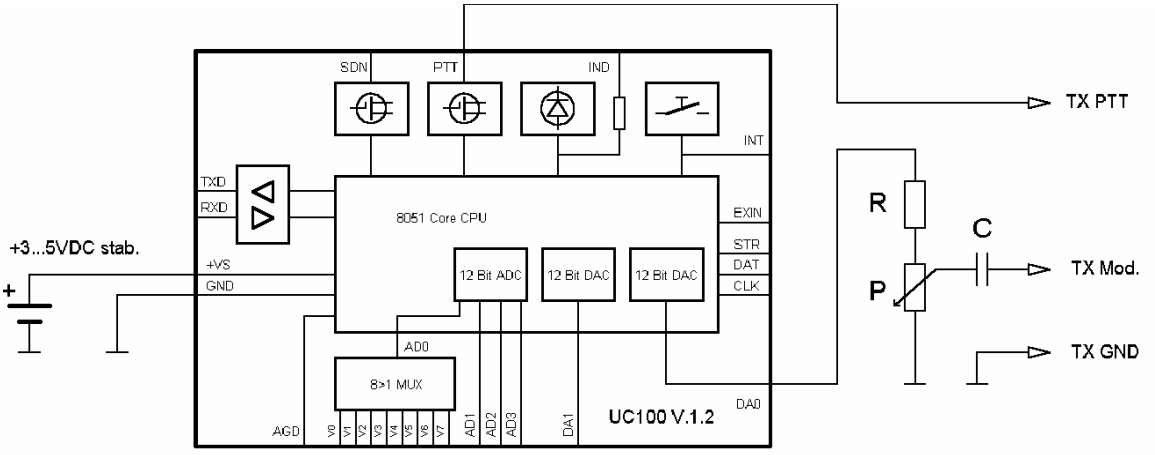


UC100 Anschaltung für Programmierung

Zur Programmierung und zum Betrieb des UC100 am PC wird eine Verbindung zwischen dem seriellen Port des Personal Computers und den TXD, RXD und GND Pins des UC100 hergestellt. Sie sollte nur so lang sein, wie zum Betrieb notwendig. Weitere Informationen zur Anschaltung befinden sich im Abschnitt „Inbetriebnahme" dieser Betriebsanleitung.

Bei Verwendung von längeren Betriebsspannungszuleitungen wird im Sinne des EMV Schutzes nahe am Board ein PI oder T-Filter zur Vermeidung von Störbeeinflussung des Controllers durch externe Quellen bzw. Emission eventueller restlicher Prozessorimpulssignale empfohlen.

2. Ansteuerung eines FM/AFSK Senders



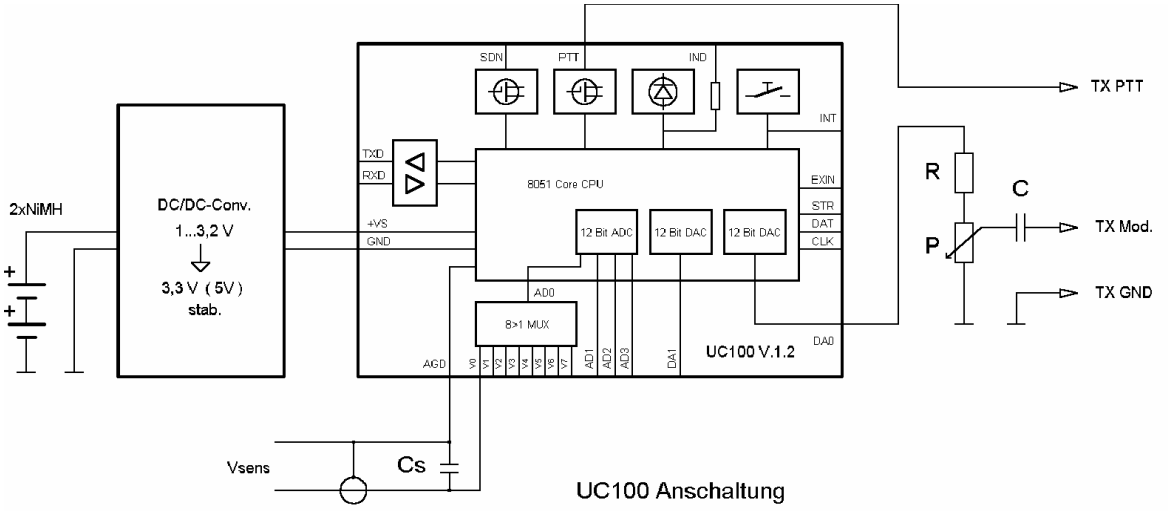
UC100 Anschaltung eines Senders (FM oder AFSK)

Der UC100 bietet die Möglichkeit, nahezu alle verfügbaren Sender oder Sendeempfänger entweder direkt oder über ein einfaches Interface anzusteuern. Ein Feldeffekttransistor führt die Sende/Empfangsumschaltung (PTT) durch, indem diese Leitung bei Senden gegen GND durchgeschaltet wird (max. 200mA).

Durch die Möglichkeit des UC100, über Software den Modulationspegel anzupassen, wäre es denkbar, Sender direkt aus dem Digital/Analog Wandler-Ausgang anzusteuern. Eine universellere Variante wird im Bild oben dargestellt. Dadurch ist es möglich, auch sehr empfindliche Modulationseingänge anzupassen, bei Beibehaltung der vollen Audioqualität, sowie DC Entkopplung zwischen UC100 und Sender (für gleichspannungsgesteuerte Modulationsvarianten wie FSK nicht geeignet!).
Der Gesamtwiderstand von R und P sollte sich dabei zwischen 500 Ohm und 10 kOhm bewegen. Der Koppelkondensator C muss auf den Eingangswiderstand des Sendemodulators abgestimmt werden. In der Praxis sollten Werte zwischen 100 nF und 1 µF gute Ergebnisse ermöglichen.

Bei längeren Verbindungsleitungen zwischen UC100 und Sender (>5cm) wird empfohlen, geschirmte Kabel zu verwenden.

3. Ansteuerung eines FM/AFSK Senders, Batterieversorgung und Sensorbeschaltung



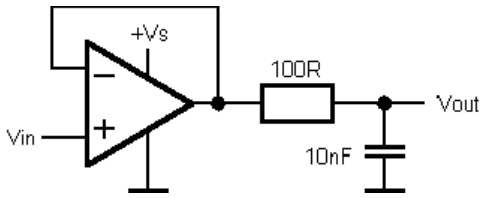
UC100 Anschaltung

Batt.-Betrieb, Sendermod. FM oder AFSK, 1 externe Sensorleitung

Es sollte vermieden werden, den UC100 direkt aus Batterien oder Akkus zu versorgen. Die optimale Messgenauigkeit wird erreicht, wenn sich die Versorgungsspannung während des Betriebs in einer Toleranz von +-5 % eines Wertes innerhalb des spezifizierten Bereiches bewegt (3…5 VDC). Daher wird empfohlen, bei Batteriebetrieb einen DC/DC-Wandler mit stabilisierter Ausgangsspannung einzusetzen.

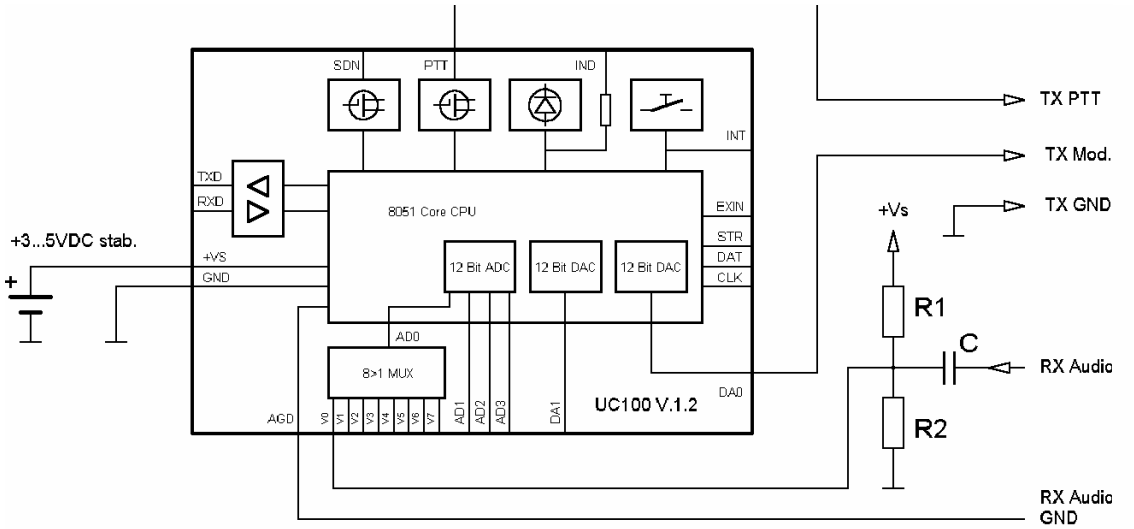
Bei der Beschaltung mit externen Sensoren muss zur Erreichung der besten Messergebnisse auf 2 wichtige Kriterien geachtet werden. Die Sensorausgangsimpedanz sollte so gering wie möglich sein und längere Sensorleitungen müssen geschirmt werden. Optimal wäre der Einsatz eines Pufferverstärkers (Rail to Rail OPV) zwischen Sensorausgang und A/D-Wandler Eingang (siehe unten).
Zur Stabilisierung des Messwertes kann ein Kondensator (Cs) direkt an den Messeingang am UC100 gesetzt werden. Sein Wert muss auf die Sensorimpedanz und die maximale Signalfrequenz abgestimmt sein, zu kleine Werte von Cs haben zu wenig Effekt auf die Messwertstabilisierung, zu große Werte reduzieren die maximal mögliche Messfrequenz.

Für die Anpassung des Modulationssignals gilt das gleiche wie in Punkt 2.



Vorschlag für Pufferverstärker

4. Ansteuerung eines FSK Senders und Auswertung einer Empfänger NF

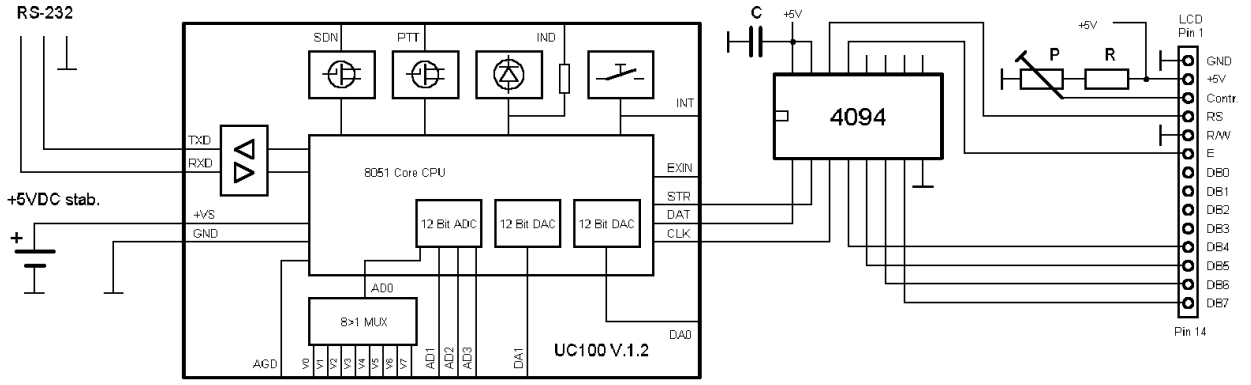


UC100 Anschaltung eines Senders (FSK) und Empfängers

Sollte z.B. die Betriebsart **h96** verwendet werden (FSK, 9600 Bd.), steht das Modulationssignal als Spannung zwischen 0 und 2,5VDC an DA0 zur Verfügung. Dieses Rechtecksignal kann direkt zur Ansteuerung des Modulators verwendet werden.

Der UC100 kann auch durch das Anlegen eines Tonbursts gesteuert werden (Befehl **wait <frequ>**). Dazu wird das auszuwertende NF Signal an einen Eingangsport (hier: Port 0) gelegt. Es ist günstig, die maximal mögliche Aussteuerbarkeit durch den Einsatz eines Spannungsteilers zu fixieren. Gleichspannungspegel von 1 bis 1,5 VDC am Port erlauben ein problemloses Verarbeiten von NF Eingangspegeln in der Größenordnung von ca. 30 bis 500 mV RMS.
Der Spannungsteiler R1/R2 sollte einen Serienwiderstand von 10 kOhm bis 500 kOhm aufweisen. Da die NF Ausgangsstufe des Empfängers durch den Spannungsteiler belastet wird und auch der Wert des Koppelkondensators C davon abhängt, sollte der Gesamtwiderstand des Teilers als Parallelschaltung R1//R2 keine Werte unter 1 kOhm erreichen. Bei ca. 10 kOhm kann der Wert des Kondensators C 100 nF betragen, bei Parallelwiderstandswerten von unter 10 kOhm sollte auf 1 µF erhöht werden.
Zu hohe Eingangspegel werden durch die im UC100 vorhandene Überspannungsbegrenzung limitiert, man muss allerdings in diesem Falle damit rechnen, dass die angelegte NF Spannung nicht mehr korrekt decodiert wird und damit Auswertefehler auftreten können.

5. Ansteuerung eines LC Displays



UC100 Ansteuerung eines LC Displays

Durch Verwendung eines zusätzlichen Schieberegisters (4094) können Standard LCDs direkt aus dem UC100 heraus angesteuert werden (Programmierbefehl **lcd**). Besonders geeignet hierfür sind 2-zeilige Displays mit je 16 bis 40 Zeichen.

Das Schieberegister muss in diesem Fall keine besonderen Eigenschaften aufweisen, es können alle verfügbaren CMOS Typen eingesetzt werden. Wichtig ist die Funktion des Kondensators C (100n) für einen störungsfreien Betrieb.
Der Kontrast lässt sich über P (1k) und R (2k7) stufenlos einstellen.

Durch Verwendung des Befehls **shft** können die 8 Ausgänge des Schieberegisters auch individuell programmiert werden, damit ist es möglich, weitere externe Steueraufgaben durchzuführen.

Programmierbefehle

Konfigurationsbefehl

cfg Dieser Befehl kann zu Beginn des Userprogramms bestimmte voreingestellte Werte verändern. Er braucht nicht verwendet zu werden, wenn die Standardeinstellung für die Anwendung verwendbar ist.

Ausgangswerte:

- Schaltsignal SDN kommt bei Sleep Mode
- LED leuchtet wenn PTT schaltet
- Programmiermode auch über Terminalbefehl einschaltbar
- max. Ausgangsspannung der D/A-Wandler = Vref (2,5V)

Syntax: **cfg** <**Binärwert**> z.B. **cfg 1100000111b** (10 Bit, davon 5 verwendet, restliche 5 Bits = 0)

Bit 0 (**000000000Xb**) :	0: SDN Signal im Sleep Mode	1: SDN durch Userprogramm bestimmt
Bit 1 (**00000000X0b**) :	0: LED leuchtet bei PTT	1: LED durch Userprogramm bestimmt
Bit 2 (**0000000X00b**) :	0: Prog.-Mode zus. über Terminal	1: Prog.-Mode nur über Taster S1
Bit 3…7 (**00XXXXX000b**) :	nicht verwendet, bleiben auf 0	
Bit 8 (**0X00000000b**) :	0: DAC0 max. bis Vref (2,5V)	1: DAC0 max. bis +Vs
Bit 9 (**X000000000b**) :	0: DAC1 max. bis Vref (2,5V)	1: DAC1 max. bis +Vs

Schreib-/ Lese – Variable

a … **j** Universell einsetzbare Variable SYNTAX : **a=<x>** (16 Bit) oder Verknüpfung aus zwei 16 Bit Variablen oder Konstanten (z.B. **a=b+c** oder **e=e+1**)

p0, **p1**, **p2**, **p3** Portadressen wie 8051 – Standard, hier p1 und p3 extern vorhanden
SYNTAX: wie bei den universellen Variablen
Achtung: Vorgang kann unzulässige Zustände an den Ports zur Folge haben! Nur verwenden, wer genau mit der Hardware vertraut ist und über fundierte Kenntnisse der 8051 Programmierung verfügt

x, **xl** String Variable, Länge der Stringvariablen, siehe auch Befehle **rdln, rs** und **pr**, Lese - Variable **xm** bzw. Beispielprogramme

lev Änderung der Modulationsamplitude SYNTAX: **lev=0** bis **lev=255** für 0…2,5V

spd Änderung der Gebegeschwindigkeit bei CW oder Dauer des Beep – Signals
(siehe auch **cwa1, cwf1, cwf2** und **beep**) , SYNTAX: **spd=0** bis **spd=255**

dc Gleichspannungsanteil am Modulationssignal SYNTAX: **dc=0** bis **dc=255** für 0..2,5V

f0 Fixiert eine Audiofrequenz zur Ausgabe an DA0, Schrittweite=0,2Hz, z.B. **f0=5000** für eine Audiofrequenz von 1 kHz, Ausgabe erfolgt über den **tone** Befehl (gilt nicht für AFSK)

f1 Definiert die Umtastfrequenz bei AFSK

txd TX – Delay für die Betriebsarten PR (AX-25), PSK und DTMF, allerdings unterschiedliche Einheiten für jede Betriebsart SYNTAX: **txd=0** bis **txd=255**

tail Nachlaufzeit für die Betriebsart PR (AX-25) SYNTAX: **tail=0** bis **tail=255**

devi Spannungshub bei analoger FSK (von Null Volt aufwärts) SYNTAX: **devi=0** bis **devi=255**

port Selektiert den gewünschten Analogport, 0…7 = ADC0 über den Multiplexer, 8=Chip-Temp-Sensor, 17…19=ADC1..3 direkt. SYNTAX: **port=0** bis **port=8** bzw. **port=17** bis **port=19**

rs Letztes Byte der Daten an der seriellen Schnittstelle. Kann auch verwendet werden, um Bytes zu senden

th0 Samplingrate für Modulation, verändert alle Gebegeschwindigkeiten und Frequenzen

shft Sendet Daten an ein externes Schieberegister über den seriellen Port (CLK, DAT, STR)
SYNTAX: **shft=0** bis **shft=255** bzw. **shft=00000000b** bis **shft=11111111b**
Ansteuerung des Schieberegisters siehe unter Anschaltungsvarianten, 5. Anschaltung eines LC Displays

Lese – Variable

adc Stellt den Spannungswert am selektierten Port dar, kann beliebig weiterverarbeitet werden
Ausgabewert (dezimal) liegt zwischen 0 und 4095 für Spannungen zwischen 0 und 2,5V DC.

temp Chiptemperatur 3-stellig auf 0.1 Grad C aufgelöst, ohne Komma (ist der bereits intern umgerechnete Messwert an Port 8) d.h. zur korrekten Ausgabe dieses Wertes muss eine Kommastelle gesetzt werden (z.B. **pr %d31:temp** gibt die Chiptemperatur über die RS-232 Schnittstelle folgendermaßen aus: **21.3**) Toleranzbereich ca. +- 1,5°C, abhängig von der momentanen Stromaufnahme der CPU.

xm maximale Größe der Stringvariablen (Größe des FIFO Puffers)

Schreib – Variable

dac0 Gibt am Ausgang DA0 eine DC zwischen 0 und 2,5V aus
SYNTAX: **dac0=0** bis **dac0=4095** für 0 bis 2,5V

dac1 Gibt am Ausgang DA1 eine DC zwischen 0 und 2,5V aus
SYNTAX: **dac1=0** bis **dac1=4095** für 0 bis 2,5V

Procedures

pr Gibt eine Datenzeile über die RS-232 Schnittstelle aus
SYNTAX für Fixdaten: **pr <x>** (x=auszugebende Information)
SYNTAX für Variable: **pr %d<y>:<x>** (d=dez, y=Ausgabeformat, x=auszugebende Variable)
Ausgabeformatierung :
Immer beginnend mit dem Formatierungszeichen **%**
Danach **d** (dezimal), **h** (hex), **$** (Stringvariable x) oder **n** (neue Zeile)
Bei Dezimalausgabe **d** folgt die Angabe der benötigten Stellen und bei Bedarf die Kommaposition
z.B. **%d31:temp** schreibt die Chiptemperatur 3-stellig und setzt das Komma auf 0,1 Grad (xx,x)
bei **%d51:temp** bleibt die Anzeige gleich, jedoch werden zusätzlich 2 Leerstellen vorne mit eingeschoben
%konstante und **%variable** geben die betreffenden Werte als ASCII Byte aus

rdln 0 Startet unlimitierten Einlesevorgang aus der RS-232 Schnittstelle (FIFO), kein EOL, stoppt durch **rdln -1**

rdln 1 Startet Einlesevorgang aus der RS-232 Schnittstelle in den String **x** mit der Länge **xl**, Ende durch CR (13)

rdln -1 Stoppt den Einlesevorgang, der mit **rdln 0** gestartet wurde

rdln 2 .. 65535 Wie **rdln 1** jedoch Ende durch CR (13) oder timeout in ms
Achtung: EOL (0Dh) im String mitgespeichert. Löschen durch **xl=xl-1**
Im Falle von timeout wird ein String mit **xl=0** zurückgesendet

rsbd Einstellung der RS-232 Baudrate. Ausgangswert ist 1200Bd
SYNTAX: **rsbd=<x>** (x=Baudrate)

cwa1 Sendet Fixtext oder Variable in Morsetelegrafie (Tastung des PTT – Signals). Gebegeschwindigkeit ist ca. 60 BpM, kann mit dem Befehl **spd** verändert werden.
(**spd=70** für ca. 45 BpM, 60 für ca. 55 BpM, 50 für ca. 65 BpM, 40 für ca. 80 BpM, 30 für ca. 105 BpM)
SYNTAX bei Fixtext: **cwa1 <x>** (x kann Buchstaben, Zahlen oder Sonderzeichen enthalten)
SYNTAX bei Variablen: **cwa1 %d<y>:<x>** (d=dez, y=Ausgabeformat, x=Variable oder Port)
Genauere Angaben zum Ausgabeformat unter Befehl **pr,** siehe auch Befehle **spd** und **init**

cwf1 Sendet Fixtext oder Variable in Morsetelegrafie (Tastung der DC - Spannung auf DA1).
Gebegeschwindigkeit ist ca. 60 BpM, kann mit dem Befehl **spd** verändert werden. Ansonsten wie **cwa1**.

cwf2 Sendet Fixtext oder Variable in Morsetelegrafie (Tontastung mit Frequenz **f0**). Gebegeschwindigkeit ist ca. 60 Bpm, kann mit dem Befehl **spd** verändert werden. Ansonsten wie **cwa1**.

h12 Sendet Fixtext oder Variable im AX-25 Protokoll (Packet Radio), AFSK, 1200 Bps
SYNTAX bei Fixtext: **h12 :<X> <Y>:<z>** (X=Empfänger-Call, Y=Sender-Call, z=beliebiger Text)
SYNTAX bei Variablen: **h12 :<X> <Y>:%d<z>:<a>** (X,Y wie oben, d=dez, z=Ausgabeformat, a=Variable) Beispiel : Ausgabe der Chiptemperatur **h12 :RXCALL TXCALL:%d31:temp**
Eingabe von bis zu 8 Digipeatern möglich, z.B. **h12 :RXCALL TXCALL DIGI1 DIGI2:test**
(auf Großschreibung der Calls achten!).

h24 Sendet Fixtext oder Variable im AX-25 Protokoll, AFSK, 2400 Bps, SYNTAX wie bei **h12**

h96 Sendet Fixtext oder Variable im AX-25 Protokoll, FSK, 9600 Bps, SYNTAX wie bei **h12**

dtmf Sendet Fixzahlenfolgen oder Variable in DTMF codierten Tönen aus
SYNTAX bei Fixzahlen: **dtmf <x>** (x = beliebige Zahlenfolge)
SYNTAX bei Variablen: **dtmf %<x>** (x = definierte Variable, wie bei Befehl **pr**)

psk Sendet Fixtext oder Variable in PSK31
SYNTAX bei Fixtext: **psk <x>** (x = beliebige Text- oder Zahlenfolge)
SYNTAX bei Variablen: **psk %<x>** (x = definierte Variable, wie bei Befehl **pr**)

beep Sendet einen kurzen Ton SYNTAX: **beep <x>** (x=Tonfrequenz in Hz), Einstellung der Dauer über den Befehl **spd** in 10ms Schritten, z.B. **init beep** gefolgt von **spd=50** für 500ms, danach z.B. **beep 1000**

tone Sendet dauernd die Frequenz **f0**, die auch als Variable definiert sein kann. Funktion wie V/f - Converter
SYNTAX: **tone** (in einer Programmzeile ohne Zusatz, danach ein Label und eine Schleife zur Abfrage der gewünschten Variable. Siehe Beispielprogramme). **tx 0** beendet die Ausgabe. Bei Verwendung der Befehle **sls** oder **slms** bleibt die zuletzt ausgegebene Tonfrequenz für die gewünschte Dauer erhalten

f3 Gibt den an einem ausgewählten Port anstehenden Spannungswert 1:1 an DA0 wieder aus

init Setzt Ausgangswerte (Parameter) für die angegebene Modulation, gilt solange, bis eine andere Modulation aufgerufen wird SYNTAX: **init <x>** (z.B. **init cwa1**)

sls Fügt eine Wartezeit in den Programmablauf ein, UC100 geht in den „Sleep" – Mode
SYNTAX: **sls <x>** (x = Wartezeit in Sekunden, nur ganze Sekunden möglich, z.B. **sls 2**)

slms Wie Befehl **sls**, jedoch Eingabe von Millisekunden

tx 1 Schaltet den Sender ein

tx 0 Schaltet den Sender aus

led 1 Schaltet die LED ein, z.B. für Signalisierungszwecke (siehe auch Befehl **cfg**)

led 0 Schaltet die LED aus (siehe auch Befehl **cfg**)

shdn 1 Schaltet den SDN Pin durch (siehe auch Befehl **cfg**)

shdn 0 Schaltet den SDN Pin wieder aus (siehe auch Befehl **cfg**)

wait Stoppt den Programmablauf, Programm wird erst bei erkennen eines Steuerimpulses am angegebenen Port fortgesetzt. Start durch positive oder negative Flanke kann ausgewählt werden (Triggerschwelle 1,25V).
SYNTAX bei steigender Flanke **wait <x>** (x = Portnummer 0...7, 17..19, z.B. **wait 0**)
SYNTAX bei fallender Flanke **wait <-x>** (x = Portnummer 0...7, 17..19, z.B. **wait -1**, bei Port 0 nicht möglich, dort nur positive Flanken)
Start durch Tonburst: **wait <frequ>** (frequ = auszuwertende Tonfrequenz in Hz), das Programm stoppt, bis am ausgewählten Port die gewünschte Tonfrequenz erscheint und wird danach fortgesetzt.

lcd Schreibt Fixtexte oder Variable auf Standard-LCDs
Für diese Funktion ist ein einfaches Hardware Interface in Form eines Schieberegisters notwendig, um die Daten des seriellen Busses in parallele Informationen umzuwandeln (siehe „Anschaltungsvarianten").

Prinzipiell lassen sich die meisten LCD Anzeigen über die verfügbaren Möglichkeiten ansteuern, am günstigsten ist jedoch die Verwendung von zweizeiligen Anzeigen mit je 16 bis 40 Zeichen.

SYNTAX: **lcd** <**string**> zum Schreiben an die erste Position der ersten Zeile (z.B. **lcd Hallo**)
lcd %128+x <**string**> zum Schreiben an eine beliebige Position in der ersten Zeile
z.B. **lcd %130 Hallo** schreibt „Hallo" an die 3. Position der ersten Zeile
lcd %192+x <**string**> zum Schreiben an eine beliebige Position in der zweiten Zeile
SYNTAX für Variable: z.B. Temperatur in Zeile 2, Spalte 3 : **lcd %194 %d31:temp C**

Steuerbefehle: Dez. 0…15 je Steuer-Nippel können an das angeschlossene Display gesendet werden.
z.B. Schirm löschen: **lcd %0 %1**
nach manchen Kommandos kann es notwendig sein, eine Pause von mehreren Millisekunden einzuschalten, bis wieder etwas ausgegeben werden kann (**slms**)
Initialisierung: Initialisierungsstring wird automatisch beim ersten Aufruf des **lcd** Befehls nach dem Einschalten übertragen.

app Hängt den folgenden String an den bestehenden String **x** bei Position **xl** (**xl** wird dabei um die Stringlänge inkrementiert)

Fügt eine Kommentarzeile ein, gilt nicht als Programmschritt, SYNTAX: **#<text>** (kein Leerzeichen)
Benötigt keinen Speicherplatz im Chip

Mathematische / Logische 2 Parameter Operanden

+	Addiert 2 Konstanten oder Variable (16 Bit), z.B. **a=b+1**
-	Subtrahiert 2 Konstanten oder Variable (16 Bit), z.B. **a=b-1**
*****	Multipliziert 2 Konstanten oder Variable (je 16 Bit, Ergebnis auf 16 Bit begrenzt), z.B. **a=adc*2**
/	Dividiert 2 Konstanten oder Variable (16 Bit), z.B. **b=a/3**
~	Rechnen mit Bruchzahlfaktor, dient zur vereinfachten Eingabe folgender Operationen: **a=b*~c** entspricht **a=b*(c/65536)** und **a=b~/c** entspricht **a=(b*65536)/c** für positive Zahlen und b<c
#	Divisionsrest, z.B. **c=a#b** (16 Bit)
&	Logische UND Verknüpfung, z.B. **a=b&c** (16 Bit)
|	Logische ODER Verknüpfung, z.B. **a=b|c** (16 Bit)
%	Logische Exklusiv ODER Verknüpfung, z.B. **a=b%c** (16 Bit)
<	Entscheidungskriterium KLEINER z.B. **if a<b** (16 Bit)
>	Entscheidungskriterium GRÖSSER z.B. **if a>b** (16 Bit)
=	Entscheidungskriterium GLEICH z.B. **if a=b** (16 Bit)
<>	Entscheidungskriterium UNGLEICH z.B. **if a<>b** (16 Bit)
>=	Entscheidungskriterium GRÖSSER-GLEICH z.B. **if a>=b** (16 Bit)
<=	Entscheidungskriterium KLEINER-GLEICH z.B. **if a<=b** (16 Bit)
^0	Bit 0 der rechten Variablen, Bits 1..15 der linken Variablen
^1	Bit 1 der rechten Variablen, Bits 0, 2..15 der linken Variablen
^2	Bit 2 der rechten Variablen, Bits 0..1, 3..15 der linken Variablen
^3	Bit 3 der rechten Variablen, Bits 0..2, 4..15 der linken Variablen
^4	Bit 4 der rechten Variablen, Bits 0..3, 5..15 der linken Variablen
^5	Bit 5 der rechten Variablen, Bits 0..4, 6..15 der linken Variablen
^6	Bit 6 der rechten Variablen, Bits 0..5, 7..15 der linken Variablen
^7	Bit 7 der rechten Variablen, Bits 0..6, 8..15 der linken Variablen
+^	Setzt das Bit n der linken Variablen (z.B. **p3=p3+^a**)
-^	Löscht das Bit n der linken Variablen (z.B. **p3=p3-^a**)

Programm Befehle

if	Wenn folgender Ausdruck nicht zutrifft, nächste Zeile überspringen (z.B. **if a=10**)
goto	Sprungbefehl zu Label Nr. 1…255 (z.B. **goto 10**)
loop	Sprung zurück zum Programmstart
halt	Stoppt das laufende Programm, UC100 geht in den „Sleep-Mode"
end	Letzte Zeile des Userprogramms im BURN - Mode
<label>**:**	Definiert den Beginn eines bestimmten Programmabschnitts für Sprungbefehl SYNTAX **<x>:** INFO: x = Beliebige Zahl zwischen 1 und 255 (z.B. **10:**) Label benötigt eine Programmzeile, nächster Programmschritt in neue Zeile

Interaktive Befehle

list	Listet das Userprogramm aus dem RAM auf dem Bildschirm
run	Startet das Userprogramm im RAM
line	Eingabe einer Programmzeile und sofortige Ausführung
test	Syntax-Test einer Programmzeile, Echo ein für Tastatureingabe
test&	Syntax Test einer Programmzeile, Echo aus für Eingabe aus Datei
burn	Speichern der folgenden Programmzeilen in das EEPROM, Echo ein für Tastatureingabe. Programmeingabe durch den Befehl **end** abschließen.
burn&	Speichern der folgenden Programmzeilen in das EEPROM, Echo aus für Eingabe aus Datei. Programmeingabe durch den Befehl **end** abschließen.
ram	Löscht die Daten im RAM und ermöglicht die Eingabe eines neuen Userprogramms, erscheint nach Betätigen des Tasters S1 (INT). Echo ein für Tastatureingabe.
ram&	Löscht die Daten im RAM und ermöglicht die Eingabe eines neuen Userprogramms, erscheint nach Betätigen des Tasters S1 (INT). Echo aus für Eingabe aus Datei.