

Ministerium Umbenennung

genehmigt

geschrieben von Andreas Potthoff | 3. Mai 2023

Westfälische Wilhelms-Universität heißt ab dem 1. Oktober Universität Münster

Das Ministerium für Kultur und Wissenschaft des Landes Nordrhein-Westfalen hat die Änderung des Namens Westfälische Wilhelms-Universität (WWU) Münster in Universität Münster genehmigt. Die Verwendung des neuen Namens tritt zum 1. Oktober 2023 in Kraft. Die Universität wird heute ihre aktualisierte Verfassung auf der Homepage veröffentlichen. In den kommenden Wochen trifft die Universität Vorbereitungen, um alle nötigen Änderungen fristgerecht vorzunehmen, wie beispielsweise an Logos, Dienstsiegeln und weiteren offiziellen Dokumenten.

Zum Hintergrund

Der Senat entschied am 5. April, dass die WWU ihren Namensgeber Wilhelm II. aus ihrem Namen streicht und künftig Universität Münster heißt. Dem Votum des Senats war ein mehrjähriger Prozess vorausgegangen, der im Jahr 2018 mit einer Initiative der Studierenden im Senat begann. Der Senat forderte seinerzeit die Universitätsleitung auf, ein „Konzept zu einem historisch verantwortlichen Umgang der WWU mit Wilhelm II.“ zu erstellen. Eine vom Senat eingesetzte Arbeitsgruppe unter der Leitung des WWU-Historikers Prof. Dr. Olaf Blaschke erarbeitete 2020 ein differenziertes Bild.

Um die Studierenden, die Beschäftigten und die Öffentlichkeit möglichst intensiv an der Diskussion teilhaben zu lassen,

initiierte das Rektorat in Abstimmung mit dem Senat im Anschluss das Projekt „Zur Sache WWU“, das im Januar 2021 startete. Ein Team unter der Leitung von Dr. Eckhard Kluth, Leiter der Zentralen Kustodie, organisierte daraufhin mehrere Kommunikations- und Veranstaltungsformate. Dazu gehörten beispielsweise Podiumsdiskussionen, eine Ausstellung, ein Kurzfilm sowie eine Radiosendung. Über die Projekt-Webseite konnten sich alle Interessierten über den Prozess informieren und sich am Austausch mit Kommentaren und Hinweisen beteiligen.

Links:

- FAQs Namensänderung
https://www.uni-muenster.de/kommunikation/FAQ_umbenennung.html
 - News der Universität Münster vom 5. April 2023
<https://www.uni-muenster.de/news/view.php?cmdid=13244>
-

Quelle: Pressemitteilung / Pressestelle der Universität Münster (upm)

CBM DOS: Floppy Fehlermeldungen

geschrieben von Andreas Potthoff | 3. Mai 2023
Die Fehlermeldungen der CBM Laufwerke für:

- CBM 1540, CBM 1541, CBM 1541-II, CBM 1541 C, CBM 1551, CBM 1570, CBM 1571, CBM 1581
- CBM 2020, CBM 2031, CBM 2040, CBM 3040, CBM 4031, CBM 4040
- CBM 8050, CBM 8060, CBM 8061, CBM 8062, CBM 8250, CBM 8280
- Enhancer 2000, FD-200, OC-118N, SFD-1001

Inhaltsverzeichnis

- Westfälische Wilhelms-Universität heißt ab dem 1. Oktober Universität Münster
 - Links:

Fehlererkennung

Am Blinken der Kontroll-LED (z.B. bei einem 1541-Laufwerk) kann man erkennen, dass bei einer Diskettenoperation eine Fehlersituation aufgetreten ist. Das CBM-DOS gibt eine entsprechende Statusmeldung über den Fehlerkanal aus. Beim BASIC werden Fehlermeldungen direkt auf dem Bildschirm ausgegeben. Leider ist die Anzeige von Laufwerksfehlern nur über das Auslesen des Fehlerkanals möglich.

Auslesen des Fehlerkanals

BASIC-Programm

Da der INPUT#-Befehl nur innerhalb eines Programms funktioniert ist ein kleines Programm zum Auslesen des Fehlerkanals nötig.

```
10 OPEN 1,8,15
```

```
20 INPUT# 1,A,B$,C,D
30 PRINT A,B$,C,D
40 CLOSE 1
```

Legende:

A : Nummer des Fehlers
B\$: Fehlerbezeichnung im Klartext
C : Track (Spur)
D : Sektor (Block)

Direktmodus

Man kann den Fehlerkanal auch ohne ein BASIC-Programm im Direktmodus auslesen.

Option A

```
OPEN 1,8,15:FOR I=0 TO 2:POKE 58,1:GET#1,F$:PRINT F$;:I=255
AND ST:NEXT:CLOSE 1
```

Option B

```
OPEN 1,8,15:FOR I=1 TO 40:POKE 781,1:SYS 65478:SYS 65487:SYS
65490:SYS 65484:IF ST=0 THEN NEXT:CLOSE 1
```

Tabelle CBM Floppy-Fehlermeldungen (deutsch)

Fehlermeldung	Beschreibung
00,OK,00,00	OK, es liegt kein Fehler vor.
01,FILES SCRATCHED,XX,00	Kein Fehler. Rückmeldung nach SCRATCH-Befehl, wobei XX die Anzahl der gelöschten Dateien darstellt.

Fehlermeldung	Beschreibung
20, READ ERROR, TT, SS	LESE-FEHLER: Blockheader nicht gefunden - Der Festplattencontroller kann den Header des angeforderten Datenblocks nicht finden. Verursacht durch eine unzulässige Blocknummer oder der Header wurde zerstört.
21, READ ERROR, TT, SS	LESE-FEHLER: Keine SYNC-Markierung - Der Laufwerkscontroller kann keine Sync-Markierung auf dem gewünschten Track erkennen. Verursacht durch Fehlausrichtung des Lese-/Schreibkopfes, keine Diskette vorhanden oder unformatiert oder falsch sitzende Diskette. Kann auch auf einen Hardwarefehler hinweisen.
22, READ ERROR, TT, SS	LESE-FEHLER: Datenblock nicht vorhanden - Der Laufwerkscontroller wurde aufgefordert, einen Datenblock zu lesen oder zu überprüfen der nicht ordnungsgemäß geschrieben wurde. Diese Fehlermeldung tritt in Verbindung mit den BLOCK-Befehlen auf und zeigt eine unzulässige Spur- und/oder Blockanforderung an.
23, READ ERROR, TT, SS	LESE-FEHLER: Prüfsummenfehler im Datenblock - Diese Fehlermeldung zeigt an, dass in einem oder mehreren der Datenbytes ein Fehler vorliegt. Die Daten wurden in den DOS-Speicher eingelesen, aber die Prüfsumme der Daten ist falsch. Diese Meldung kann auch auf Erdungsprobleme hinweisen.

Fehlermeldung	Beschreibung
24, READ ERROR, TT, SS	LESE-FEHLER: Byte-Dekodierungsfehler - Die Daten oder der Header wurden in den DOS-Speicher gelesen, aber es wurde ein Hardwarefehler wegen einen ungültigen Bitmuster im Datenbyte erzeugt. Diese Meldung kann auch auf Erdungsprobleme hinweisen.
25, WRITE ERROR, TT, SS	SCHREIB-FEHLER: Schreib-Verifizierungsfehler - Diese Nachricht wird generiert, wenn der Laufwerkscontroller eine Diskrepanz zwischen den geschriebenen Daten und den Daten im DOS-Speicher erkennt.
26, WRITE PROTECT ON, TT, SS	SCHREIBSCHUTZ EIN - Diese Meldung wird generiert, wenn der Laufwerkscontroller aufgefordert wurde einen Datenblock zu Schreiben während der Schreibschutz aktiviert ist. Normalerweise wird dies durch die Verwendung einer Diskette verursacht die mit einer Schreibschutzlasche über der Kerbe versehen ist.
27, READ ERROR, TT, SS	LESE-FEHLER: Prüfsummenfehler im Header - Der Laufwerkscontroller hat einen Fehler im Header des angeforderten Datenblocks festgestellt. Der Block wurde nicht in den DOS-Speicher eingelesen. Diese Meldung kann auch auf Erdungsprobleme hinweisen.

Fehlermeldung	Beschreibung
28,WRITE ERROR,TT,SS	SCHREIB-FEHLER: Langer Datenblock - Der Laufwerkscontroller versucht, die SYNC-Markierung des nächsten Headers nach dem Schreiben eines Datenblocks zu lesen. Wenn die SYNC-Markierung nicht innerhalb einer vorgegebenen Zeit erscheint, wird diese Fehlermeldung generiert. Der Fehler wird durch ein falsches Diskettenformat verursacht (die Daten erstrecken sich in den nächsten Block) oder durch Hardwarefehler.
29,DISK ID MISMATCH,TT,SS	DISK ID NICHTÜBEREINSTIMMUNG: - Diese Nachricht wird generiert, wenn der Laufwerkscontroller aufgefordert wurde, auf eine nicht initialisierte Diskette zuzugreifen. Die Meldung kann auch auftreten, wenn eine Diskette einen fehlerhaften Header hat.
30,SYNTAX ERROR,00,00	SYNTAX-FEHLER: Allgemeine Syntax - Das DOS kann den Befehl der an den Befehlskanal gesendet wurde nicht interpretieren. Typischerweise wird dies durch eine unzulässige Anzahl von Dateinamen oder Mustern die illegal verwendet werden ausgelöst. Beispiel: Beim COPY-Befehl werden zwei Dateinamen auf der linken Seite verwendet.
31,SYNTAX ERROR,00,00	SYNTAX-FEHLER: Ungültiger Befehl - Das DOS erkennt den Befehl nicht. Der Befehl muss an erster Stelle stehen.
32,SYNTAX ERROR,00,00	SYNTAX-FEHLER: Ungültiger Befehl - Der gesendete Befehl ist länger als 40 Zeichen.
33,SYNTAX ERROR,00,00	SYNTAX-FEHLER: Ungültiger Dateiname - Musterabgleich der beim Befehl OPEN-oder SAVE-Befehl verwendet wird ist ungültig.

Fehlermeldung	Beschreibung
34, SYNTAX ERROR, 00, 00	SYNTAX-FEHLER: Keine Datei angegeben - Der Dateiname wurde bei einem Befehl weggelassen oder das DOS erkennt es nicht als solches. Typischerweise wurde ein Doppelpunkt (:) bei dem Befehl weggelassen.
39, SYNTAX ERROR, 00, 00	SYNTAX-FEHLER: - Ungültiger Befehl - Dieser Fehler kann auftreten, wenn der Befehl an den Befehlskanal (Sekundäradresse 15) vom DOS nicht erkannt wird.
50, RECORD NOT PRESENT, 00, 00	EINTRAG NICHT VORHANDEN: - Ergebnis des Lesens der Diskette über den letzten Datensatz hinaus durch INPUT#- oder GET#-Befehle. Diese Meldung wird auch dann erscheinen wenn die Positionierung auf einen Datensatz bei einer relativen Datei über das Dateiende hinausgeht. Wenn die Absicht darin besteht, die Datei zu erweitern, indem der neue Datensatz hinzugefügt wird (mit einem PRINT# Befehl) kann die Fehlermeldung ignoriert werden. INPUT oder GET sollten nicht versucht werden, nachdem dieser Fehler ohne vorherige Neupositionierung erkannt wurde.
51, OVERFLOW IN RECORD, 00, 00	ÜBERLAUF IM EINTRAG: - Die PRINT#-Anweisung überschreitet die Datensatzgrenze. Informationen sind abgeschnitten. Da der Wagenrücklauf (RETURN) als Datensatz gesendet wird und der Eintragsabschluss in der Datensatzgröße mitgezählt wird. Diese Meldung wird angezeigt, wenn die Gesamtzahl der Zeichen im Datensatz (einschließlich des letzten RETURNs) die definierte Größe überschreitet.

Fehlermeldung	Beschreibung
52,FILE TOO LARGE,00,00	DATEI ZU GROSS: - Die Aufnahmeposition innerhalb einer relativen Datei zeigt an, dass ein Disküberlauf zur Folge hat.
60,WRITE FILE OPEN,00,00	SCHREIBDATEI OFFEN: - Diese Nachricht wird generiert, wenn eine Schreibdatei die nicht geschlossen wurde, zum Lesen geöffnet wird.
61,FILE NOT OPEN,00,00	DATEI NICHT GEÖFFNET: - Diese Meldung wird generiert, wenn auf eine Datei ein Zugriff erfolgt, die nicht im DOS geöffnet wurde. Manchmal, in diesem Fall, wird eine Nachricht nicht generiert; die Anfrage wird einfach ignoriert.
62,FILE NOT FOUND,00,00	DATEI NICHT GEFUNDEN: - Die angeforderte Datei existiert nicht auf dem angegebenen Laufwerk.
63,FILE EXISTS,00,00	DATEI EXISTIERT: - Der Dateiname der Datei, die erstellt erstellt werden soll, ist auf der Diskette bereits vorhanden.
64,FILE TYPE MISMATCH,00,00	DATEITYP NICHTÜBEREINSTIMMUNG: - Der Dateityp stimmt nicht mit dem Dateityp im Verzeichniseintrag für die angeforderte Datei überein.
65,N0 BLOCK,TT,SS	KEIN BLOCK: - Diese Meldung tritt in Verbindung mit dem B-A Befehl auf. Es zeigt an, dass der zuzuweisende Block zuvor vergeben wurde. Die Parameter geben die Spur und den Sektor an mit der nächsthöheren Nummer der verfügbar ist. Wenn die Parameter null sind (0), dann werden alle Blöcke mit höherer Nummer verwendet.

Fehlermeldung	Beschreibung
66,ILLEGAL TRACK OR SECTOR,TT,SS	ILLEGALER TRACK ODER SEKTOR: - Das DOS hat versucht, auf eine Spur oder Block zuzugreifen, der im verwendeten Format nicht vorhanden ist. Dies kann auf ein Problem beim Lesen des Zeigers auf den nächsten Block hinweisen.
67,ILLEGAL TRACK OR SECTOR,TT,SS	ILLEGALER SYSTEM T ODER S: - Diese spezielle Fehlermeldung weist auf einen illegalen Track (Spur) oder Sector (Block) hin.
70,NO CHANNEL,00,00	KEIN KANAL: - Der angeforderte Kanal ist nicht verfügbar, oder alle Kanäle sind belegt. Es dürfen maximal fünf sequentielle Dateien auf einmal für das DOS geöffnet sein. Direktzugriffskanäle können sechs geöffnete Dateien haben.
71,DIR ERROR,TT,SS	VERZEICHNIS-FEHLER: - Die BAM stimmt nicht mit dem internen Zähler überein. Es liegt ein Problem bei der BAM-Zuordnung vor oder die BAM wurde im DOS-Speicher überschrieben. Um dieses Problem zu beheben, initialisieren Sie die Diskette, um die BAM im Speicher wiederherzustellen. Einige aktive Dateien werden möglicherweise durch die Korrekturmaßnahme beendet. HINWEIS: BAM = Blockverfügbarkeit Karte
72,DISK FULL,00,00	DISKETTE VOLL: - Entweder werden die Blöcke auf der Diskette alle verwendet oder das Verzeichnis ist an der Eintragungsgrenze angekommen. DISK FULL wird gesendet, wenn nur noch zwei Blöcke auf dem 1541 Laufwerk verfügbar sind, damit die aktuelle Datei geschlossen werden kann.

Fehlermeldung	Beschreibung
73,CBM DOS V2.6 1541,00,00	DOS MISMATCH (73, CBM DOS V2.6 1541): - DOS 1 und DOS 2 werden kompatibel gelesen, sind aber nicht schreibkompatibel. Disketten können zwischen den DOS-Versionen austauschbar sein und mit DOS gelesen werden, aber eine Diskette, die auf einer Version formatiert ist, kann nicht mit der anderen Version geschrieben, weil das Format anders ist. Dieser Fehler wird immer angezeigt, wenn versucht wird, auf einen Datenträger zu schreiben, der in einem nicht kompatiblen Format formatiert wurde. (Ein Dienstprogramm ist verfügbar, um bei der Konvertierung von einem Format in das andere zu helfen.) Diese Meldung kann auch nach dem Einschalten des Laufwerks erscheinen.
74,DRIVE NOT READY,00,00	LAUFWERK NICHT BEREIT: - Es wurde versucht, auf ein 1541 Laufwerk zuzugreifen ohne Diskette in einem Laufwerk.

Tabelle CBM Floppy-Fehlermeldungen (englisch)

CBM FLOPPY ERROR MESSAGES

0	OK, no error exists.
1	Files scratched response. Not an error condition.
2-19	Unused error messages: should be ignored.
20	Block header not found on disk.
21	Sync character not found.
22	Data block not present.
23	Checksum error in data.
24	Byte decoding error.
25	Write-verify error.
26	Attempt to write with write protect on.

27	Checksum error in header.
28	Data extends into next block.
29	Disk id mismatch.
30	General syntax error
31	Invalid command.
32	Long line.
33	Invalid filename.
34	No file given.
39	Command file not found.
50	Record not present.
51	Overflow in record.
52	File too large.
60	File open for write.
61	File not open.
62	File not found.
63	File exists.
64	File type mismatch.
65	No block.
66	Illegal track or sector.
67	Illegal system track or sector.
70	No channels available.
71	Directory error.
72	Disk full or directory full.
73	Power up message, or write attempt with DOS Mismatch.
74	Drive not ready.

DESCRIPTION OF DOS ERROR MESSAGES

NOTE: Error message numbers less than 20 should be ignored with the exception of 01 which gives information about the number of files scratched with the SCRATCH command.

20: READ ERROR (block header not found) -- The disk controller is unable to locate the header of the requested data block. Caused by an illegal block number, or the header has been destroyed.

21: READ ERROR (no sync character) -- The disk controller is unable to detect a sync mark on the desired track. Caused by misalignment of the read/writer head, no diskette is present,

or unformatted or improperly seated diskette. Can also indicate a hardware failure.

22: READ ERROR (data block not present) -- The disk controller has been requested to read or verify a data block that was not properly written. This error message occurs in conjunction with the BLOCK commands and indicates an illegal track and/or block request.

23: READ ERROR (checksum error in data block) -- This error message indicates that there is an error in one or more of the data bytes. The data has been read into the DOS memory, but the checksum over the data is in error. This message may also indicate grounding problems.

24: READ ERROR (byte decoding error) -- The data or header as been read into the DOS memory, but a hardware error has been created due to an invalid bit pattern in the data byte. This message may also indicate grounding problems.

25: WRITE ERROR (write-verify error) -- This message is generated if the controller detects a mismatch between the written data and the data in the DOS memory.

26: WRITE PROTECT ON -- This message is generated when the controller has been requested to write a data block while the write protect switch is depressed. Typically, this is caused by using a diskette with a write a protect tab over the notch.

27: READ ERROR (checksum error in header) -- The controller has detected an error in the header of the requested data block. The block has not been read into the DOS memory. This message may also indicate grounding problems.

28: WRITE ERROR (long data block) -- The controller attempts to detect the sync mark of the next header after writing a data block. If the sync mark does not appear within a predetermined time, the error message is generated. The error is caused by a bad diskette format (the data extends into the next block), or by hardware failure.

29: DISK ID MISMATCH -- This message is generated when the controller has been requested to access a diskette which has not been initialized. The message can also occur if a diskette has a bad header.

30: SYNTAX ERROR (general syntax) -- The DOS cannot interpret the command sent to the command channel. Typically, this is caused by an illegal number of file names, or patterns are illegally used. For example, two file names may appear on the left side of the COPY command.

31: SYNTAX ERROR (invalid command) -- The DOS does not recognize the command. The command must start in the first position.

32: SYNTAX ERROR (invalid command) -- The command sent is longer than 58 characters.

33: SYNTAX ERROR (invalid file name) -- Pattern matching is invalidly used in the OPEN or SAVE command.

34: SYNTAX ERROR (no file given) -- the file name was left out of a command or the DOS does not recognize it as such. Typically, a colon (:) has been left out of the command,

39: SYNTAX ERROR (invalid command) -- This error may result if the command sent to command channel (secondary address 15) is unrecognized by the DOS.

50: RECORD NOT PRESENT -- Result of disk reading past the last record through INPUT#, or GET# commands. This message will also occur after positioning to a record beyond end of file in a relative file. If the intent is to expand the file by adding the new record (with a PRINT# command), the error message may be ignored. INPUT or GET should not be attempted after this error is detected without first repositioning.

51: OVERFLOW IN RECORD -- PRINT# statement exceeds record boundary. Information is cut off. Since the carriage return is sent as a record terminator is counted in the record size.

This message will occur if the total characters in the record (including the final carriage return) exceeds the defined size.

52: FILE TOO LARGE -- Record position within a relative file indicates that disk overflow will result.

60: WRITE FILE OPEN -- This message is generated when a write file that has not been closed is being opened for reading.

61: FILE NOT OPEN -- This message is generated when a file is being accessed that has not been opened in the DOS. Sometimes, in this case, a message is not generated; the request is simply ignored.

62: FILE NOT FOUND -- The requested file does not exist on the indicated drive.

63: FILE EXISTS -- The file name of the file being created already exists on the diskette.

64: FILE TYPE MISMATCH -- The file type does not match the file type in the directory entry for the requested file.

65: NO BLOCK -- This message occurs in conjunction with the B-A command. It indicates that the block to be allocated has been previously allocated. The parameters indicate the track and sector available with the next highest number. If the parameters are zero (0), then all blocks higher in number are in use.

66: ILLEGAL TRACK AND SECTOR -- The DOS has attempted to access a track or block which does not exist in the format being used. This may indicate a problem reading the pointer to the next block.

67: ILLEGAL SYSTEM T OR S -- This special error message indicates an illegal system track or block.

70: NO CHANNEL (available) -- The requested channel is not available, or all channels are in use. A maximum of five

sequential files may be opened at one time to the DOS. Direct access channels may have six opened files.

71: DIRECTORY ERROR -- The BAM does not match the internal count. There is a problem in the BAM allocation or the BAM has been overwritten in DOS memory. To correct this problem, reinitialize the diskette to restore the BAM in memory. Some active files may be terminated by the corrective action.
NOTE: BAM = Block Availability Map

72: DISK FULL -- Either the blocks on the diskette are used or the directory is at its entry limit. DISK FULL is sent when two blocks are available on the 1541 to allow the current file to be closed.

73: DOS MISMATCH (73, CBM DOS V2.6 1541) -- DOS 1 and 2 are read compatible but not write compatible. Disks may be interchangeably read with either DOS, but a disk formatted on one version cannot be written upon with the other version because the format is different. This error is displayed whenever an attempt is made to write upon a disk which has been formatted in a non-compatible format. (A utility routine is available to assist in converting from one format to another.) This message may also appear after power up.

74: DRIVE NOT READY -- An attempt has been made to access the 1541 Single Drive Floppy Disk without any diskettes present in either drive.

C64 BASIC V2.0: Befehle, Schlüsselwörter und Token

geschrieben von Andreas Potthoff | 3. Mai 2023

Beim Commodore 64 (auch VC-20) werden bei der internen Verarbeitung der **BASIC-Befehle (Schlüsselwörter)** eines BASIC-

Programms für eine schnellere Verarbeitung und Speicherplatzersparnis sogenannte **Token (Schlüsselzeichen)** statt der Schlüsselwörter verwendet.

Die folgende Tabelle zeigt die BASIC-Schlüsselwörter, den dazugehörigen Token (dezimal / hexadezimal), die ROM-Einsprungsadresse (hexadezimal), die Abkürzung und den Typ des Befehls.

Nach der Tabelle erhalten Sie weiterführende Informationen zur Tokenisation der Befehle.

Inhaltsverzeichnis

- Westfälische Wilhelms-Universität heißt ab dem 1. Oktober Universität Münster
- Links:

Tabelle C64 Token

BASIC Schlüsselwort	Token dez	Token hex	ROM Exec dez	ROM Exec hex	Abk.	Typ
END	128	\$80	43057	\$A831	eN	Anweisung/Befehl
FOR	129	\$81	42818	\$A742	f0	Anweisung/Befehl
NEXT	130	\$82	44318	\$AD1E	nE	Anweisung/Befehl
DATA	131	\$83	43256	\$A8F8	dA	Anweisung/Befehl
INPUT#	132	\$84	43941	\$ABA5	iN	Anweisung/Befehl
INPUT	133	\$85	43967	\$ABBF	-	Anweisung/Befehl
DIM	134	\$86	45185	\$B081	dI	Anweisung/Befehl
READ	135	\$87	44038	\$AC06	rE	Anweisung/Befehl
LET	136	\$88	43429	\$A9A5	lE	Anweisung/Befehl
GOTO	137	\$89	43168	\$A8A0	g0	Anweisung/Befehl

BASIC Schlüsselwort	Token dez	Token hex	ROM Exec dez	ROM Exec hex	Abk.	Typ
RUN	138	\$8A	43121	\$A871	rU	Anweisung/Befehl
IF	139	\$8B	43304	\$A928	-	Anweisung/Befehl
RESTORE	140	\$8C	43037	\$A81D	reS	Anweisung/Befehl
GOSUB	141	\$8D	43139	\$A883	goS	Anweisung/Befehl
RETURN	142	\$8E	43218	\$A8D2	reT	Anweisung/Befehl
REM	143	\$8F	43323	\$A93B	-	Anweisung/Befehl
STOP	144	\$90	43055	\$A82F	sT	Anweisung/Befehl
ON	145	\$91	43339	\$A94B	-	Anweisung/Befehl
WAIT	146	\$92	47149	\$B82D	wA	Anweisung/Befehl
LOAD	147	\$93	57704	\$E168	lO	Anweisung/Befehl
SAVE	148	\$94	57686	\$E156	sA	Anweisung/Befehl
VERIFY	149	\$95	57701	\$E165	vE	Anweisung/Befehl
DEF	150	\$96	46003	\$B3B3	dE	Anweisung/Befehl
POKE	151	\$97	47140	\$B824	pO	Anweisung/Befehl
PRINT#	152	\$98	43648	\$AA80	pR	Anweisung/Befehl
PRINT	153	\$99	43680	\$AAA0	?	Anweisung/Befehl
CONT	154	\$9A	43095	\$A857	cO	Anweisung/Befehl
LIST	155	\$9B	42652	\$A69C	lI	Anweisung/Befehl
CLR	156	\$9C	42590	\$A65E	cL	Anweisung/Befehl
CMD	157	\$9D	43654	\$AA86	cM	Anweisung/Befehl
SYS	158	\$9E	57642	\$E12A	sY	Anweisung/Befehl
OPEN	159	\$9F	57790	\$E1BE	oP	Anweisung/Befehl
CLOSE	160	\$A0	57799	\$E1C7	clO	Anweisung/Befehl
GET	161	\$A1	43899	\$AB7B	gE	Anweisung/Befehl
NEW	162	\$A2	42562	\$A642	-	Anweisung/Befehl
TAB(163	\$A3	43752	\$AAE8	tA	Anweisung/Befehl, Spezial

BASIC Schlüsselwort	Token dez	Token hex	ROM Exec dez	ROM Exec hex	Abk.	Typ
T0	164	\$A4	42861	\$A76D	-	Anweisung/Befehl, Spezial
FN	165	\$A5	46068	\$B3F4	-	Anweisung/Befehl, Spezial
SPC (166	\$A6	43769	\$AAF9	sP	Anweisung/Befehl, Spezial
THEN	167	\$A7	43314	\$A932	tH	Anweisung/Befehl, Spezial
NOT	168	\$A8	44756	\$AED4	n0	Anweisung/Befehl, Spezial
STEP	169	\$A9	42905	\$A799	stE	Anweisung/Befehl, Spezial
+	170	\$AA	47210	\$B86A	-	Operator, numerisch/string
-	171	\$AB	47187	\$B853	-	Operator, numerisch
*	172	\$AC	47659	\$BA2B	-	Operator, numerisch
/	173	\$AD	47890	\$BB12	-	Operator, numerisch
^	174	\$AE	49019	\$BF7B	-	Operator, numerisch
AND	175	\$AF	45033	\$AFE9	aN	Operator, logisch
OR	176	\$B0	45030	\$AFE6	-	Operator, logisch
>	177	\$B1	49076	\$BFB4	-	Operator, logisch
=	178	\$B2	44756	\$AED4	-	Operator, logisch
<	179	\$B3	45078	\$B016	-	Operator, logisch
SGN	180	\$B4	48185	\$BC39	sG	Funktion, numerisch

BASIC Schlüsselwort	Token dez	Token hex	ROM Exec dez	ROM Exec hex	Abk.	Typ
INT	181	\$B5	48332	\$BCCC	-	Funktion, numerisch
ABS	182	\$B6	48216	\$BC58	abS	Funktion, numerisch
USR	183	\$B7	784	\$0310	uS	Funktion, numerisch/string
FRE	184	\$B8	45949	\$B37D	fR	Funktion, numerisch, Spezial
POS	185	\$B9	45982	\$B39E	-	Funktion, numerisch, Spezial
SQR	186	\$BA	49009	\$BF71	sQ	Funktion, numerisch
RND	187	\$BB	57495	\$E097	rN	Funktion, numerisch
LOG	188	\$BC	45794	\$B9EA	-	Funktion, numerisch
EXP	189	\$BD	49133	\$BFED	eX	Funktion, numerisch
COS	190	\$BE	57956	\$E264	-	Funktion, numerisch
SIN	191	\$BF	57963	\$E26B	sI	Funktion, numerisch
TAN	192	\$C0	58036	\$E2B4	-	Funktion, numerisch
ATN	193	\$C1	58128	\$E30E	aT	Funktion, numerisch
PEEK	194	\$C2	47117	\$B80E	pE	Funktion, numerisch

BASIC Schlüsselwort	Token dez	Token hex	ROM Exec dez	ROM Exec hex	Abk.	Typ
LEN	195	\$C3	46972	\$B77C	-	Funktion, numerisch
STR\$	196	\$C4	46181	\$B465	stR	Funktion, string
VAL	197	\$C5	47021	\$B7AD	vA	Funktion, numerisch
ASC	198	\$C6	46987	\$B78B	aS	Funktion, numerisch
CHR\$	199	\$C7	46828	\$B6EC	cH	Funktion, string
LEFT\$	200	\$C8	46848	\$B700	leF	Funktion, string
RIGHT\$	201	\$C9	46892	\$B72C	rI	Funktion, string
MID\$	202	\$CA	46903	\$B737	mI	Funktion, string
GO	203	\$CB	43026	\$A812	-	Anweisung/Befehl, Spezial
π	255	\$FF	44702	\$AE9E	-	Funktion, numerisch, Konstante
ST (STATUS)	-	-	65463	\$FFB7	-	Systemvariable
TI (TIME)	-	-	?	?	-	Systemvariable
TI\$ (TIME\$)	-	-	43488	\$A9E0	-	Systemvariable

Tokenisation

Hier werden die Schlüsselwörter (Befehle) in ein Single-Byte-Wert (Token) umgewandelt, wenn sie in einem Programm gespeichert sind. Das geschieht entweder durch einen Programmstart mit RUN oder im Direktmodus (Konsole) durch das Drücken der Taste RETURN, wenn dort Kommandos eingegeben

worden sind. Der BASIC-Interpreter arbeitet die Token der Reihenfolge nach ab.

De-Tokenisation

Hier werden die Token (Bytes) in lesbare BASIC-Befehle (Schlüsselwörter) umgewandelt, was eigentlich *nur für das LIST-Kommando* als Ausgabe und für gute menschliche Lesbarkeit zutrifft. Ansonsten wird die De-Tokenisation nicht angewandt.

Single-Byte-Token

Ein Token kann beim Commodore BASIC V2 (ab 4.0) einen Single-Byte-Wert zwischen 128-255 (\$80-\$FF) haben und belegt nur 1 Byte Arbeitsspeicher. Token-Codes sind immer größer oder gleich 128 (\$80); d.h. das höchstwertige Bit in einem Byte, das einen Token repräsentiert, ist immer gesetzt und somit werden keine PETSCII-Zeichen (<128/<\$80) als Token-Code benutzt.

Two-Bytes-Token

Ab dem Commodore BASIC 7.0 (C 128) werden wegen des umfangreichen BASIC-Befehlssatzes zwei Bytes für ein Token benötigt.

Ausführen von Token

Jedes Token hat eine sogenannte Ausführungsadresse (EXEC) im ROM, wo dann der entsprechende Code für den jeweiligen Token bzw. BASIC-Befehl ausgeführt wird.

Schlüsselwörter

BASIC V2 enthält 76 Schlüsselwörter, 8 Operatoren, 1 Konstante und 3 Systemvariablen, die in verschiedenen Gruppen gegliedert sind:

- 128-162 (\$80-\$A2): *Befehle*
- 163-169 (\$A3-\$A9): *“Bywords“*, die Teil der Syntax der vorherigen Befehle sind
- 170-179 (\$AA-\$B3): *Arithmetische und logische Operatoren*
- 180-202 (\$B4-\$CA): *Funktionen*
- 203 (\$CB): *Befehl* – GO (der hier als Ausnahme hinter den Funktionen liegt)
- 204-254 (\$CC-\$FE): Ein freier Bereich für 51 zusätzliche Token, z.B. der für BASIC-Erweiterungen von Drittanbietern genutzt wird
- 255 (\$FF): *Konstante* – Pi
- *Systemvariablen*: ST (STATUS), TI (TIME), TI\$ (TIME\$). Im BASIC V2 ROM werden die Systemvariablen als Ausnahmen in den Routinen zur Behandlung *normaler* Variablen behandelt.

Abkürzungen

Die meisten BASIC-Schlüsselwörter kann man bei der Eingabe abkürzen. Abgekürzte Schlüsselwörter werden i.d.R. gebildet, indem man die ersten (manchmal bis zu drei) Zeichen eintippt und das nächste Zeichen mit SHIFT eingibt.

Auch hier gibt es wieder einige Ausnahmen. Einige Schlüsselwörter (CLOSE, GOSUB, LEFT\$, RESTORE, RETURN, STEP, STR\$) benötigen gekürzt 3 statt 2 Zeichen. Der BASIC-Befehl PRINT wird nur mit einem Zeichen, dem ? abgekürzt. Es gibt auch einige BASIC-Befehle (INPUT, COS, FN, TO, IF, INT, LEN, LOG, NEW, ON, OR, POS, REM, TAN) die nicht abkürzbar sind.

Speichern

Es kann schon mal vorkommen, dass *eine Programmzeile* 80 Zeichen überschreitet, also dafür mehr als 2 Zeilen auf dem Bildschirm angezeigt werden. Dies geschieht dadurch, dass die Ausgabe der Token beim LIST-Befehl eben ungekürzt passiert und somit Programmzeilen mit mehr als 80 Zeichen auftreten können.

Wenn sie eine solche Programmzeile ändern wollen, müssen sie die Abkürzungen erneut eingeben bevor sie das Programm speichern. Achten sie dann darauf, dass sie nicht mehr als 80 Zeichen für eine Programmzeile insgesamt verwenden. Alle zusätzlichen Zeichen danach werden nach dem Drücken von RETURN automatisch abgeschnitten. Beim Speichern eines Programms auf einen Datenträger werden die Token und nicht die Schlüsselwörter benutzt.

Befehle	CLOSE, CLR, CMD, CONT, DATA, DEF, DIM, END, FOR, GET, GET#, GOSUB, GOTO, IF, INPUT, INPUT#, LET, LIST, LOAD, NEW, NEXT, ON, OPEN, POKE, PRINT, PRINT#, READ, REM, RESTORE, RETURN, RUN, SAVE, STOP, SYS, VERIFY, WAIT
Befehle Spezial (Bywords)	FN, GO, NOT, SPC(, TAB(, THEN, TO, STEP
Arithmetische und logische Operatoren	+, -, *, /, ^, >, =, <, AND, OR
Funktionen	ABS, ASC, ATN, CHR\$, COS, EXP, FRE, INT, LEFT\$, LEN, LOG, MID\$, PEEK, POS, RIGHT\$, RND, SGN, SIN, SQR, STR\$, TAN, USR, VAL
Konstanten und Systemvariablen	Pi, ST, TI, TI\$

C64 BASIC V2.0: Ableitung

mathematischer Funktionen

geschrieben von Andreas Potthoff | 3. Mai 2023

Mathematische Funktionen, die nicht Commodore 64 BASIC eigen sind, können wie folgt berechnet werden:

Mathematische Funktion	BASIC Äquivalent
SECANT	$\text{SEC}(X) = 1/\text{COS}(X)$
COSECANT	$\text{CSC}(X) = 1/\text{SIN}(X)$
COTANGENT	$\text{COT}(X) = 1/\text{TAN}(X)$
INVERSE SINE	$\text{ARCSIN}(X) = \text{ATN}(X/\text{SQR}(-X*X+1))$
INVERSE COSINE	$\text{ARCCOS}(X) = -\text{ATN}(X/\text{SQR}(-X*X+1))+\{\pi\}/2$
INVERSE SECANT	$\text{ARCSEC}(X) = \text{ATN}(X/\text{SQR}(X*X-1))$
INVERSE COSECANT	$\text{ARCCSC}(X) = \text{ATN}(X/\text{SQR}(X*X-1))+(\text{SGN}(X)-1*\{\pi\})/2$
INVERSE COTANGENT	$\text{ARCOT}(X) = \text{ATN}(X)+\{\pi\}/2$
HYPERBOLIC SINE	$\text{SINH}(X) = (\text{EXP}(X)-\text{EXP}(-X))/2$
HYPERBOLIC COSINE	$\text{COSH}(X) = (\text{EXP}(X)+\text{EXP}(-X))/2$
HYPERBOLIC TANGENT	$\text{TANH}(X) = \text{EXP}(-X)/(\text{EXP}(X)+\text{EXP}(-X))*2+1$
HYPERBOLIC SECANT	$\text{SECH}(X) = 2/(\text{EXP}(X)+\text{EXP}(-X))$
HYPERBOLIC COSECANT	$\text{CSCH}(X) = 2/(\text{EXP}(X)-\text{EXP}(-X))$
HYPERBOLIC COTANGENT	$\text{COTH}(X) = \text{EXP}(-X)/(\text{EXP}(X)-\text{EXP}(-X))*2+1$
INVERSE HYPERBOLIC SINE	$\text{ARCSINH}(X) = \text{LOG}(X+\text{SQR}(X*X+1))$
INVERSE HYPERBOLIC COSINE	$\text{ARCCOSH}(X) = \text{LOG}(X+\text{SQR}(X*X-1))$

INVERSE HYPERBOLIC TANGENT	$\text{ARCTANH}(X) = \text{LOG}((1+X)/(1-X))/2$
INVERSE HYPERBOLIC SECANT	$\text{ARCSECH}(X) = \text{LOG}((\text{SQR}(-X*X+1)+1/X)$
INVERSE HYPERBOLIC COSECANT	$\text{ARCCSCH}(X) = \text{LOG}((\text{SGN}(X)*\text{SQR}(X*X+1/X)$
INVERSE HYPERBOLIC COTANGENT	$\text{ARCCOTH}(X) = \text{LOG}((X+1)/(X-1))/2$

Die Verzeichnisstruktur von Linux (Debian based)

geschrieben von Andreas Potthoff | 3. Mai 2023

Dieser Beitrag beschreibt die typische Verzeichnisstruktur für Linux-Systeme, welche auf Debian basieren. Dazu gehören z.B.: Ubuntu, Raspberry Pi OS, Raspbian, Linux Mint, Kali Linux, Kubuntu, Proxmox u.v.m. Eine umfangreiche Liste von Debian based-Linux-Distributionen findet ihr hier: [Distrowatch.com](https://distrowatch.com) – Debian based

Alle elementaren Programme und Konfigurationsdateien liegen in verschiedenen Systemverzeichnissen. Anhand des Filesystem Hierarchie Standard (FHS) wird die Eingliederung der verschiedenen Dateiarten in die Verzeichnisstruktur gezeigt. Der FHS ist eine Richtlinie für die Verzeichnisstruktur unter Unix-ähnlichen Betriebssystemen.

Sie können den folgenden Informationen entnehmen, wo Sie Konfigurationsdateien finden, welche Verzeichnisse Programme enthalten und wo Dokumentationen zu finden sind.

Die folgende tabellarische Übersicht der Verzeichnisstruktur berücksichtigt nicht eine Installation mit gesonderten

Partionen (z.B. boot, home, etc.). Außerdem wird Beschreibung der Dateisystem-Hierarchie durch den Konsolenbefehl **man** (manual) gezeigt.

Inhaltsverzeichnis

- Westfälische Wilhelms-Universität heißt ab dem 1. Oktober Universität Münster
 - Links:

Tabelle Verzeichnisstruktur

Die *wichtigsten* Verzeichnisse haben folgende Bedeutungen bzw. Inhalte:

Verzeichnis	Beschreibung
/	Wurzelverzeichnis: Das Wurzelverzeichnis und oberste Verzeichnis der Hierarchie.
/bin	Systemprogramme: Die wichtigsten Systemkommandos. Dieses Verzeichnis darf keine Unterverzeichnisse enthalten.
/boot	Bootloader: Die statischen Dateien des Bootloaders und die Kernel.
/dev	Geräte-dateien: Schnittstellen zur Ansteuerung der gesamten Hardware.
/etc	Systemkonfiguration: Die Konfigurationsdateien.
/etc/init.d	Start- und Stopscripte.
/etc/network	Konfigurationsdateien des Netzwerkes. z.B. interfaces
/etc/opt	Konfigurationsdateien für Programme im Verzeichnis /opt

Verzeichnis	Beschreibung
/home/Benutzername	<p>Heimatverzeichnis: Die eigenen Dateien. Für jeden Benutzer finden Sie ein eigenes Verzeichnis mit dem Benutzernamen.</p>
/lib	<p>Bibliotheken: Dynamische Bibliotheken und Kernelmodule die für den Systemstart notwendig sind.</p>
/lib/modules	<p>Kernelmodule</p>
/lost+found	<p>Verloren+Gefunden: Dateien und Dateifragmente die bei der Reparatur eines defekten Dateissystems übrig geblieben sind.</p>
/media	<p>Wechselmedien: Einhängpunkt und Unterverzeichnisse für transportable Medienspeicher (z.B. USB-Stick, externe Festplatte, Floppy, CD-ROM, DVD, u.a).</p>
/mnt	<p>Mountpunkt: Einhängpunkt für ein temporär eingehängtes Dateisystem.</p>
/opt	<p>Optional: Für die manuelle Installation von Programmen die ihre eigenen Bibliotheken mitbringen und nicht zum Standard der Distribution gehören. Unterverzeichnisse sind z.B. minecraft-pi, sonic-pi, vs, Wolfram</p>
/proc	<p>Prozess- und Systeminformationen: Schnittstellen zum aktuell geladenen Kernel und seinen Prozeduren. Dateien lassen sich mittel cat auslesen z.B. "cat proc/version" gibt die aktuelle Kernelversion aus.</p>

Verzeichnis	Beschreibung
/root	<p>Root Heimatverzeichnis: Das Heimatverzeichnis des Superusers (root). Das root-Verzeichnis liegt im Wurzelverzeichnis, falls das home-Verzeichnis auf eine andere Partiton ausgelagert wurde oder ein Zugriff auf home aus irgendeinem Grund nicht möglich ist.</p>
/run	<p>Prozesse: Dateien für laufende Prozesse. Hier befinden sich auch die meisten PID-Dateien (Process Identifier).</p>
/sbin	<p>Systemprogramme (root): Die Systemkommandos zur Systemverwaltung für die Rootrechte benötigt werden. Dieses Verzeichnis darf keine Unterverzeichnisse enthalten.</p>
/srv	<p>Systemdienste: Daten für Systemdienste (in der Regel leer). Dieses Verzeichnis ist noch nicht genau spezifiziert.</p>
/sys	<p>System: Systemweites Geräteverzeichnis welches Informationen und Statistiken über das System und die Komponenten enthält.</p>
/tmp	<p>Temporär: Temporäre Dateien von Programmen und Benutzern. Dieses Verzeichnis wird nach jedem Neustart automatisch geleert.</p>
/usr	<p>UNIX Systemressourcen: Die meisten Systemprogramme, Bibliotheken und installierten Programme.</p>
/usr/bin	<p>Die meisten Benutzerbefehle. Das primäre Verzeichnis für ausführbare Dateien des Systems.</p>
/usr/games	<p>Spiele</p>
/usr/include	<p>Header-Dateien für C-Programme.</p>

Verzeichnis	Beschreibung
/usr/lib	Allgemeine Bibliotheken
/usr/local	Distributionsunabhängige lokale Hierarchie enthält noch einmal die gleiche Verzeichnisstruktur wie das /usr Verzeichnis. Für Programme und Daten gedacht, die von der entsprechenden Distribution des jeweiligen Systems unabhängig installiert worden sind, wie etwa selbstkompilierte oder unabhängig von der Distribution heruntergeladene Programme und Dateien.
/usr/sbin	Weniger wichtige Systemprogramme die im Gegensatz zu /sbin nicht während des Bootvorganges verwendet werden.
/usr/share	Statische architekturunabhängige Dateien z.B. Dokumentationen, Manpages und Wörterbücher.
/usr/src	Source Code zu den Paketen.
/var	Variable Daten: Diese Daten entstehen z.B. im Zuge einer Abarbeitung
/var/backups	Sicherungskopien der Liste der installierten Programme (dpkg).
/var/cache	Zwischenspeicher von Programmen
/var/lib	Variable Statusinformationen
/var/local	Variable Daten im Zusammenhang mit /usr/local
/var/lock	Lock-Dateien zur Prozesssynchronisation
/var/log	Protokolldateien
/var/mail	Mailboxen der Benutzer
/var/opt	Variable Daten der optionalen Programme
/var/run	Dateien zu laufenden Prozessen

Verzeichnis	Beschreibung
/var/spool	Von Programmen gespoolte Daten z.B. Druckaufträge.
/var/tmp	Variable Daten die zwischen Reboots erhalten bleiben.
/var/www	Standardverzeichnis für Inhalte des Webservers .

Verzeichnisstruktur mit dem Konsolenbefehl tree anzeigen

Um sich einen Überblick der Verzeichnisstruktur zu verschaffen kann man den Konsolenbefehl **tree** benutzen:

Als erstes wechseln wir in das Wurzelverzeichnis und dann lassen wir uns den Verzeichnisbaum mit einer Leveltiefe von 1 anzeigen. Die Leveltiefe kann natürlich auch erweitert werden, indem man die 1 durch die gewünschte Tiefe ersetzt.

```
pi@pi4b:~ $ cd /  
pi@pi4b:/ $ tree -L 1
```

```
.  
├── bin  
├── boot  
├── boot.bak  
├── dev  
├── etc  
├── home  
├── lib  
├── lost+found  
├── media  
├── mnt  
├── opt  
├── proc  
├── root  
├── run  
├── sbin  
└── srv
```

```
|— sys
|— tmp
|— usr
└— var
```

20 directories, 0 files

Beschreibung der Dateisystem-Hierarchie mit dem Konsolenbefehl man (englisch)

Wenn Sie folgenden Konsolenbefehl eingeben:

```
man hier
```

erhalten Sie die Beschreibung der Dateisystem-Hierarchie mit folgender Beispielausgabe:

```
HIER(7) Linux Programmer's Manual HIER(7)
```

NAME

```
hier - description of the filesystem hierarchy
```

DESCRIPTION

```
A typical Linux system has, among others, the following directories:
```

```
/ This is the root directory. This is where the whole tree starts.
```

```
/bin This directory contains executable programs which are needed in single user mode and to bring the system up or repair it.
```

```
/boot Contains static files for the boot loader. This directory holds only the files which are needed during the boot process. The map installer and configuration files should go to /sbin and /etc.
```

```
/dev Special or device files, which refer to physical devices.
```

See `mknod(1)`.

`/etc` Contains configuration files which are local to the machine. Some larger software packages, like X11, can have their own subdirectories below `/etc`. Site-wide configuration files may be placed here or in `/usr/etc`. Nevertheless, programs should always look for these files in `/etc` and you may have links for these files to `/usr/etc`.

`/etc/opt`

Host-specific configuration files for add-on applications installed in `/opt`.

`/etc/sgml`

This directory contains the configuration files for SGML and XML (optional).

`/etc/skel`

When a new user account is created, files from this directory are usually copied into the user's home directory.

`/etc/X11`

Configuration files for the X11 window system (optional).

`/home` On machines with home directories for users, these are usually beneath this directory, directly or not. The structure of this directory depends on local administration decisions.

`/lib` This directory should hold those shared libraries that are necessary to boot the system and to run the commands in the root filesystem.

`/media` This directory contains mount points for removable media such as CD and DVD disks or USB sticks.

`/mnt` This directory is a mount point for a temporarily mounted filesystem. In some distributions, `/mnt` contains

subdirectories intended

to be used as mount points for several temporary filesystems.

`/opt` This directory should contain add-on packages that contain static files.

`/proc` This is a mount point for the `proc` filesystem, which provides information about running processes and the kernel. This pseudo-filesystem is described in more detail in `proc(5)`.

`/root` This directory is usually the home directory for the root user (optional).

`/sbin` Like `/bin`, this directory holds commands needed to boot the system, but which are usually not executed by normal users.

`/srv` This directory contains site-specific data that is served by this system.

`/tmp` This directory contains temporary files which may be deleted with no notice, such as by a regular job or at system boot up.

`/usr` This directory is usually mounted from a separate partition. It should hold only sharable, read-only data, so that it can be mounted by various machines running Linux.

`/usr/X11R6`

The X-Window system, version 11 release 6 (optional).

`/usr/X11R6/bin`

Binaries which belong to the X-Window system; often, there is a symbolic link from the more traditional `/usr/bin/X11` to here.

`/usr/X11R6/lib`

Data files associated with the X-Window system.

`/usr/X11R6/lib/X11`

These contain miscellaneous files needed to run X; Often, there is a symbolic link from /usr/lib/X11 to this directory.

/usr/X11R6/include/X11

Contains include files needed for compiling programs using the X11 window system. Often, there is a symbolic link from /usr/include/X11 to this directory.

/usr/bin

This is the primary directory for executable programs. Most programs executed by normal users which are not needed for booting or for repairing the system and which are not installed locally should be placed in this directory.

/usr/bin/X11

is the traditional place to look for X11 executables; on Linux, it usually is a symbolic link to /usr/X11R6/bin.

/usr/dict

Replaced by /usr/share/dict.

/usr/doc

Replaced by /usr/share/doc.

/usr/etc

Site-wide configuration files to be shared between several machines may be stored in this directory. However, commands should always reference those files using the /etc directory. Links from files in /etc should point to the appropriate files in /usr/etc.

/usr/games

Binaries for games and educational programs (optional).

/usr/include

Include files for the C compiler.

/usr/include/X11

Include files for the C compiler and the X-Window system. This is usually a symbolic link to /usr/X11R6/include/X11.

`/usr/include/asm`

Include files which declare some assembler functions. This used to be a symbolic link to `/usr/src/linux/include/asm`.

`/usr/include/linux`

This contains information which may change from system release to system release and used to be a symbolic link to `/usr/src/linux/include/linux` to get at operating-system-specific information.

(Note that one should have include files there that work correctly with the current libc and in user space. However, Linux kernel

source is not designed to be used with user programs and does not know anything about the libc you are using. It is very likely

that things will break if you let `/usr/include/asm` and `/usr/include/linux` point at a random kernel tree. Debian systems don't do

this and use headers from a known good kernel version, provided in the `libc*-dev` package.)

`/usr/include/g++`

Include files to use with the GNU C++ compiler.

`/usr/lib`

Object libraries, including dynamic libraries, plus some executables which usually are not invoked directly. More complicated programs may have whole subdirectories there.

`/usr/lib/X11`

The usual place for data files associated with X programs, and configuration files for the X system itself. On Linux, it usually

is a symbolic link to `/usr/X11R6/lib/X11`.

`/usr/lib/gcc-lib`

contains executables and include files for the GNU C compiler, `gcc(1)`.

/usr/lib/groff

Files for the GNU groff document formatting system.

/usr/lib/uucp

Files for uucp(1).

/usr/local

This is where programs which are local to the site typically go.

/usr/local/bin

Binaries for programs local to the site.

/usr/local/doc

Local documentation.

/usr/local/etc

Configuration files associated with locally installed programs.

/usr/local/games

Binaries for locally installed games.

/usr/local/lib

Files associated with locally installed programs.

/usr/local/include

Header files for the local C compiler.

/usr/local/info

Info pages associated with locally installed programs.

/usr/local/man

Man pages associated with locally installed programs.

/usr/local/sbin

Locally installed programs for system administration.

/usr/local/share

Local application data that can be shared among different architectures of the same OS.

/usr/local/src

Source code for locally installed software.

`/usr/man`

Replaced by `/usr/share/man`.

`/usr/sbin`

This directory contains program binaries for system administration which are not essential for the boot process, for mounting `/usr`, or for system repair.

`/usr/share`

This directory contains subdirectories with specific application data, that can be shared among different architectures of the same OS. Often one finds stuff here that used to live in `/usr/doc` or `/usr/lib` or `/usr/man`.

`/usr/share/dict`

Contains the word lists used by spell checkers.

`/usr/share/doc`

Documentation about installed programs.

`/usr/share/games`

Static data files for games in `/usr/games`.

`/usr/share/info`

Info pages go here.

`/usr/share/locale`

Locale information goes here.

`/usr/share/man`

Manual pages go here in subdirectories according to the man page sections.

`/usr/share/man/<locale>/man[1-9]`

These directories contain manual pages for the specific locale in source code form. Systems which use a unique language and code set for all manual pages may omit the `<locale>` substring.

`/usr/share/misc`

Miscellaneous data that can be shared among different architectures of the same OS.

`/usr/share/nls`

The message catalogs for native language support go here.

`/usr/share/sgml`

Files for SGML and XML.

`/usr/share/terminfo`

The database for terminfo.

`/usr/share/tmac`

Troff macros that are not distributed with groff.

`/usr/share/zoneinfo`

Files for timezone information.

`/usr/src`

Source files for different parts of the system, included with some packages for reference purposes. Don't work here with your own projects, as files below `/usr` should be read-only except when installing software.

`/usr/src/linux`

This was the traditional place for the kernel source. Some distributions put here the source for the default kernel they ship.

You should probably use another directory when building your own kernel.

`/usr/tmp`

Obsolete. This should be a link to `/var/tmp`. This link is present only for compatibility reasons and shouldn't be used.

`/var` This directory contains files which may change in size, such as spool and log files.

`/var/adm`

This directory is superseded by `/var/log` and should be a

symbolic link to /var/log.

/var/backups

Reserved for historical reasons.

/var/cache

Data cached for programs.

/var/catman/cat[1-9] or /var/cache/man/cat[1-9]

These directories contain preformatted manual pages according to their man page section. (The use of preformatted manual pages is deprecated.)

/var/cron

Reserved for historical reasons.

/var/lib

Variable state information for programs.

/var/local

Variable data for /usr/local.

/var/lock

Lock files are placed in this directory. The naming convention for device lock files is LCK.<device> where <device> is the device's name in the filesystem. The format used is that of HDU UUCP lock files, that is, lock files contain a PID as a 10-byte

ASCII decimal number, followed by a newline character.

/var/log

Miscellaneous log files.

/var/opt

Variable data for /opt.

/var/mail

Users' mailboxes. Replaces /var/spool/mail.

/var/msg

Reserved for historical reasons.

/var/preserve

Reserved for historical reasons.

/var/run

Run-time variable files, like files holding process identifiers (PIDs) and logged user information (utmp). Files in this directory are usually cleared when the system boots.

/var/spool

Spooled (or queued) files for various programs.

/var/spool/at

Spooled jobs for at(1).

/var/spool/cron

Spooled jobs for cron(8).

/var/spool/lpd

Spooled files for printing.

/var/spool/mail

Replaced by /var/mail.

/var/spool/mqueue

Queued outgoing mail.

/var/spool/news

Spool directory for news.

/var/spool/rwho

Spooled files for rwhod(8).

/var/spool/smail

Spooled files for the smail(1) mail delivery program.

/var/spool/uucp

Spooled files for uucp(1).

/var/tmp

Like /tmp, this directory holds temporary files stored for an unspecified duration.

/var/yp

Database files for NIS.

Selten und leistungsstark: Kryoelektronenmikroskop eingeweiht

geschrieben von Andreas Potthoff | 3. Mai 2023

Technik ermöglicht Einblicke in Zellen bis hin zur atomaren Auflösung / Symposium am Center for Soft Nanoscience

Bei einem Eröffnungssymposium mit rund 150 Gästen ist im Center for Soft Nanoscience (SoN) der Westfälischen Wilhelms-Universität Münster (WWU) am 19. April (Mittwoch) ein technisch herausragendes Hochleistungs-Kryoelektronenmikroskop („Kryo-EM“) offiziell eingeweiht worden. Das Gerät ist weltweit eines der leistungsstärksten seiner Art und wird unter der Leitung von Prof. Dr. Christos Gatsogiannis künftig von rund 20 Arbeitsgruppen sowie Forschungsverbänden aus den Disziplinen Medizin, Biologie und Chemie genutzt. Es soll dazu beitragen, dass die Universität Münster auch international ihre führende Rolle in der multiskaligen Bildgebung beibehält und ausbaut.

Das Kryo-EM kann kleinste Bestandteile von Zellen abbilden – bis hin zu einzelnen Atomen – und somit helfen, die Prozesse im Inneren der Zellen zu verstehen. Mit seiner Hilfe wollen Wissenschaftlerinnen und Wissenschaftler zudem die Strukturen von Proteinen sichtbar machen und dadurch Rückschlüsse auf ihre Funktionsweise ziehen oder auch herausfinden, wie die Proteine in lebenden Zellen Krankheiten hervorrufen.

Die Deutsche Forschungsgemeinschaft (DFG) und das Land Nordrhein-Westfalen hatten im Rahmen des Förderprogramms „Forschungsgroßgeräte“ insgesamt 7,5 Millionen Euro für die Ausstattung zur Verfügung gestellt. Neben dem Hochleistungs-Kryo-EM gehören zwei weitere Geräte dazu: ein automatisiertes Screening-Elektronenmikroskop, das eine optimale Vorauswahl der Proben ermöglicht, und ein Kryo-fokussiertes-Ionenstrahl-/Rasterelektronenmikroskop, das zur Präparation der Proben benötigt wird.

Voraussetzung für die Anschaffung des Kryo-EM war die besondere bauliche Ausstattung des SoN. Unter anderem ist der Gebäudeteil im Erdgeschoss, in dem das Gerät steht, von einem Erdhügel bedeckt. Dadurch ist das Kryo-EM-Labor von äußeren Schwingungen isoliert. Zudem ist der Boden des Labors fast perfekt schwingungsgedämpft, schließlich ist der Raum gegen störende Magnetfelder abgeschirmt.

Der Biologe Christos Gatsogiannis leitet die Arbeitsgruppe „Kryo-EM komplexer Nanosysteme“ der Medizinischen Fakultät an der Universität Münster. Die Gruppe ist im SoN untergebracht.

Hintergrundinformationen zur Kryoelektronenmikroskopie

Die Kryo-EM ist eine Variante der Transmissionselektronenmikroskopie. Dabei werden mikroskopische Objekte mithilfe von Elektronenstrahlen bei kryogenen, also extrem niedrigen Temperaturen abgebildet. Durch ein spezielles Verfahren werden die Proben so schnell auf minus 196 Grad Celsius abgekühlt, dass sie nahezu

unversehrt erhalten bleiben. Die Wissenschaftler Jacques Dubochet, Joachim Frank und Richard Henderson erhielten im Jahr 2017 den Chemie-Nobelpreis für die Entwicklung dieser Methode. 2020 gelang es einem Forscherteam erstmals, einzelne Atome in einer Proteinstruktur mittels Kryo-EM zu beobachten. Durch diese Technik ist es möglich, nachzuvollziehen, wie Proteine in lebenden Zellen funktionieren oder Krankheiten hervorrufen.

Links:

- Artikel aus der Unizeitung wissen|leben Nr. 2, 29. März 2023
<https://www.uni-muenster.de/news/view.php?cmdid=13215>
- Video „Inauguration of a high-performance cryogenic electron microscope at the Center for Soft Nanoscience“
<https://youtu.be/spGc8WKDo1Y>
- AG „KryoEM komplexer Nanosysteme“ an der WWU Münster
<https://www.medizin.uni-muenster.de/impb/das-institut/cryoem.html>
- Center for Soft Nanoscience
<https://www.uni-muenster.de/SON/>
- Forschungsschwerpunkt „ Zelldynamik und Bildgebung“ an der Universität Münster
<https://www.uni-muenster.de/forschung/profil/schwerpunkt/zelldynamik.html>
- Cells in Motion (CiM) Interfaculty Centre
<https://www.uni-muenster.de/Cells-in-Motion/de/index.html>

Quelle: Pressemitteilung / Pressestelle der Universität Münster (upm)

C64 BASIC V2: Video: 10 selten benutzte BASIC- Features

geschrieben von Andreas Potthoff | 3. Mai 2023

In diesem Video (00:31:19, engl.) erklärt Robin Harbron einige Features von CBM BASIC v2.0 die eigentlich selten benutzt werden. Robin zeigt Tipps und Beispiele zu jedem BASIC-Feature live am C-64.

Folgende Themen werden dabei angesprochen:

1. DEF und FN
2. ON GOSUB / ON GOTO
3. Wissenschaftliche Notation
4. INPUT Möglichkeiten
5. LIST Parameter
6. STOP und END
7. CONT
8. RND()
- 9.USR()
10. WAIT

Forscher

untersuchen

wasserbasiertes Batterie-Recycling

geschrieben von Andreas Potthoff | 3. Mai 2023

Millionen-Förderung für Verbundprojekt: Nachhaltigkeit und zirkuläre Wirtschaft im Fokus

Das Recycling von Lithium-Ionen-Batterien spielt nicht nur aufgrund seiner ökologischen Nachhaltigkeit eine immer größere Rolle in der Batterieforschung, sondern auch aufgrund der Rohstoffknappheit. Während in Deutschland und Europa die Anzahl an Produktionsstätten steigt, fehlt es an natürlichen Vorkommen zahlreicher Ausgangsmaterialien für die Batteriezellproduktion. An diesem Punkt setzt das neue Forschungsprojekt „ProRec“ (kurz für „Neuartige Prozesse während des Recyclings von wässrig prozessierten und zukünftigen Batterien“) an, das das MEET Batterieforschungszentrum der Universität Münster gemeinsam mit den im dortigen Fachbereich Chemie und Pharmazie ansässigen Instituten für betriebswirtschaftliches Management sowie für Anorganische und Analytische Chemie koordiniert. Mithilfe eines wasserbasierten Recyclingverfahrens wollen die Forscherinnen und Forscher Materialien aus wässrig prozessierten – also unter Verwendung von Wasser als Lösungsmittel hergestellten – Elektroden zurückgewinnen und auf ihre Weiternutzung hin analysieren. Das Bundesministerium für Bildung und Forschung fördert das Verbundprojekt über drei Jahre mit 3,2 Millionen Euro.

Ökologische und ökonomische Nachhaltigkeit im Blick

In der Kathodenherstellung für Lithium-Ionen-Batterien werden fluorfreie Binder eingesetzt, die wässrig prozessiert werden

können und damit bereits in der Herstellung umweltfreundlich sind. „Für das Recycling bringen diese Binder neue Möglichkeiten mit sich“, erklärt Dr. Sascha Nowak, einer der Projektkoordinatoren und Leiter des Forschungsbereichs Analytik und Umwelt am MEET. Ziel des Projekts ist es, herauszufinden, wie sich die Wasserlöslichkeit der Binder nutzen lässt, um mittels wasserbasierter Recyclingprozesse sowohl das Aktivmaterial von den Stromsammlern abzulösen als auch die weiteren wasserlöslichen Bestandteile, wie zum Beispiel Lithiumsalze, zu extrahieren.

Um das neue Verfahren mit gängigen Recyclingmethoden vergleichen zu können, werden die wässrig prozessierten Elektroden zusätzlich über eine klassische Prozessroute rezykliert. Anschließend charakterisiert das Wissenschaftsteam die zurückgewonnenen Materialien, bereitet sie auf und untersucht sie im Hinblick auf einen Einsatz in neuen Batteriezellen. Zum Abschluss beurteilen die Forschenden den ökologischen Nutzen sowie die Wirtschaftlichkeit des Verfahrens. „Mit dem Projekt stärken wir die zirkuläre Wirtschaft von Batterien“, sagt Sascha Nowak.

An dem Verbundprojekt beteiligt sind außerdem die Fraunhofer-Einrichtung Forschungsfertigung Batteriezelle FFB in Münster, das Institut für Aufbereitung, Recycling und Kreislaufwirtschaftssysteme der Technischen Universität Clausthal, das Institut für Partikeltechnik der Technischen Universität Braunschweig und das physikalisch-chemische Institut der Justus-Liebig-Universität Gießen.

Links:

- MEET Batterieforschungszentrum an der Universität Münster
<https://www.uni-muenster.de/MEET/>

- Informationen zum Projekt "ProRec"
<https://cris.uni-muenster.de/portal/project/83571213>
 - Meldung aus dem MEET (April 2022): "Neues Verfahren für wässriges Prozessieren von Kathodenmaterialien"
<https://www.uni-muenster.de/MEET/presse/news/aqueous-processing-cathode-materials.shtml>
 - Forschungsschwerpunkt Batterieforschung an der WWU
<https://www.uni-muenster.de/forschung/profil/schwerpunkt/batterieforschung.html>
-

Quelle: Pressemitteilung / Pressestelle der Universität Münster (upm)

Umfrage: Welches C64/C128 Laufwerk/Drive hast Du verwendet?

geschrieben von Andreas Potthoff | 3. Mai 2023

Umfrage:

C64: Welches Laufwerk/Drive hast Du verwendet?

- Floppy: 1540 - 5,25"
- Floppy: 1541 - 5,25"
- Floppy: 1541-II - 5,25"
- Floppy: 1570 - 5,25"
- Floppy: 1571 - 5,25"
- Floppy: 1581 - 3,5"

- Floppy: Espera ESL22 - 5,25"
- Floppy: FD-2000 - 3,5"
- Floppy: OC-118N - 5,25"
- Floppy: REX 9900 - 5,25"
- Floppy: SFD-1002 - 5,25"
- Harddrive: CMD-HD
- Tape: 1530 - MC Tape
- Tape: 1531 - MC Tape

Abstimmen

Ergebnisse anzeigen

 Wird geladen ...

Abstimmungsergebnisse:

C64: Welches Laufwerk/Drive hast Du verwendet?

- Tape: 1530 - MC Tape (23%, 3 Stimmen)
- Floppy: 1541 - 5,25" (23%, 3 Stimmen)
- Floppy: 1541-II - 5,25" (23%, 3 Stimmen)
- Floppy: 1571 - 5,25" (23%, 3 Stimmen)
- Floppy: 1540 - 5,25" (8%, 1 Stimmen)
- Floppy: REX 9900 - 5,25" (0%, 0 Stimmen)
- Floppy: OC-118N - 5,25" (0%, 0 Stimmen)
- Harddrive: CMD-HD (0%, 0 Stimmen)
- Floppy: FD-2000 - 3,5" (0%, 0 Stimmen)
- Floppy: SFD-1002 - 5,25" (0%, 0 Stimmen)
- Floppy: 1581 - 3,5" (0%, 0 Stimmen)
- Floppy: 1570 - 5,25" (0%, 0 Stimmen)
- Tape: 1531 - MC Tape (0%, 0 Stimmen)
- Floppy: Espera ESL22 - 5,25" (0%, 0 Stimmen)

Wähler insgesamt: 3

 Wird geladen ...

Mit aktiven Teilchen Quantenmechanik verstehen

geschrieben von Andreas Potthoff | 3. Mai 2023

Physiker entdecken unerwartete Verbindung zwischen aktiven Teilchen und quantenmechanischen Systemen / Studie in „Nature Communications“

Die Untersuchung von aktiven Teilchen ist eines der am schnellsten wachsenden Teilgebiete der Physik. Als aktive Teilchen bezeichnen Physikerinnen und Physiker Objekte, die sich durch einen internen Antrieb von alleine fortbewegen. Dazu zählen Lebewesen wie schwimmende Bakterien und Fische, fliegende Vögel oder herumlaufende Menschen, aber auch künstliche Nanoroboter, die zum Beispiel für den Medikamententransport im Körper eingesetzt werden können. Insbesondere interessieren sich die Fachleute für das Verhalten von Systemen aus vielen aktiven Teilchen, um hierdurch beispielsweise Vogelschwärme, Biofilme oder Menschenansammlungen zu verstehen. Die Physiker Dr. Michael te Vrugt, Tobias Frohoff-Hülsmann, Prof. Dr. Uwe Thiele und Prof. Dr. Raphael Wittkowski vom Institut für Theoretische Physik der Westfälischen Wilhelms-Universität (WWU) Münster haben nun

in Zusammenarbeit mit Prof. Dr. Eyal Heifetz von der Universität Tel Aviv (Israel) ein neues Modell („active model I+“) für die Dynamik von Systemen aus vielen aktiven Teilchen entwickelt. Die Studie ist in der Fachzeitschrift „Nature Communications“ veröffentlicht.

„Dieses Modell beschreibt insbesondere Teilchen, auf die nur geringe Reibungskräfte wirken, ein bislang nur wenig untersuchter Fall“, erklärt Erstautor Michael te Vrugt. Hierbei hat das Team festgestellt, dass dieses Modell für bestimmte Parameterwerte genauso aussieht wie die Schrödingergleichung. Die Schrödingergleichung ist die Grundgleichung der Quantenmechanik, welche das Verhalten von extrem kleinen Teilchen wie Elektronen oder Protonen beschreibt. Durch diese Analogie ist es möglich, in aktiven Systemen Analogien zu aus der Quantenmechanik bekannten Effekten zu finden. Die Physiker untersuchten in der aktuellen Arbeit zum einen den Tunneleffekt und zum anderen dunkle Materie.

Der Tunneleffekt ist ein quantenmechanisches Phänomen, bei dem ein Teilchen durch eine Barriere hindurchdringt („tunnelt“), obwohl es dafür eigentlich zu wenig Energie hat. Dieser Effekt spielt eine Rolle beim radioaktiven Zerfall, ist aber auch beispielsweise für den Speichervorgang in USB-Sticks wichtig. Die Autoren konnten nun zeigen, dass sich die Dichteverteilung von aktiven Teilchen, die mit einem Laserstrahl beleuchtet werden, in etwa wie die Wahrscheinlichkeitsverteilung eines quantenmechanischen Teilchens beim Tunneleffekt verhält.

Dunkle Materie ist eine Form von Materie, die nicht mit sichtbarem Licht wechselwirkt und deren Zusammensetzung bislang nicht verstanden ist, von deren Existenz man aber aus einer Vielzahl astronomischer Beobachtungen weiß. In der Studie wies das Team nun durch einen Vergleich der entsprechenden mathematischen Modelle nach, dass sich elektrisch geladene aktive Teilchen ähnlich wie dunkle Materie verhalten. „Dies eröffnet eine Möglichkeit, kosmologische

Strukturbildungsprozesse im Labor nachzustellen“, kommentiert Raphael Wittkowski.

Finanzierung

Die Promotionen von Michael te Vrugt und Tobias Frohoff-Hülsmann wurden durch die Studienstiftung des deutschen Volkes unterstützt. Die Arbeitsgruppe Wittkowski erhält finanzielle Unterstützung durch die Deutsche Forschungsgemeinschaft (DFG, Project-ID 433682494 – SFB 1459).

Originalpublikation

M. te Vrugt, T. Frohoff-Hülsmann, E. Heifetz, U. Thiele, R. Wittkowski (2023). From a microscopic inertial active matter model to the Schrödinger equation. *Nature Communications* 14, 1302; DOI: 10.1038/s41467-022-35635-1

Links:

- Originalveröffentlichung in “Nature Communications”
<https://doi.org/10.1038/s41467-022-35635-1>
 - AG Wittkowski an der WWU
<https://www.uni-muenster.de/Physik.TP/research/wittkowski/>
 - AG Thiele an der WWU
<https://www.uni-muenster.de/Physik.TP/research/thiele/>
 - AG Heifetz an der Universität Tel Aviv
<https://eyalheifetz.weebly.com/>
-

Quelle: Pressemitteilung / Pressestelle der Universität Münster (upm)