*** SCP-Beschreibung für A5105 - Erweiterung SUBM ***

Erweiterungsmöglichkeiten für das SUBM-Kommando

1. Vorbemerkungen

Das transiente Kommando SUBM ist im SCP-Handbuch Teil 1 Abschnitt 4.2.5. beschrieben. Es dient zur automatischen Abarbeitung von Befehlsfolgen im CCP-Status. Die Kommandofolge, die alle residenten und transienten Kommandos
enthalten kann, ist mit TP als Programmdatei name.SUB zu erfassen. Durch das Kommando SUBM wird diese Datei in eine temporäre Datei $$$.SUB umgewandelt,
welche anschließend vom CCP abgearbeitet wird. Die Kommandoeingaben werden
nicht mehr von der Tastatur, sondern von dieser Datei entnommen.
Durch Verwendung von Variablen $1 bis $9 kann die Kommandodatei in gewissem Grade allgemein gestaltet werden, jedoch bleiben noch zahlreiche Wünsche für die effektive Gestaltung einer programmierten Kommandofolge offen.

- Es ist nicht möglich, in einer Kommandofolge Kommandozeilen zu überspringen.
- Es ist nicht möglich, eine zweite Kommandofolge als Unterprogramm aufzurufen.
- Die Datei $$$.SUB muß auf Laufwerk A angelegt werden, das bringt bei Benutzung eines RAM-Floppy nicht den gewünschten Zeitgewinn.

2. Überspringen von Kommandozeilen

Oft besteht die Notwendigkeit, nur bestimmte Teile einer Kommandodatei abzuarbeiten. Das war bisher nur durch Verwendung von mehreren fast gleichen Kommandodateien möglich. Änderungen mußten dann immer in allen Dateien durchgeführt werden. Es war einfacher, die erforderliche Befehlsfolge von Hand einzugeben.
Zum wahlweisen Überspringen von Kommandozeilen dient das transiente Kommando LOE (löschen). Vom Bediener wird eine Tastatureingabe angefordert. Durch Drücken der Taste **SPACE** werden die nachfolgenden Kommandozeilen abgearbeitet, bei **ESC** werden sie gelöscht (übersprungen).
Eine entsprechende Handlungsaufforderung kann in der Kommandodatei als Kommentarzeile stehen.

Syntax: LOE zahl

zahl ist eine Dezimalzahl und gibt die Anzahl der nachfolgenden zu löschenden Zeilen an. Das Löschen wird nur ausgeführt, wenn die Taste E**SC** gedrückt wurde.

Beispiel:
```
;Kommandofile zur bedingten Uebersetzung
;Soll Programm 1 uebersetzt werden ?
;JA --> SPACE           NEIN --> ESC
LOE 1
ASM =PROG1
;Soll Programm 2 uebersetzt werden ?
;JA --> SPACE           NEIN --> ESC
LOE 1
ASM =PROG2
;Sollen die Programme gelinkt werden ?
;JA --> SPACE           NEIN --> ESC
LOE 1
LINK PROG1,PROG2,PROG/N/E
```

3. Aufruf einer Kommandodatei als Unterprogramm

3.1. Realisierungsmöglichkeiten

Eine Kommandodatei muß den Namen $$$.SUB besitzen, um vom CCP anerkannt zu werden. Um eine Unterprogrammtechnik zu realisieren, insbesondere um nach Beendigung eines Unterprogrammes in das aufrufende Programm an die richtige Stelle zurückzukehren, müssen mehrere solcher Dateien vorhanden sein, die alle den Namen $$$.SUB besitzen.
Diese Dateien können entweder auf verschiedenen Laufwerken oder auf verschiedenen USER-Bereichen durch das Kommando SUBM angelegt werden.
Nach der Einstellung des gewünschten aktuellen Laufwerkes oder USER-Bereiches ist das Kommando SUBM innerhalb einer Kommandodatei aufzurufen, um das "Unterprogramm" bzw. Unterkommandofile zu erzeugen.
Durch geeignete Maßnahmen ist dem Rechner anzuweisen, daß die Kommandodatei eines anderen Laufwerkes bzw. USER-Bereiches ab sofort zu verwenden ist.
Dazu werden im Folgenden mehrere Möglichkeiten beschrieben.
Der Wechsel des Systemlaufwerkes ist einfacher zu handhaben, weil gegebenenfalls durch Angabe der Laufwerke vor den Dateibezeichnungen noch alle Dateien ansprechbar sind. Beim USER-Wechsel sind die Dateien in anderen USER-Bereichen nicht mehr problemlos erreichbar.

3.2. Unterprogrammverschachtelung durch Wechsel des Systemlaufwerkes

3.2.1. aktuelles Laufwerk und Systemlaufwerk

Vom CCP wird die Datei $$$.SUB zuerst auf dem aktuellen USER des Systemlaufwerkes (unabhängig vom aktuellen Laufwerk) gesucht, wenn sie sich dort nicht befindet, kann auch eine im USER 0 des Systemlaufwerkes befindliche Datei $$$.SUB unter bestimmten Bedigungen abgearbeitet werden.
Das Systemlaufwerk ist standardmäßig immer das Laufwerk A. Dieses Laufwerk wird nach jedem Warmstart in den Zustand ON LINE versetzt, auch wenn es nicht das aktuelle Laufwerk ist. Außerdem werden auf dem Systemlaufwerk die transienten Kommandos gesucht, die auf dem aktuellen Laufwerk nicht vorhanden sind (entsprechend Vorwort zum Abschnitt 4.2. SCP-Handbuch).
Das aktuelle Laufwerk wird im CCP-Status am Anfang jeder Kommandozeile angezeigt und kann durch Eingabe von z.B.

```
A>B:
```

auch innerhalb einer Kommandodatei verändert werden.
Eine Anzeige und Umschaltung des Systemlaufwerkes ist durch das transiente Kommando SYSDV möglich. Soll das zuvor aktuelle Systemlaufwerk wieder eingestellt werden, ist das Kommando SUBRET zu verwenden. Damit ist eine mehrfache Verschachtelung möglich.

3.2.2. Die Kommandos SYSDV und SUBRET

Syntax: SYSDV [lw:]

Das angegebene Laufwerk wird zum Systemlaufwerk. Wird kein Laufwerk angegeben, dann erfolgt eine Anzeige des aktuellen Systemlaufwerkes.

Syntax: SUBRET

Durch dieses Kommando wird das vor dem letzten SYSDV-Kommando aktuelle Systemlaufwerk wieder eingestellt.

Mit diesem Kommando können als Unterprogramm aufgerufene Kommandofiles beendet werden, um eine Rückkehr zum aufrufenden Kommandofile zu ermöglichen.

3.2.3. Unterprogrammaufruf

Im folgenden Beispiel soll die Übersetzung von 2 Dateien aus Abschnitt 2 mit Unterprogrammtechnik durchgeführt werden. Dazu wurden vorher die Kommandofiles HP.SUB und UP.SUB mit TP erfaßt. Zum Aufruf muß das aktuelle Laufwerk gleich dem Systemlaufwerk sein.

```
SUBM HP
```

Falls sich die Dateien SUBM.COM oder HP.SUB nicht auf diesem Laufwerk befinden, kann das entsprechende Laufwerk mit angegeben werden.

```
lw:SUBM lw:HP
```

Beispiel: Kommandodatei HP.SUB (Hauptprogramm)

```
;Aufruf Kommandofile als Unterprogramm
D:
SUBM A:UP PROG1
A:
SYSDV D:
D:
SUBM A:UP PROG2
A:
SYSDV D:
LINK PROG1,PROG2,PROG/N/E
```

Kommandodatei UP.SUB (Unterprogramm)

```
;Unterprogramm
ASM =$1
SUBRET
```

Funktionsweise:

Nach Umschaltung des aktuellen Laufwerkes wird mit SUBM die Kommandodatei $$$.SUB des Unterprogrammes auf dem Laufwerk D: angelegt, aber die Abarbeitung noch nicht gestartet. Als Parameter $1 wird PROG1 bzw. beim zweiten Aufruf des Unterprogrammes PROG2 übergeben. Danach wird als aktuelles Laufwerk A: eingestellt. Bis zu diesem Zeitpunkt werden alle Kommandos aus der zum Hauptprogramm gehörigen Kommandodatei, die sich im aktuellen Systemlaufwerk befindet, entnommen. Die zum Unterprogramm gehörige Kommandodatei liegt auf dem Laufwerk D: bereit und wird durch das Kommando SYSDV D: aktiviert.
Das nächste Kommando ist die Kommentarzeile aus dem Unterprogramm. Nach Abarbeitung des Assemblers (der Parameter $1 wird jeweils ersetzt) erfolgt mit SUBRET die Rückschaltung des Systemlaufwerkes und die Abarbeitung des Hauptprogrammes wird fortgesetzt. Dort wiederholt sich der Aufruf des Unterprogrammes mit einem anderen Parameter. Zuletzt wird mit LINK ein ausführbares Programm erzeugt.

```
A>subm hp

A>AUFRUF KOMMANDOFILE ALS UNTERPROGRAMM
A>D:
D>SUBM A:UP PROG1

D>A:
A>SYSDV D:

A>;UNTERPROGRAMM
A>ASM =PROG1

No Fatal error(s)

A>SUBRET

A>D:
D>SUBM A:UP PROG2

D>A:
A>SYSDV D:

A>;UNTERPROGRAMM
A>ASM =PROG2

No Fatal error(s)

A>SUBRET

A>LINK PROG1,PROG2,PROG/N/E

LINK RBWS(SCPX) V1/1

Data    0103    0105    <    2>

47205 Bytes Free
_[0000   0105        1]

A>
```

3.3. Unterprogrammverschachtelung durch Wechsel des aktuellen USER

Die grundlegende Funktionsweise wurde bereits in den Abschnitten 3.1. und 3.2.1. erklärt.
Das residente Kommando USER kann innerhalb einer Kommandodatei zum Umschalten des aktuellen USER-Bereiches benutzt werden. Eine im USER 0 befindliche Kommandodatei $$$.SUB wird in jedem anderen USER zunächst weiter abgearbeitet (falls sich dort keine andere Datei $$$.SUB befindet).
Durch Aufruf des Kommandos SUBM kann dort die Kommandodatei des Unterprogrammes erzeugt werden, die sofort abgearbeitet wird. Nach dem Ende dieser Unterprogramm-Datei bleibt die Abarbeitung stehen, weil eine leergewordene $$$.SUB-Datei CCP-intern als Ende des Kommandofiles gewertet wird. Ein Fortsetzen wäre durch Eingabe von ^C möglich oder man sorgt mit

```
ERA $$$.SUB
```

als letztem Befehl im Unterprogramm dafür, daß die Datei vor dem Leerwerden gelöscht wird. Durch diese Maßnahmen kann die Datei $$$.SUB im USER 0 weiter bearbeitet werden.
Ein Stehenbleiben der Abarbeitung tritt außerdem auf, wenn in einem anderen aktuellen USER Kommandos aus der Kommandodatei des USER 0 abgearbeitet werden und mit einem Warmstart enden. Das ist darauf zurückzuführen, daß beim Warmstart vom CCP die BDOS-Funktion 13 "Rücksetzen

Diskettensystem" (siehe SCP-Handbuch Teil 2 Abschnitt 9.4.) ausgeführt wird und vom BDOS das Vorhandensein (bzw. Nichtvorhandensein) einer Batchmode-Datei gemeldet wird.
Die Batchmode-Datei muß auf dem aktuellen USER im Systemlaufwerk vorhanden sein und der Name muß mit einem "$"-Zeichen beginnen. Somit ist die Datei $$$.SUB eine Batchmode-Datei.
Die Abarbeitung bleibt also deshalb stehen, weil sich im aktuellen USER kein Dateiname befindet, der mit einem "$"-Zeichen beginnt. Das Problem kann einfach gelöst werden, indem eine leere Datei z.B. mit dem Kommando

```
SAVE 0 $
```

erzeugt wird. Das BDOS wird dadurch überlistet und meldet eine vorhandene Batchmode-Datei im aktuellen USER. Das CCP sucht infolgedessen eine Datei mit dem Namen $$$.SUB, die aber nicht im aktuellen USER, sondern im USER 0 gefunden wird.
Nach der Überwindung dieser Probleme muß nur noch dafür gesorgt werden, daß sich die benötigten Dateien jeweils im richtigen USER befinden. Ein Kopieren aus einem anderen USER in den aktuellen USER ist mit

```
PIP zieldatei=quelldatei [Gn]
```

möglich. n ist die USER-Nummer der Quelldatei (siehe Abschnitt 4.2.4. SCP-Handbuch). Die transienten Kommandos sollten sich alle im USER 0 befinden, denn dort werden sie auch von allen anderen USER-Bereichen gefunden.

Um Ordnung auf der Diskette zu halten sollten nicht mehr benötigte "$"-Dateien und durch Kopieren auf mehreren USER-Bereichen befindliche Dateien wieder gelöscht werden.

Nachfolgend wird das schon bekannte Beispiel mit USER-Umschaltung demonstriert. Alle benötigten Dateien HP2.SUB, UP2.SUB, PROG1.MAC, PROG2.MAC und die Programme SUBM.COM, PIP.COM, ASM.COM und LINK.COM befinden sich auf Laufwerk A: USER 0. Der Aufruf erfolgt mit:

```
A>SUBM HP2
```

Beispiel: Kommandodatei HP2.SUB (Hauptprogramm)

```
;Aufruf Unterprogramm mit USER-Umschaltung
USER 1
SAVE 0 $$
PIP UP2.SUB=UP2.SUB_[G0]
PIP A:=PROG?.MAC_[G0]
SUBM UP2 PROG1
SUBM UP2 PROG2
ERA PROG?.MAC
USER 0
PIP A:=PROG?.REL_[G1]
LINK PROG1,PROG2,PROG/N/E
```

Kommandodatei UP2.SUB (Unterprogramm)

```
;Unterprogramm
ASM =$1
ERA $$$$$$.SUB
```

Funktionsweise:

Durch das Aufrufkommando wird im USER 0 die Datei $$$.SUB des Hauptprogrammes abgelegt.

Nach Umschaltung wird im USER 1 die Datei "$" angelegt, um das BDOS zu überlisten und mit PIP werden die benötigten Dateien aus dem USER 0 "geholt".
SUBM startet das Unterprogramm durch Anlegen der Datei $$$.SUB im USER 1, nach Abarbeitung des Assemblers ist diese Datei zu Ende und wird wieder gelöscht. Von der Datei $$$.SUB aus dem USER 0 erfolgt der 2. Aufruf des Unterprogrammes mit dem Parameter PROG2.
Der aktuelle USER ist noch USER 1, jetzt können die nicht mehr benötigten Dateien gelöscht werden. Im Beispiel werden nur die Dateien PROG1.MAC und PROG2.MAC gelöscht, es ist aber auch möglich die Dateien UP2.SUB und $ zu löschen. Die Ergebnisse der Abarbeitung des Assemblers heißen PROG1.REL und PROG2.REL und befinden sich ebenfalls im USER 1.
Nach Rückschaltung auf den USER 0 müssen diese .REL-Dateien in den USER 0 geholt werden, um mit LINK weiterverarbeitet zu werden.
Die Abarbeitung des Kommandos LINK stellt dann keine Besonderheit mehr dar.

Die Abarbeitung dieser Kommandodatei ergibt folgende Bildschirmanzeige:

```
A>subm hp2

A>;AUFRUF UNTERPROGRAMM MIT USER UMSCHALTUNG
A>USER 1
A1>SAVE 0 $
A1>PIP UP2.SUB=UP2.SUB_[G0]

A1>PIP A:=PROG?.MAC_[G0]

COPYING -
PROG1.MAC
PROG2.MAC

A1>SUBM UP2 PROG1

A1>;UNTERPROGRAMM
A1>ASM =PROG1

No Fatal error(s)

A1>ERA $$$.SUB
A1>SUBM UP2 PROG2

A1>;UNTERPROGRAMM
A1>ASM =PROG2

No Fatal error(s)

A1>ERA $$$.SUB
A1>ERA PROG?.MAC
A1>USER 0
A>PIP A:=PROG?.REL_[G1]

COPYING -
PROG1.REL
PROG2.REL

A>LINK PROG1,PROG2,PROG/N/E

LINK RBWS(SCPX) V1/1

Data    0103    0105    <    2>

47669 Bytes Free
_[0000   0105        1]

A>
```

Bemerkung zu den Beispielen:

Die gewählten Beispiele lösen nur ein sehr einfaches Problem, die Übersetzung zweier Assemblerprogramme und die Erzeugung einer .COM-Datei.
Mit Sicherheit ist für die Lösung dieses Problems der Aufwand der Unterprogrammtechnik zu hoch und uneffektiv. Jedoch sollte mit diesen noch relativ leicht verständlichen Beispielen die Unterprogrammtechnik demonstriert werden und zu eigenen Anwendungen anregen.

Leicht vorstellbar ist der Start einer Kommandodatei mit dem Kaltstartkommando SUBM AUTOEXEC und eine anschließende menügesteuerte Bedienerführung auf CCP-Ebene. Damit wird im Betriebssystem SCPX 5105 eine voll programmierte Arbeitsweise möglich.

Bei der Übersetzung von nachladbaren Treiberprogrammen mit dem Kommandofile TR.SUB als Unterprogramm und der Möglichkeit des Überspringens von Kommandozeilen entsprechend Abschnitt 2. lassen sich mit einem Kommandofile wahlweise verschiedene Varianten eines Treibers erstellen. So sind z.B. die 4 Varianten des RAM-Floppy Treiberprogrammes entstanden. Das dazu benutzte Kommandofile wird im Folgenden abgedruckt:

```
;            Uebersetzen Bildschirm RAM-Floppy ohne CRC ?
LOE 7
PIP RFIF.MAC=RFIFV.MAC
D:
SUBM A:TR A:$1 $2
A:
SYSDV D:
ERA RFV.COM
REN RFV.COM=$1.COM
;            Uebersetzen Bildschirm RAM-Floppy mit CRC ?
LOE 7
PIP RFIF.MAC=RFIFVC.MAC
D:
SUBM A:TR A:$1 $2
A:
SYSDV D:
ERA RFVC.COM
REN RFVC.COM=$1.COM
;            Uebersetzen Modul RAM-Floppy ohne CRC ?
LOE 7
PIP RFIF.MAC=A:RFIFM.MAC
D:
SUBM A:TR A:$1 $2
A:
SYSDV D:
ERA RFM.COM
REN RFM.COM=$1.COM
;            Uebersetzen Modul RAM-Floppy mit CRC ?
LOE 7
PIP RFIF.MAC=RFIFMC.MAC
D:
SUBM A:TR A:$1 $2
A:
SYSDV D:
ERA RFMC.COM
REN RFMC.COM=$1.COM
;            Quelldateien kopieren ?
LOE 1
D:POWER COPY *.MAC
```

4. Hinweise zur Nutzung eines RAM-Floppy

Die Benutzung eines RAM-Floppy bringt in erster Linie Zeitgewinn durch die erhebliche Verkürzung der Zugriffszeiten auf die Daten der Diskette. Außerdem stellt es ein zusätzliches Laufwerk dar, was für viele Anwendungsfälle Voraussetzung ist.
Beim Bildungscomputer A 5105 steht ein Treiberprogramm für ein RAM-Floppy im Bildspeicher zur Verfügung. Dieses kann im Betriebssystem SCPX 5105 grundsätzlich immer geladen werden. Außerdem kann durch Stecken eines RAM-Floppy-Moduls eine zusätzliche Diskette von 256 KByte in das System eingebunden werden.
Beim Warmstart und von einigen Anwenderprogrammen erfolgt ein Zugriff auf das Laufwerk A:. Dieser Zugriff tritt gerade bei der Abarbeitung von Kommandofiles häufig auf und reduziert den Zeitgewinn erheblich. Außerdem ist bei der Arbeit mit RAM-Floppy ein Verzicht auf die Diskette im Laufwerk A: wünschenswert.
Aus diesem Grunde ist es günstig das Laufwerk A: mit dem RAM-Floppy logisch zu vertauschen. Dafür gibt es das residente Kommando

```
SWAP lw1: lw2:
```

Damit sind beliebige vorhandene Laufwerke vertauschbar. Bei diesem Kommando treten einige Verwirrungen in der Statuszeile auf, deshalb wurde ein transientes Kommando für den selben Zweck bereitgestellt.

```
SW lw:
```

Damit ist nur ein Vertauschen eines Laufwerkes mit dem Laufwerk A: möglich.
Ist Laufwerk A: bereits vertauscht, kann nur zurückgetauscht werden, ein Vertauschen mehrerer Laufwerke wird abgelehnt.

Das folgende Beispiel zeigt ein Kommandofile, das als Kaltstartkommando verwendet werden kann, wenn es den Dateinamen AUTOEXEC.SUB erhält.

Beispiel:
```
RFVC
D:
SUBM A:AUTO1 D:
A:
SW D:
```

Zuerst wird der Treiber für Bildschirm RAM-Floppy gestartet. Dann wird auf dem neuen Laufwerk D: mit SUBM aus der Datei AUTO1.SUB eine Datei $$$.SUB erzeugt, aber noch nicht abgearbeitet.
Mit SW D: werden die Laufwerke A: und D: vertauscht. Das Systemlaufwerk (von dem die Datei $$$.SUB abgearbeitet wird) heißt weiterhin A:, ist also jetzt das RAM-Floppy. Dort befindet sich bereits die zu AUTO1 gehörende Datei $$$.SUB.
Die folgenden Kommandos werden also mit wesentlich höherer Geschwindigkeit vom RAM-Floppy gelesen. Deshalb sollte die Umschaltung so zeitig wie möglich erfolgen und weitere Kommandos sollten in der Datei AUTO1.SUB erst abgearbeitet werden. Diese Datei kann z.B. folgenden Inhalt haben:

```
$1PIP A:=$1PIP.COM_[R]
GO A:=$1LOE.COM[R]
;Alle Dateien fuer Kommandofile kopieren ?
;JA --> SPACE     NEIN --> ESC
LOE 8
PIP A:=$1HP*.SUB
GO A:=$1UP*.SUB
GO A:=$1PROG?.MAC
GO A:=$1SUBM.COM[R]
```

```
GO A:=$1ASM.COM[R]
GO A:=$1LINK.COM[R]
GO A:=$1SYSDV.COM[R]
GO A:=$1SUBRET.COM[R]
;TP kopieren ?
;JA --> SPACE     NEIN --> ESC
LOE 2
PIP A:=$1TPLX86.COM[R]
GO A:=$1TP*.OVR[R]
```

Die Kommentarzeilen und LOE-Kommandos werden aufgrund des RAM-Floppys sehr schnell abgearbeitet. Der Zugriff auf die Diskette erfolgt nur noch, um die zu kopierenden Dateien zu lesen.
Falls die Dateien zur Abarbeitung der Kommandofiles kopiert wurden, kann anschließend der Aufruf mit SUBM direkt im RAM-Floppy erfolgen. Es wurden die in den vorangegangenen Abschnitten dieser Beschreibung gewählten Beispiele auch hier benutzt.

```
SUBM HP
```

startet das Beispiel mit Umschaltung des Systemlaufwerkes. Daß die Laufwerke bereits vertauscht sind stört nicht. Zum Ablegen des Unterprogramm-Kommandofiles wird hier die Diskette noch benötigt.

Der Zeitgewinn bei der Arbeit mit RAM-Floppy wird erst im vollen Umfang sichtbar, wenn ausschließlich damit gearbeitet wird. Das Beispiel mit USER-Umschaltung verdeutlicht das eindrucksvoll.

```
SUBM HP2
```

startet das Programmbeispiel aus Abschnitt 3.3. Der Programmdurchlauf dauert

Diskette	133 Sekunden
Bildschirm RAM-Floppy mit CRC	29 Sekunden
Bildschirm RAM-Floppy ohne CRC	16 Sekunden
Modul RAM-Floppy mit CRC	23 Sekunden
Modul RAM-Floppy ohne CRC	14 Sekunden