1 Einleitung

Der universelle Transaktionsmonitor *open*UTM bietet als umfassende Middleware-Plattform alle Möglichkeiten, die Sie für das Design und den Einsatz transaktionsorientierter OLTP-Anwendungen benötigen. Darüber hinaus ist in *open*UTM die Funktionalität eines kompletten Message Queuing Systems integriert.

Durch optimale Performance, ausgereifte Security-Funktionen und höchste Verfügbarkeit ist *open*UTM auch für Anwendungsszenarien geeignet, bei denen herkömmliche OLTP-Systeme längst an ihre Grenzen stoßen.

*open*UTM bildet ein sicheres und effizientes Framework für moderne multi-tier Client/Server-Architekturen: *open*UTM sorgt für die Steuerung globaler Transaktionen, optimiert den Einsatz von System-Ressourcen (Arbeitsspeicher, CPU etc.), übernimmt das Management von parallelen Zugriffen, kümmert sich um Zugangs- und Zugriffskontrollen, den Aufbau von Netzverbindungen und vieles mehr.

Der Name „*open*UTM“ weist bereits auf wichtige Leistungsmerkmale hin:

open ... weil *open*UTM dem von X/Open definierten Referenzmodell für „Distributed Transaction Processing (DTP)“ entspricht und die von X/Open standardisierten offenen Schnittstellen unterstützt.

Universal ... weil *open*UTM unterschiedliche Welten verbindet und für die unterschiedlichsten Einsatzszenarien konzipiert ist: *open*UTM integriert heterogene Netze, Plattformen, Resource Manager und Anwendungen.

Transaction ... weil *open*UTM anwendungsübergreifend volle Transaktionssicherheit gewährleistet, entsprechend den klassischen ACID-Eigenschaften Atomicity, Consistency, Isolation und Durability.

Monitor ... weil *open*UTM nicht „nur“ Transaktionsverarbeitung bietet, sondern das Management von verteilten, unternehmensweiten IT-Lösungen ermöglicht.

1.1 Konzept und Zielgruppen dieses Handbuchs

Dieses Handbuch soll Assembler-Programmierer von *open*UTM-Anwendungen in ihrer Arbeit unterstützen. Es ergänzt das Basishandbuch „Anwendungen programmieren mit KDCS für COBOL, C und C++“.

Zu seinem Verständnis sind Grundkenntnisse des Betriebssystems und von *open*UTM sowie das Basishandbuch „Anwendungen programmieren mit KDCS für COBOL, C und C++“ notwendig. Zur Vertiefung der gebotenen Informationen können die *open*UTM-Handbücher „Anwendungen generieren und betreiben“, „Meldungen, Test und Diagnose“, sowie „Konzepte und Funktionen“ herangezogen werden.

Dieses Handbuch beschreibt die sprachspezifischen Besonderheiten bei der Erstellung von Assembler-Teilprogrammen.

Es enthält ein in Assembler erstelltes Beispiel zum KDCS-Aufruf INIT.

Im Kapitel „Assembler-Datenstrukturen“ auf Seite 17 finden Sie eine Aufstellung aller Assembler-Datenstrukturen.

Readme-Datei

Sie finden die Informationen auf Ihrem BS2000-Rechner entweder in der Freigabemitteilung (Dateiname SYSFGM.*produkt.version.sprache)* oder einer Readme-Datei (Dateiname SYSRME.*produkt.version.sprache)*. Die Benutzerkennung, unter der sich die Datei befindet, erfragen Sie bitte bei Ihrer zuständigen Systembetreuung. Die Datei können Sie mit dem Kommando /SHOW-FILE oder mit einem Editor ansehen oder auf einem Standarddrucker mit folgendem Kommando ausdrucken:

```
/PRINT-DOCUMENT dateiname, LINE-SPACING=*BY-EBCDIC-CONTROL
```

bei SPOOL -Versionen kleiner 3.0A:

```
/PRINT-FILE FILE-NAME=dateiname,LAYOUT-CONTROL=
    PARAMETERS(CONTROL-CHARACTERS=EBCDIC)
```

2 Assembler-Teilprogramme

UTM-Teilprogramme können Sie in Assembler schreiben:

– Teilprogramme, die nicht ICLS-fähig sind, definieren Sie bei der Generierung mit der KDCDEF-Steueranweisung PROGRAM ...,COMP=ASSEMB.

– Bei Teilprogrammen, die ILCS (Inter Language Communication Services) unterstützen, müssen Sie PROGRAM ...,COMP=ILCS angeben (siehe *open*UTM-Handbuch „Anwendungen generieren und betreiben“).

Für das Erstellen von Assembler-Teilprogrammen stellt *open*UTM Makroaufrufe zur Verfügung. Mit Hilfe der Makroaufrufe können Sie:

– KDCS-Funktionen aufrufen
– Verknüpfungskonventionen einhalten
– die Programmierung vereinfachen

Die Makroaufrufe werden in der Bibliothek SYSLIB.UTM.050.ASS bereitgestellt.

Die Makroaufrufe ZSTRT, ZCALL und ZEND erzeugen keinen Shared Code. Im Abschnitt „Shareable Assembler-Module“ auf Seite 10 finden Sie Hinweise, wie shareable Teilprogramme in Assembler zu schreiben sind.

Die Verwendung der oben genannten Makroaufrufe ist nur dann notwendig, wenn Sie kompatible KDCS-Programme erstellen wollen.

Wollen Sie Assemblerprogramme ILCS-fähig machen, gibt es zwei Möglichkeiten, siehe dazu unter Tabelle „Anmerkungen zur Tabelle:“ auf Seite 4, (zu ILCS-Konventionen siehe CRTE-Handbuch).

Über ILCS lassen sich Teilprogramme aus mehreren Quellcodes unterschiedlicher Programmiersprachen verbinden. Bei der Übergabe von Parametern bzw. Zugriff auf gemeinsame Datenstrukturen muß unbedingt sichergestellt sein, daß die Datendarstellungen identisch sind. Im Handbuch „Anwendungen generieren und betreiben“ für *open*UTM (BS2000/OSD) finden Sie eine Auflistung aller Compiler und Laufzeitsysteme, die eine Mischung erlauben.

Ab UTM V3.1 setzt UTM beim Aufruf eines Teilprogramms in der letzten Adresse der in Register 1 übergebenen Parameterliste das Bit 2**7 ("Stop-Bit") nicht mehr.

2.1 Compiler, Laufzeitsysteme und Generierungsoptionen

Die folgende Tabelle zeigt die Compiler, Laufzeitsysteme und Generierungsoptionen, die Sie nutzen können, um Assembler-Teilprogramme zu erzeugen und in einem UTM-Anwenderprogramm ablaufen zu lassen:

– Versionen des Compilers, die Sie benutzen können, um die Objekte (OM oder LLM) eines UTM-Teilprogramms zu erzeugen

– Versionen des Laufzeitsystems, in denen diese Teilprogramme reibungslos ablaufen können

– Werte für den Operanden COMP der Steueranweisung PROGRAM, die Sie bei der KDCDEF-Generierung angeben müssen, um diese Teilprogramme in die Anwendungskonfiguration aufzunehmen.

Assembler-Compiler	**Laufzeitsystem**	**PROGRAM..., COMP=**	
ASSGEN	—	ASSEMB	1.
ASSEMB ≥ V30	—	ASSEMB	1.
ASSEMBH V1.0 bis V1.2	—	ASSEMB	1.
ASSEMBH V1.2A	ASSEMBH V1.2A	ILCS	2.

Anmerkungen zur Tabelle:

1. Wenn Sie COMP=ASSEMB angeben, dann dürfen Sie das ASSEMBH-Laufzeitsystem **nicht** benutzen. Der Grund dafür ist, daß das Laufzeitsystem von ASSEMBH ab Version V1.1 das ILCS nutzt. Es kommt dann zu einer verbotenen Mischung von Nicht-ILCS- mit ILCS-Programmteilen.

2. Das Assembler-Programm **muß** ILCS-fähig sein.
   Es gibt zwei Möglichkeiten, ein Assembler-Programm ILCS-fähig zu machen:

   – Sie benutzen die Assembler-Makros ZSTRT, ZCALL und ZEND in der Ausprägung ZSTRT ILCS=YES. Bitte beachten Sie, daß die Ausprägung ZSTRT ILCS=NO (nicht-ILCS-fähig) Standardwert ist!

   – Sie benutzen die Makros @ENTR ... ILCS=YES..., @PASS und @EXIT (siehe auch Handbuch „ASSEMBH“).

   Der Compiler und das Laufzeitsystem müssen mindestens den Korrekturstand 10 haben.

2.2 Aufbau eines Assembler-Teilprogramms

Für die Programmierung von Assembler-Teilprogrammen sind dieselben Regeln zu beachten, wie bei Teilprogrammen in anderen Programmiersprachen:

– Der erste UTM-Aufruf muß ein INIT sein.

– Der Ablauf des Teilprogramms endet mit dem UTM-Aufruf PEND. Danach erhält das Teilprogramm die Steuerung nicht mehr zurück.

– Jedes Teilprogramm, das einen Dialogschritt abschließt, muß einen MPUT enthalten.

– Der Code muß seriell wiederverwendbar sein (siehe auch Basishandbuch „Anwendungen programmieren mit KDCS für COBOL, C und C++“).

Programmbeginn

Assembler-Teilprogramme sind Unterprogramme der Main-Routine KDCROOT.

Damit die Verknüpfungskonventionen eingehalten werden, muß jedes Teilprogramm mit dem Makro ZSTRT beginnen.

Außerdem müssen Sie den Kommunikationsbereich (KB) und den Standard-Primären Arbeitsbereich (SPAB) adressierbar machen, sowie eventuell weitere gemeinsam nutzbare Bereiche, die mit der KDCDEF-Anweisung AREA definiert wurden (siehe *open*UTM-Handbuch „Anwendungen generieren und betreiben“).

UTM übergibt die Adressen der Parameter in einer Parameterliste. Deren Adresse steht in Register 1.

Wort 1: Adresse des KB

Wort 2: Adresse des SPAB

Wort 3: Adresse des 1. gemeinsamen Bereichs

.
.
.

Wort *n*+2: Adresse des *n*-ten gemeinsamen Bereichs

Beispiel

Der Beginn eines Teilprogramms könnte folgendermaßen aussehen:

```
prgnam  ZSTRT BASIS=r1,REGS=n,PARM=savpar
           USING kckb,r2
           L     r2,0(R1)
           USING spab,r3
           L     r3,4(R1)
           USING bereich1,r4
           L     r4,8(R1)
              .
              .
              .
```

Dabei bedeutet

prgnam	Einsprungname des Teilprogramms, wie bei der Generierung in der PROGRAM-Anweisung angegeben.
kckb	Name der DSECT, die den KB beschreibt.
r2	Register, das den KB adressiert.
spab	Name der DSECT, die den SPAB beschreibt.
r3	Register, das den SPAB adressiert.
bereich1	Name der DSECT, die den 1. gemeinsamen Bereich beschreibt.
r4	Register, das *bereich1* adressiert.

Zu den Einträgen *r1*, *n* und *savpar* siehe auch Kapitel „Makroaufrufe“ auf Seite 13.

2.3 Aufruf aus Teilprogrammen

UTM-Funktionen aufrufen

Um von einem Teilprogramm aus UTM aufzurufen, gehen Sie in drei Schritten vor:

1. Sie versorgen den KDCS-Parameterbereich mit den notwendigen Angaben.
   Die notwendigen Angaben entnehmen Sie der Beschreibung des jeweiligen Funktionsaufrufs im Basishandbuch „Anwendungen programmieren mit KDCS für COBOL, C und C++“.
   UTM stellt den Makro KCPAA zur Verfügung. KCPAA generiert eine DSECT mit dem Aufbau des KDCS-Parameterbereichs. Die Namen der Parameter stimmen mit den Namen in dem COBOL-COPY-Element KCPAC überein. Verwendet der Funktionsaufruf einen Nachrichtenbereich (NB), muß dieser in der Regel versorgt sein (MPUT, FPUT). Der KDCS-Parameterbereich und der NB sollten in den SPAB gelegt werden.
2. Sie rufen den Makro ZCALL auf:

   ```
   ZCALL KDCS,kcpaa[,nb]
   ```

   Dabei bedeutet:

kcpaa	Name des KDCS-Parameterbereichs.
nb	Name des Nachrichtenbereichs.

3. Sie können die Rückgaben von UTM auswerten: Returncodes im Kommunikationsbereich und Daten im Nachrichtenbereich.

Beispiel

Der folgende Teilprogramm-Ausschnitt zeigt ein Beispiel für einen INIT-Aufruf:

```
INITA  EQU    *
       MVC    KCOP,INIT
       LA     R9,80
       STH    R9,KCLA
       LA     R9,138
       STH    R9,KCLM
       ZCALL  KDCS,KCPAA
       CLC    KCRCCC,=C'000'
       BNE    FEHLER
        .
```

Das Teilprogramm mit diesem INIT-Aufruf benützt einen KB-Programmbereich mit 80 Byte Länge und einen SPAB mit 138 Byte Länge. Nach dem INIT-Aufruf wird der UTM-Returncode abgefragt.

In diesem Beispiel wurden Programmierhilfen benutzt, die in Kapitel „Assembler-Datenstrukturen“ auf Seite 17 beschrieben werden.

Assembler-Unterprogramme aufrufen

Assembler-Teilprogramme, die mit PROGRAM...,COMP=ASSEMB generiert sind, können Assembler-Unterprogramme aufrufen. Dabei ist folgendes zu beachten:

– Aus Assembler-Teilprogrammen rufen Sie Unterprogramme mit dem Makro ZCALL auf.
– Den Rücksprung aus dem Unterprogramm programmieren Sie mit dem Makro ZEND.

Auch ILCS-Teilprogramme, die mit PROGRAM...,COMP=ILCS generiert wurden, können Assembler-Unterprogramme aufrufen. Dabei müssen die ILCS-Teilprogramme der ILCS-Konvention genügen. Voraussetzung hierfür ist, daß Sie folgende @-Makros des Compilers ASSEMBH verwenden (siehe auch Handbuch „ASSEMBH“):

– @ENTR mit ILCS=YES
– @PASS
– @EXIT

oder die UTM-Makros:

– ZSTRT mit ILCS=YES
– ZCALL
– ZEND

Von ILCS-Teilprogrammen können Sie jedes ILCS-fähige Unterprogramm aufrufen, also auch solche Unterprogramme, die in einer anderen von ILCS unterstützten Sprache geschrieben sind.

Weitere Informationen zum Aufruf von Unterprogrammen aus Teilprogrammen finden Sie im Basishandbuch „Anwendungen programmieren mit KDCS für COBOL, C und C++“.

2.4 Übersetzen und Binden von ILCS-fähigen Assemblerprogrammen

Beim Übersetzen eines Assembler-Programmes mit ZSTRT ILCS=YES muß als zusätzliche Makrobibliothek

$TSOS.SYSLIB.ASSEMBH.012

zugewiesen werden. In dieser Bibliothek sind die von den Z-Makros benutzten Makros @ENTR, @PASS und @EXIT enthalten.

Beim Binden der UTM-Anwendung muß die Bibliothek

$TSOS.SYSLIB.ASSEMBH.012

zugewiesen werden. Diese Bibliothek enthält das Assembler-Laufzeitsystem.

Weitere Information sind im Handbuch „ASSEMBH V1.2A BHB" enthalten.

2.5 Shareable Assembler-Module

Ein shareable (gemeinsam benutzbares) Assembler-Modul kann nur in einen Common Memory Pool im Benutzerspeicher (Klasse 6 Speicher) geladen werden. Es darf nur ein Untermodul von UTM-Teilprogrammen sein. Es darf **nicht** als UTM-Teilprogramm in der KDCDEF-Steueranweisung PROGRAM definiert sein.

In der MODULE-Anweisung geben Sie mit LOAD=(POOL, *poolname*) an, wohin das shareable Assembler-Modul geladen werden soll. Weitere Einsprungpunkte können Sie mit der KDCDEF-Steueranweisung ENTRY definieren.

Arbeiten Sie mit dem BLS (Binder, Lader, Starter), dürfen Sie nicht die MODULE-Anweisung verwenden, sondern die Generierungsanweisung LOAD-MODULE.

Assembler-Module, die in der PROGRAM-Anweisung als UTM-Teilprogramme definiert sind, können selbst nicht shareable sein.

Shareable Module dürfen keine Extern-Adressen, keine V-Konstanten, keine lokalen Registersicherstellungsbereiche und keine lokalen Arbeitsbereiche enthalten.

Die Makros ZSTRT, ZCALL, ZEND sind **nicht** zugelassen.

Die Kommunikation zwischen UTM und shareable Modulen im Common Memory Pool erfolgt ausschließlich über UTM-Teilprogramme, die in der PROGRAM-Anweisung definiert sind und die ihrerseits mit den shareable Modulen nach den üblichen Konventionen für Untermodul-Ansprung verkehren (BASR Rx,Ry).

Enthält ein UTM-Teilprogramm einen Verweis auf ein shareable Modul, so sind unterschiedliche Angaben in der PROGRAM-Anweisung erforderlich, abhängig davon, ob Sie die BLS-Schnittstelle nutzen oder nicht:

– ohne BLS-Schnittstelle:
  Sie müssen in der PROGRAM-Anweisung den Operanden LOAD=STARTUP angeben

– mit BLS-Schnittstelle:
  Das Teilprogramm muß in einem Lademodul enthalten sein, das Sie in der Anweisung LOAD-MODULE beschreiben. Über den Parameter LOAD-MODE=STARTUP bzw. LOAD-MODE=ONCALL legen Sie fest, daß das Lademodul als eigenständige Einheit nachgeladen wird.
  In der PROGRAM-Anweisung müssen Sie im Parameter LOAD-MODULE den Namen des Lademoduls angeben.

Weitere Einzelheiten zur Generierung von Modulen, die shareable sein sollen, finden Sie im *open*UTM-Handbuch „Anwendungen generieren und betreiben“.

Beispiel

1. Im folgenden Beispiel nutzt die UTM-Anwendung nicht die Funktionen des BLS.

   Das Assembler-Teilprogramm TPRGASS ruft das shareable Assembler-Modul SHARASS auf. TPRGASS wird in der PROGRAM-Anweisung und SHARASS in der MODULE-Anweisung definiert.

```
TPRGASS  ZSTRT
          .
          .
         L     R15,=V(SHARASS)
         BALR  R14,R15
          .
          .
         ZCALL KDCS,PARM1,PARM2
          .
          .
         END


SHARASS CSECT PUBLIC
         USING *,R15
          .
          .
         BR    R14
          .
          .
         END
```

2. In diesem Beispiel nutzt die UTM-Anwendung die BLS-Funktionen.

   Beide Module, TPRGASS und SHARASS, werden mit dem Binder zu einem LLM (Link and Load Module) namens LMODASS mit Private und Public Slice gebunden.

   Mit der LOAD-MODULE-Anweisung werden Namen, Versionen und Eigenschaften des Lademoduls LMODASS festgelegt:

```
LOAD-MODULE    LMODASS, VERSION=xxx, LIB= lmod-lib,
               LOAD-MODE = (POOL, poolname, STARTUP)
```

   Mit der PROGRAM-Anweisung werden Namen und Eigenschaften des Teilprogramms festgelegt, das Lademodul wird benannt:

```
PROGRAM   TPRGASS, COMP= ASSEMB
          LOAD-MODULE= LMODASS
```

3 Makroaufrufe

Dieses Kapitel beschreibt Makroaufrufe, die *open*UTM für die Programmierung von Assembler-Teilprogrammen bzw. Unterprogrammen zur Verfügung stellt.

ZSTRT - Programmbeginn und Parameterübergabe

Der Makro ZSTRT muß als erste Anweisung eines Assembler-Teilprogramms bzw. eines Unterprogramms stehen. Er führt folgendes aus:

– CSECT mit Einsprungnamen generieren
– Registerinhalte sicherstellen
– Basisadreßregister zuweisen und laden
– Sicherstellungsbereich generieren
– Adresse des Sicherstellungsbereichs in Register 13 laden, Parameteradresse sichern
– C-Konstante mit dem Namen des Teilprogramms oder Unterprogramms generieren
– Registernummern mit R0-R15 gleichsetzen

Format

Name	Operation	Operanden
name	ZSTRT	[BASIS={(r1) | Rr1}] [,REGS=n] [,PARM={savpar | (r2) | Rr2}] [,ILCS={YES | NO}]

Die Einträge bedeuten:

name Name des Teilprogramms oder Unterprogramms

BASIS=(*r1*) *r1* ist die Nummer des Basisregisters

BASIS=Rr1	Rr1 ist der Name (gebildet aus „R“ und der Registernummer) des Basisadreßregisters, das zugewiesen werden soll, z.B. (3) oder R3 - wahlfrei. Bei mehreren Basisadreßregistern ist es die Nummer des ersten Basisadreßregisters. Die Register werden fortlaufend gezählt. Die Registernummern müssen zwischen 3 und 12 liegen. Standardwert: (12) bzw. R12
REGS=*n*	Es werden *n* Basisadreßregister verwendet - wahlfrei. Standardwert: 1
PARM=*savpar*	*savpar* ist die Adresse eines Wortes, in das der Inhalt von Register 1 gesichert wird - wahlfrei.
(*r2*)	*r2* ist die Nummer eines Registers, in das Register 1 umgeladen werden soll.
Rr2	Rr2 der Name (gebildet aus „R“ und der Registernummer) eines Registers, in das Register 1 umgeladen werden soll. Standard: Register 1 wird nicht gesichert **Hinweis** Register 1 enthält die Adresse der Parameterliste und wird von ZSTRT nicht verändert.
ILCS=YES/NO	Bei YES werden zur Programmverknüpfung die ILCS Konventionen verwendet. In diesem Fall muß das Teilprogramm mit COMP=ILCS generiert werden. Standardwert: NO

Mögliche MNOTES

Modulname fehlt

Formalfehler bei Parameter

ZCALL - UTM bzw. Unterprogramm aufrufen

Der Makro ZCALL führt folgende Funktionen aus:

– V-Konstante mit der Zieladresse generieren
– Rücksprungadresse speichern
– Parameterliste generieren
– ins Unterprogramm verzweigen

Format

Name	Operation	Operanden
[name]	ZCALL	{ KDCS / upgnam / (r1) } [,{ par / /par / (r2) }] [,...]

Die Einträge bedeuten:

name symbolische Adresse des Makroaufrufs - wahlfrei.

KDCS Aufruf einer UTM-Funktion.

upgnam Aufruf eines Unterprogramms mit Namen *upgnam* (Zieladresse).

(*r1*) Nummer oder Name eines Registers mit der Zieladresse.

par Die Adresse von *par* wird in die Parameterliste eingetragen - wahlfrei.
Es sind maximal 78 Parameter möglich.
Wird hier kein Operand angegeben, bleibt bei ILCS=NO Register 1 unverändert, bei ILCS=YES wird Register 1 auf 0 gesetzt.
Andernfalls schreibt ZCALL die Adresse der Parameterliste in Register 1.

/par Die mit DSECT definierte Adresse von *par* wird eingetragen.

(*r2*) Die Adresse, die in Register *r2* steht, wird eingetragen.
Die Parameterliste muß lückenlos angegeben werden.
Standard: Es wird keine Parameterliste erzeugt und Register 1 bleibt unverändert.

Mögliche MNOTES

Ansprungname fehlt

Formfehler bei Parameter

ZEND - Unterprogramm beenden

ZEND steuert den Rücksprung in das aufrufende Programm. Dabei kann ein Returncode mit übergeben werden. ZEND führt folgende Funktionen aus:

– Register zurückladen

– Rücksprung in das aufrufende Programm

– Returncode übertragen

Format

Name	Operation	Operanden
[name]	ZEND	[RC=⎧zahl⎫] ⎩(r1)⎭

Die Einträge bedeuten:

[*name*]	symbolische Adresse des Makros - wahlfrei.
RC=	Angabe des Returncodes, der dem aufrufenden Programm im Register 15 übergeben werden soll.
zahl	1 bis 4096
(*r1*)	der Returncode wird aus Register *r1* in Register 15 übernommen. Register 13 darf dabei nicht benutzt werden.
	Standard: kein Returncode, Register 15 wird gelöscht.

Mögliche MNOTES

Operand größer 5 Stellen (Returncode größer 5 Stellen)

Numerischer Operandenteil fehlerhaft (alpha-Zeichen in Registernotation)

Returncode größer 4096

Registerangabe größer 15

Register 13 angewendet

Bei ILCS=YES wird RC= nicht unterstützt.

4 Assembler-Datenstrukturen

Folgende Datenstrukturen erleichtern Ihnen die Programmierung und helfen, daß Programmteile leicht geändert oder ausgetauscht werden können:

KCAPROA	definiert einen optionalen zweiten Parameterbereich für den APRO-Aufruf. Dieser Bereich dient zur Auswahl spezieller OSI TP-Funktionskombinationen.
KCATA	definiert die UTM-Attributfunktionen für FHS.
KCKBA	erzeugt eine DSECT mit dem Aufbau des KB-Kopf im KDCS-Kommunikationsbereich. An diesen KB-Kopf kann sich ein KB-Programmbereich anschließen, den Sie selbst definieren müssen. Die Adresse des realen KB übergibt UTM in einer Parameterliste (siehe Abschnitt „Aufbau eines Assembler-Teilprogramms“ auf Seite 5). Aufruf mit KCKBA C unterdrückt die DSECT-Anweisung.
KCPAA	erzeugt eine DSECT mit dem Aufbau des KDCS-Parameterbereichs. Der KDCS-Parameterbereich ist am besten in den SPAB zu legen. Zur Adressierung des SPAB siehe Abschnitt „Aufbau eines Assembler-Teilprogramms“ auf Seite 5. Aufruf mit KCPAA C unterdrückt die DSECT-Anweisung. Aufruf mit KCPAA PREFIX=*xx*: Hiermit wird ein anderer als der Standardpräfix KC gewählt. Diese Funktion ist z.B. notwendig, wenn für einen INFO-Aufruf mit KCOM=CK das Layout eines zweiten KDCS-Parameterbereichs benötigt wird. Beachten Sie, daß KCPAA auf Doppelwort ausgerichtet sein muß!
KCOPA	definiert Konstanten mit den Namen der KDCS-Operationen.
KCDFA	definiert die KDCS-Bildschirmausgabefunktionen.
KCINIA	definiert einen zweiten Parameterbereich für den INIT-Aufruf (nur notwendig bei INIT PU). In diesen Parameterbereich liefert UTM die mit dem INIT PU Aufruf angeforderten Informationen zurück.
KCINFA	definiert Datenstrukturen für die Rückgabe beim UTM-Aufruf INFO.
KCMSGA	definiert eine DSECT mit der Datenstruktur der UTM-Meldungen.
KCDADA	definiert Datenstrukturen für den DADM-Aufruf.

KCPADA definiert Datenstrukturen für den PADM-Aufruf.

KCINPA definiert eine DSECT für den Aufbau des Parameterbereichs für den Event-Exit INPUT.

KCCFA definiert den zweiten Parameter, den UTM beim Event-Exit INPUT übergibt. In diesem Parameter übergibt UTM die Inhalte der Steuerfelder von Bildschirmformaten an das Teilprogramm. Dieser zweite Parameter wird deshalb auch Steurfeldbereich (Control Fields) genannt.

Diese Datenstrukturen werden in der Bibliothek SYSLIB.UTM.050.ASS bereitgestellt.

Im folgenden werden die Datenstrukturen für KCKBA und KCPAA dargestellt.

Datenstruktur KCKBA

```
************************************************************************
*                                                                      ***
*      COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1992       ***
*                    ALL RIGHTS RESERVED                               ***
*      COPYRIGHT (C) SIEMENS AG 1998 ALL RIGHTS RESERVED               ***
*                                                                      ***
************************************************************************
*      SIEMENS AG openUTM  5.0                                         ***
MACRO            500       980708   51311101
&NAME    KCKBA &C,&CSECT=NO
         SPACE
************************************************************************
*                                                                      *
*        KDCS COMMUNICATION AREA     (KB)                              *
*                                                COPY:  KCKBA          *
************************************************************************
         SPACE
************************************************************************
*        KDCS KB HEADER                                                *
************************************************************************
         SPACE
         AIF   (&C EQ 'C').P1
&NAME    DSECT                          KDCS COMMUNICATION AREA
         AGO   .P2
.P1      ANOP
         AIF   (&CSECT EQ 'NO').P3
&NAME    AMODE ANY
&NAME    RMODE ANY
&NAME    CSECT
         AGO   .P2
.P3      ANOP
&NAME    DS     0D                      KDCS COMMUNICATION AREA
.P2      ANOP
         SPACE
KCKBKOPF DS    0CL68                    KB HEADER
KCBENID  DS    D                        . USER IDENTIFICATION
KCVORG   DS    0CL24                    . CONVERSATION-SPECIFIC DATA:
KCTACVG  DS    CL8                        . TRANSACTION CODE
KCDATVG  DS    0CL9                       . DATE:
KCTAGVG  DS    CL2                           . DAY
KCMONVG  DS    CL2                           . MONTH
KCJHRVG  DS    CL2                           . YEAR
KCTJHVG  DS    CL3                           . DAY OF YEAR
KCUHRVG  DS    0CL6                        . TIME:
KCSTDVG  DS    CL2                           . HOUR
```

```
KCMINVG  DS    CL2                        . MINUTE
KCSEKVG  DS    CL2                        . SECOND
KCKNZVG  DS    CL1                       . CONVERSATION ID
KCAKTUEL DS    OCL16                   . DATA TO CURRENT PROGRAM RUN:
KCTACAL  DS    CL8                       . TRANSACTION CODE
KCUHRAL  DS    OCL6                      . TIME:
KCSTDAL  DS    CL2                          . HOUR
KCMINAL  DS    CL2                          . MINUTE
KCSEKAL  DS    CL2                          . SECOND
KCAUSW   DS    CL1                     . A = CARD IN READER
KCTAIND  DS    CL1                     . TRANSACTION INDICATOR
KCLOGTER DS    CL8                     . NAME OF UTM TERMINAL (LTERM)
KCTERMN  DS    CL2                     . TERMINAL MNEMONIC
KCLKBPB  DS    H                       . MAXIMUM LENGTH OF
*                                      . KB PROGRAM AREA
KCSTA    DS    OCL3                    . STACK INFORMATION:
KCHSTA   DS    H                       . CURRENT STACK LEVEL
KCDSTA   DS    CL1                     . CHANGE IN STACK LEVEL
         DS    CL1
KCPRIND  DS    CL1                     . PROGRAM INDICATOR
KCOF1    DS    CL1                     . OSI-TP FUNCTION1
KCCP     DS    CL1                     . CLIENT PROTOCOL
KCTARB   DS    CL1                     . TRANSACTION IS MARKED ROLLBACK
KCYEARVG DS    CL4                     . YEAR START CONVERSATION
         SPACE
         DS    CL12
         SPACE
*************************************************************************
*        KDCS RETURN AREA                                               *
*************************************************************************
         SPACE
KCRFELD  DS    OCL24                 KDCS RETURN AREA
KCRI     DS    CL2                     . RETURN IDENTIFICATION
*                                      . (NOT USED)
         ORG   KCRI
KCRDF    DS    H                       . RETURN DEVICE FEATURE
KCRLM    DS    H                       . RETURN LENGTH
KCRINFCC DS    CL3                     . INFO CALL ERROR CODE
         ORG   KCRINFCC
KCRSTATE DS    OCL2                    . CONVERSATION AND
*                                      . TRANSACTION STATUS
KCRST    EQU   KCRSTATE
KCVGST   DS    CL1                     . CONVERSATION STATUS
KCTAST   DS    CL1                     . TRANSACTION STATUS
         DS    CL1                     . NOT USED
         ORG   KCRINFCC
KCRSIGN  DS    OCL3                    . STATUS OF SIGN-ON:
KCRSIGN1 DS    CL1                     . PRIMARY CODE
```

```
KCRSIGN2 DS    CL2                    . SECONDARY CODE
KCRMGT   DS    CL1                    . RETURN INFO MGET
KCRC     DS    0CL8                   . RETURN CODES:
KCRCCC   DS    CL3                      . KDCS ERROR CODE
KCRCKZ   DS    CL1                      . INDICATOR
*                                       . P=PRODUCTION, T=UTM-T
KCRCDC   DS    CL4                      . ADDITIONAL ERROR CODE FROM
*                                       . UTM (NOT COMPATIBLE)
KCRMF    DS    CL8                   .RETURN MESSAGE FORMAT
KCRPI    DS    CL8                   . RETURN CONVERSATION ID
         ORG   KCRPI
KCRUS    DS    CL8                   . RETURN USER (SIGN ST)
         SPACE
*************************************************************************
*        KDCS KB PROGRAM AREA                                           *
*************************************************************************
         SPACE
KCKBPRG  EQU   *                     KB PROGRAM AREA
         SPACE 2
         MEND
```

Datenstruktur KCPAA

```
************************************************************************
*                                                                      ***
*      COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1992       ***
*                    ALL RIGHTS RESERVED                               ***
*      COPYRIGHT (C) SIEMENS AG 1998 ALL RIGHTS RESERVED               ***
*                                                                      ***
************************************************************************
*      SIEMENS AG openUTM  5.0                                         ***
         MACRO            500       980708   51311102
&NAME    KCPAA &C,                                                      -
               &PREFIX=KC
         SPACE
************************************************************************
*                                                                      *
*        KDCS STANDARD PRIMARY WORKING AREA                            *
*        (SPAB)                                                        *
*                                                                      *
************************************************************************
         SPACE
         AIF   (&C EQ 'C').P1
&NAME    DSECT                KDCS STANDARD PRIMARY WORKING AREA (SPAB)
         SPACE
         AGO   .P2
.P1      ANOP
&NAME    DS    OF             KDCS STANDARD PRIMARY WORKING AREA (SPAB)
.P2      ANOP
&PREFIX.OP     DS    CL4            . OPERATION CODE
&PREFIX.OM     DS    CL2            . OPERATION MODIFICATION
&PREFIX.LA     DS    H              . LENGTH OF AREA
&PREFIX.LKBPRG EQU   &PREFIX.LA          . LENGTH OF KB PROGRAM AREA
&PREFIX.LM     DS    H              . LENGTH OF MESSAGE
&PREFIX.LPAB   EQU   &PREFIX.LM          . LENGTH OF SPAB
&PREFIX.RN     DS    CL8            . REFERENCE NAME
*                                   . TAC/LTERM/STORAGE AREA
&PREFIX.MF     DS    CL8            . MESSAGE FORMAT
&PREFIX.LT     EQU   &PREFIX.MF         . NAME OF UTM TERMINAL
&PREFIX.US     EQU   &PREFIX.MF         . USER ID
&PREFIX.PA     EQU   &PREFIX.MF         . NAME OF THE PARTNER
*                                       . APPLICATION
&PREFIX.DF     DS    H              . SCREEN FUNCTION
&PREFIX.LI     EQU   &PREFIX.DF         . LENGTH OF INIT AREA
&PREFIX.EXTENT DS    OCL14          . EXTENTION SINCE V3.0
*
&PREFIX.DPUT   DS    OCL10          . FOR DPUT-CALL:
&PREFIX.MOD    DS    CL1            . MODE: A=ABSOLUTE,R=RELATIVE
```

```
*                                       .       SPACE= NO TIME
&PREFIX.TAG    DS    CL3                . DAY
&PREFIX.STD    DS    CL2                . HOUR
&PREFIX.MIN    DS    CL2                . MINUTE
&PREFIX.SEK    DS    CL2                . SECOND
               DS    CL4                . NOT USED
*
         ORG   &PREFIX.EXTENT
&PREFIX.APRO   DS    OCL8               . FOR APRO-CALL:
&PREFIX.PI     DS    CL8                . CONVERSATION ID
&PREFIX.OF     DS    CL1                . OSI-TP FUNCTIONS
               DS    CL5                . NOT USED
*
         ORG   &PREFIX.EXTENT
&PREFIX.PADM   DS    OCL11              . FOR PADM-CALL:
&PREFIX.ACT    DS    CL3                . ACTION (ON/OFF/CON/DIS)
&PREFIX.ADRLT  DS    CL8                . ADDRESSED LTERM, DESTINATION
               DS    CL3                . (NOT USED)
*
         ORG   &PREFIX.MF
&PREFIX.MCOM   DS    OCL24              . FOR MCOM-CALL:
&PREFIX.POS    DS    CL8                . DESTINATION IN POSITIVE CASE
&PREFIX.NEG    DS    CL8                . DESTINATION IN NEGATIVE CASE
&PREFIX.COMID  DS    CL8                . COMPLEX IDENTIFICATION
*
         ORG   &PREFIX.EXTENT
&PREFIX.SGCL   DS    OCL12              . FOR SIGN CL CALL:
&PREFIX.LANGID DS    CL2                . LANGUAGE ID
&PREFIX.TERRID DS    CL2                . TERRITORY ID
&PREFIX.CSNAME DS    CL8                . CODED CHARACTER SET NAME
               DS    CL2                . (NOT USED)
*
         ORG
         SPACE
&PREFIX.PAREND EQU   *                  . END OF PARAMETER AREA
         SPACE 2
         MEND
```

Stichwörter

Inhalt

*open*UTM V5.0 (BS2000/OSD)

Anwendungen programmieren mit KDCS für Assembler

Zielgruppe
Dieses Handbuch soll Assembler-Programmierer von *open*UTM-Anwendungen in ihrer Arbeit unterstützen.

Inhalt

Dieses Handbuch beschreibt die sprachspezifischen Besonderheiten bei der Erstellung von Assembler-Teilprogrammen. Es ergänzt das Basishandbuch „Anwendungen programmieren mit KDCS für COBOL, C und C++“.

Ausgabe: November 1998

Datei: utm_ass.pdf

Information on this document

On April 1, 2009, Fujitsu became the sole owner of Fujitsu Siemens Computers. This new subsidiary of Fujitsu has been renamed Fujitsu Technology Solutions.

This document from the document archive refers to a product version which was released a considerable time ago or which is no longer marketed.

Please note that all company references and copyrights in this document have been legally transferred to Fujitsu Technology Solutions.

Contact and support addresses will now be offered by Fujitsu Technology Solutions and have the format *...@ts.fujitsu.com*.

The Internet pages of Fujitsu Technology Solutions are available at *http://ts.fujitsu.com/*...
and the user documentation at *http://manuals.ts.fujitsu.com*.

Hinweise zum vorliegenden Dokument

Zum 1. April 2009 ist Fujitsu Siemens Computers in den alleinigen Besitz von Fujitsu übergegangen. Diese neue Tochtergesellschaft von Fujitsu trägt seitdem den Namen Fujitsu Technology Solutions.

Das vorliegende Dokument aus dem Dokumentenarchiv bezieht sich auf eine bereits vor längerer Zeit freigegebene oder nicht mehr im Vertrieb befindliche Produktversion.

Bitte beachten Sie, dass alle Firmenbezüge und Copyrights im vorliegenden Dokument rechtlich auf Fujitsu Technology Solutions übergegangen sind.

Kontakt- und Supportadressen werden nun von Fujitsu Technology Solutions angeboten und haben die Form *...@ts.fujitsu.com*.

Die Internetseiten von Fujitsu Technology Solutions finden Sie unter *http://de.ts.fujitsu.com/*..., und unter *http://manuals.ts.fujitsu.com* finden Sie die Benutzerdokumentation.