

# 1 Einleitung

AID V2.0B enthält die ESA<sup>TM</sup>-Unterstützung (Enterprise Systems Architecture) analog zu der von BS2000 V11. Außerdem kann mit AID V2.0B der schreibende Zugriff auf eine Speicherstelle überwacht werden. AID V2.0B können Sie ab BS2000 V9.5 einsetzen, die ESA-Unterstützung und die Schreibüberwachung können Sie aber erst ab BS2000 V11 nutzen.

## 1.1 Konzept des Handbuchs

Diese Ergänzung enthält die zu AID V2.0B entwickelten Änderungen. Zusammen mit den Handbüchern zu AID V2.0A enthält sie alle Informationen zum Testen mit AID V2.0B. Zu Beginn jedes Abschnitts werden die Handbücher und die Abschnitte genannt, die von den jeweiligen Änderungen betroffen sind.

Die Dokumentation von AID V2.0A besteht aus einem Basishandbuch und den sprachspezifischen Handbüchern für das symbolische Testen sowie dem Handbuch für das Testen auf Maschinencode-Ebene.

### **AID - Basishandbuch**

Im Basishandbuch finden Sie einen Überblick über AID und die Beschreibung der Sachverhalte und Operanden, die in allen Programmiersprachen gleich sind. Im Überblick wird die BS2000-Umgebung beschrieben, es werden die grundlegenden Begriffe erläutert und der AID-Kommandovorrat vorgestellt. Die anderen Kapitel beschreiben die Testvorbereitung, die Kommandoeingabe, die Operanden Subkommando, komplexe Speicherreferenz und Medium-und-Menge, die AID-Literale und die Schlüsselwörter. Das Handbuch enthält außerdem die Meldungen, die in Kommandofolgen unzulässigen BS2000-Kommandos, die nur noch in dieser Version unterstützten Operanden und eine Gegenüberstellung von AID und IDA.

### **AID - Testen auf Maschinencode-Ebene**

### **AID - Testen von COBOL-Programmen**

### **AID - Testen von FORTRAN-Programmen**

### **AID - Testen von PL/I-Programmen**

### **AID - Testen von ASSEMBH-Programmen**

### **AID - Testen von C-Programmen**

In den sprachspezifischen Handbüchern und dem Handbuch für das Testen auf Maschi-

nencode-Ebene finden Sie die Kommandos in alphabetischer Reihenfolge. Alle einfachen Speicherreferenzen sind in diesen Handbüchern beschrieben.

In den sprachspezifischen Handbüchern ist die Beschreibung der Operanden auf die jeweilige Programmiersprache zugeschnitten. Es wird vorausgesetzt, daß Ihnen der Sprachumfang und die Handhabung des jeweiligen Compilers geläufig sind.

Das Handbuch für das Testen auf Maschinencode-Ebene können Sie für Programme einsetzen, zu denen keine LSD-Sätze vorhanden sind oder für die die Informationen aus dem symbolischen Testen zur Diagnose nicht ausreichen. Beim Testen auf Maschinencode-Ebene sind Sie bei der Anwendung der AID-Kommandos unabhängig von der Programmiersprache, in der das Programm erstellt wurde.

## 1.2 Änderungen gegenüber AID V2.0A

Mit %DISPLAY %LINK wird der Name des Moduls bzw. Segments ausgegeben, das beim letzten erfolgreichen %LPOV-Ereignis (AID-Kommando %ON) nachgeladen wurde.

Mit %ON kann der schreibende Zugriff auf eine Speicherstelle überwacht werden. Hierzu wurde als neues Ereignis %WRITE(speicherref) eingeführt.

Zur ESA-Unterstützung wurden folgende Erweiterungen eingeführt:

Die ALET/SPID-Qualifikation ermöglicht den Zugriff auf virtuelle Adressen in einem Datenraum.

Über die Schlüsselwörter %0AR bis %15AR können Sie die Zugriffsregister lesen und verändern.

Das Schlüsselwort %DS[(ALET/SPID-qua)] liefert Informationen über Datenräume und über die zugehörigen SPIDs und ALETs.

Das Schlüsselwort %ASC informiert über den ASC-Modus, der bestimmt, ob der AR-Modus gesetzt ist und Datenräume adressiert werden oder nicht.

Das %TRACE-Protokoll enthält zusätzlich zum bisherigen Umfang nun auch den Inhalt der Zugriffsregister, die neuen ESA-Maschinenbefehle und einen Stern (\*) vor Adressen aus Datenräumen.

Adressen aus Datenräumen sind bei der Ausgabe mit %DISPLAY mit einem Stern (\*) gekennzeichnet.

## 1.3 Metasyntax

Für die Darstellung der Kommandos wird folgende Metasyntax verwendet. Die Aufstellung zeigt die verwendeten Symbole und beschreibt ihre Bedeutung.

### GROSSBUCHSTABEN

Zeichenfolge, die Sie unverändert übernehmen müssen, wenn Sie eine Funktion auswählen.

### kleinbuchstaben

Zeichenfolge, die eine Variable bezeichnet. An ihre Stelle müssen Sie einen der zugelassenen Operandenwerte setzen.

$$\left\{ \begin{array}{c} \text{alternativ} \\ \dots \\ \text{alternativ} \end{array} \right\}$$

{alternativ | ... | alternativ}

Alternativen, unter denen Sie eine auswählen müssen. Die beiden Formate sind gleichbedeutend.

[wahlweise]

Die in eckige Klammern eingeschlossenen Angaben können entfallen.

Bei AID-Kommandonamen kann der in eckigen Klammern stehende Teil nur komplett entfallen, andere Abkürzungen ergeben einen Syntaxfehler.

[...]

Wiederholbarkeit einer wahlfreien syntaktischen Einheit. Muß vor jede Wiederholung ein Trennzeichen, z.B. Komma gesetzt werden, steht es vor den Wiederholungspunkten.

{...}

Wiederholbarkeit einer syntaktischen Einheit, die einmal angegeben werden muß. Muß vor jede Wiederholung ein Trennzeichen, z.B. Komma gesetzt werden, steht es vor den Wiederholungspunkten.

### Unterstreichung

Die Unterstreichung kennzeichnet den Standardwert, den AID einsetzt, wenn Sie für einen Operanden keinen Wert angeben.

- Der dickere Punkt trennt Qualifikationen, steht für eine *vorqualifikation* (siehe %QUALIFY), ist der Operator für einen Adreßversatz oder Teil des Durchlaufzählers bzw. Subkommando-Namens. Eingegeben wird der dickere Punkt mit dem Punkt, der auf der Tastatur ist. Er wurde nur der besseren Lesbarkeit wegen dicker dargestellt.

Im Fließtext werden Operanden in *kursiven kleinbuchstaben* dargestellt.

## 2 %WRITE(speicherref), die Schreibüberwachung beim %ON

Erweiterung des %ON-Kommandos in allen sprachspezifischen Handbüchern und im Handbuch für das Testen auf Maschinencode-Ebene. Erweiterung der Ereignis-Tabelle im AID-Basishandbuch, Abschnitt 9.13, Ereignis bei %ON.

Die Schreibüberwachung des %ON ist über das neue Ereignis %WRITE(speicherref) realisiert. Dieses Ereignis bewirkt, daß AID immer dann den Programmablauf unterbricht und das zugehörige Subkommando bearbeitet, wenn das Programm schreibend auf ein Byte innerhalb des festgelegten Speicherbereichs zugreift. Die Schreibüberwachung kann nur verwendet werden, wenn das zu testende Programm in einer BS2000-Version  $\geq$  V11 abläuft.

### 2.1 Funktion der Schreibüberwachung

Im folgenden wird das neue Ereignis zur Schreibüberwachung im Kontext der %ON-Beschreibung, wie sie in den sprachspezifischen Handbüchern existiert, erläutert.

Kommando	Operand
%ON	$\left\{ \begin{array}{l} \text{write-ereignis} \\ \text{ereignis} \end{array} \right\} \quad [ <\text{subkdo}> ]$

Für *write-ereignis* gelten alle Regeln des %ON mit folgenden Zusätzen:

- Es kann immer nur ein *write-ereignis* angemeldet sein. Die Eingabe von einem neuen *write-ereignis* überschreibt ein bereits bestehendes. Mit anderen Ereignissen jedoch kann *write-ereignis* gleichzeitig angemeldet sein. Wenn sie auch gleichzeitig zutreffen, wird das Subkommando von *write-ereignis* zuerst bearbeitet.
- *write-ereignis* löschen Sie mit %REMOVE %WRITE ohne Angabe von (*speicherref*) oder indem Sie mit %REMOVE %ON alle aktiven Ereignisse löschen.

write-ereignis

Mit dem Schlüsselwort %WRITE wird die Schreibüberwachung eingeschaltet. Dahinter wird in Klammern der zu überwachende Speicherbereich festgelegt. Wenn das Programm schreibend auf ein Byte innerhalb des festgelegten Bereichs zugreift, wird der Programmablauf unterbrochen und *subkdo* ausgeführt. Die Unterbrechungsstelle liegt hinter dem Befehl, der schreibend zugegriffen hat.

Der zu überwachende Speicherbereich kann jedes, wie auch immer angesprochene Speicherobjekt sein, sofern es innerhalb des vom Programm belegten Speicherbereichs liegt. Register z.B. können nicht überwacht werden. Es wird der Speicherbereich überwacht, der sich zum Zeitpunkt der Eingabe aus der Adresse und Länge der angegebenen Speicherreferenz ergibt. Die Adreß- und Längenattribute von Speicherreferenzen sind im AID-Basishandbuch, Kapitel 6, Operand komplexe Speicherreferenz, beschrieben.

- Wenn eine virtuelle Adresse angegeben wird, sind das 4 Bytes ab dieser Adresse.
- Wenn ein Datenname angegeben wird, ist das der Speicherbereich, der sich aus dem zugehörigen Adreß- und Längenattribut ergibt. Wenn ein Datenobjekt im Verlauf des Programmes auf einen anderen Speicherplatz zu liegen kommt, muß %ON *write-ereignis* erneut eingegeben werden.

Die resultierende Länge, d.h. der zu überwachende Bereich, darf 1024 Byte nicht überschreiten, sonst wird eine Fehlermeldung ausgegeben. Wenn bei einem Programm mit Überlagerungsstruktur die Adresse des angegebenen Speicherobjekts überladen wird, wird anschließend der entsprechende Bereich des neu geladenen Programmteils überwacht.

Beim Testen von ESA-Programmen können nur Speicherobjekte im Programmraum überwacht werden.

write-ereignis-OPERAND - - - - -

$$\%WRITE\left(\left\{\begin{array}{l} [\bullet][qua\bullet] \\ \left\{\begin{array}{l} V'f\dots f' \\ C=csect \\ datenname \\ anweisungsname-\> \\ S'\dots'-\> \end{array}\right\} \end{array}\right\}\left[\begin{array}{l} \left\{\begin{array}{l} \{ganzzahl\} \\ \bullet \\ \{(ausdruck)\} \end{array}\right\} \\ [T/L-mod] \end{array}\right]\left[\dots\right] [T/L-mod]\left[\dots\right] \right) \\ \left\{\begin{array}{l} \%@(speicherref)-\> \\ [T/L-mod] -\> \end{array}\right\}$$

- - - - -

Wie Sie eine Speicherreferenz angeben, ist im AID-Basishandbuch, Kapitel 6, beschrieben. Die Speicherreferenz in einem *write-ereignis* darf nicht mit einer Adreßkonstanten (Anweisungsname, Source-Referenz) enden. Sie muß immer ein Speicherobjekt bezeichnen.

qua	Qualifikationen
C=csect	C-Qualifikation
V'f...f'	virtuelle Adresse
datenname	Datennamen
anweisungsname	Anweisungsnamen
S'...'	Source-Referenzen
%@(speicherref)	Adreßselektor
T/L-mod	Typ- und/oder Längenmodifikation
• {...}	Adreßversatz (Byte-Offset)
->	indirekte Adressierung (Pointer-Operator)
(ausdruck)	Arithmetischer Ausdruck

## 2.2 Wechselwirkungen mit anderen AID-Kommandos

Folgende Wechselwirkungen zwischen %ON *write-ereignis* und anderen AID-Kommandos bestehen:

- Wenn ein %CONTROLn oder ein %TRACE mit maschinennahem *kriterium* angemeldet ist, wird die Eingabe von %ON *write-ereignis* mit einer Fehlermeldung abgewiesen und umgekehrt.
- Den schreibenden Zugriff eines Maschinenbefehls, auf dessen Adresse ein mit %INSERT gesetzter Testpunkt liegt, kann AID nicht erkennen.
- Wenn ein %CONTROLn oder ein %TRACE mit symbolischem *kriterium* angemeldet ist, kann es sein, daß nicht alle schreibenden Zugriffe erkannt werden: AID markiert für einen symbolischen %CONTROLn oder %TRACE intern einige Maschinenbefehle mit X'0A81'. Wenn auf einem schreibenden Befehl eine solche Markierung liegt, kann AID den schreibenden Zugriff dieses Befehls nicht erkennen.

Für eine lückenlose Schreibüberwachung empfiehlt es sich, alle %CONTROLn- und %INSERT-Kommandos mit %REMOVE zu löschen. Ein eventuell noch eingetragener %TRACE wird gelöscht, wenn Sie %RESUME oder %T 1 %INSTR eingeben.

## 2.3 Beispiele

1.

```
%ON %WRITE(V'63C')
%R
STOPPED AT V'16' = ASSTEST + #'16' , EVENT: WRITE

%R
STOPPED AT V'16' = ASSTEST + #'16' , EVENT: WRITE

%R
STOPPED AT V'16' = ASSTEST + #'16' , EVENT: WRITE

%R
STOPPED AT V'16' = ASSTEST + #'16' , EVENT: WRITE

%R
STOPPED AT V'16' = ASSTEST + #'16' , EVENT: WRITE

%R
```

Jedesmal, wenn eines der 4 Bytes von V'63C' bis V'63F' verändert wurde, wird das Programm angehalten und eine STOP-Meldung ausgegeben. Ein Befehl, der vor der Adresse V'16' steht, veränderte fünfmal den Speicherinhalt zwischen V'63C' und V'63F'.

2.

```
%ON %WRITE(VAR1) <Schreib: %D 'Schreibender Zugriff auf Var1'>
%ON %TERM
%R

SCHREIBENDER ZUGRIFF AUF VAR1
SCHREIBENDER ZUGRIFF AUF VAR1
SCHREIBENDER ZUGRIFF AUF VAR1
SCHREIBENDER ZUGRIFF AUF VAR1
SCHREIBENDER ZUGRIFF AUF VAR1
STOPPED AT V'3C' = ASSTEST + #'3C' , EVENT: TERM (NORMAL,PROGRAM,NODUMP)
```

Fünfmal wurde vom Programm auf VAR1 schreibend zugegriffen. Nach dem Ereignis %TERM wird das Programm angehalten und eine STOP-Meldung ausgegeben.



3.

```

%ON %WRITE(VAR1) <SCHREIB: >
%ON %TERM <%DISPLAY %.SCHREIB;%STOP>
%R

STOPPED AT V'3C' = ASSTEST + #'3C', EVENT: TERM (NORMAL,PROGRAM,NODUMP)
** ITN: #'0001003A' *** TSN: 0B1V *****
CURRENT PC: 0000003C CSECT: ASSTEST *****
%.SCHREIB = 5

```

Diesmal wird nur der Durchlaufzähler %.SCHREIB als Subkommando zum Ereignis %WRITE angelegt. Er wird bei jedem Durchlauf um eins erhöht. Bei Programmende (Ereignis %TERM) wird das Programm angehalten, eine STOP-Meldung und der Durchlaufzähler ausgegeben. Der Durchlaufzähler enthält den Wert 5, d.h. VAR1 wurde fünfmal verändert.

4.

```

%DISPLAY INHALT
INHALT = 000001

%DISPLAY INHALT%XL2
V'004007D4' = INHALT + #'000007D4'
004007D4 (000007D4) F0F0 00

%ON %WRITE(INHALT%L2) <(INHALT%XL2 EQ X'FFFF'):%STOP>
%R

STOPPED AT LABEL: SPRINGEN, SRC_REF: 235, SOURCE: A4, PROC: A4, EVENT: WRITE

%DISPLAY INHALT%XL2
V'004007D4' = INHALT + #'000007D4'
004007D4 (000007D4) FFFF ~~

```

AID überwacht die ersten zwei Bytes der Variablen INHALT. Wenn die beiden Bytes X'FFFF' enthalten, wird das Programm angehalten und eine STOP-Meldung ausgegeben.

5.

```
%DISPLAY V'400578'
CURRENT PC: 00400000 CSECT: A4 *****
V'00400578' = A4 + #'00000578'
00400578 (00000578) F1F2F3F4 1234

%ON %WRITE(V'400578') <%DA 4 FROM %PC->.(-6); %DA 4 FROM %PC->.(-4);%STOP>
%R

A4+CC MVC 576(4,R3),57A(R3) D2 03 3576 357A
A4+D2 LA R6,57E(R0,R3) 41 60 357E
A4+D6 L R7,88A(R0,R3) 58 70 388A
A4+DA LA R8,6AA(R0,R3) 41 80 36AA

A4+CE LRER R7,R6 35 76
A4+D0 LRER R7,R10 35 7A
A4+D2 LA R6,57E(R0,R3) 41 60 357E
A4+D6 L R7,88A(R0,R3) 58 70 388A

STOPPED AT V'4000D2' = A4 + #'D2', EVENT: WRITE

%DISPLAY V'400578'
CURRENT PC: 004000D2 CSECT: A4 *****
V'00400578' = A4 + #'00000578'
00400578 (00000578) C5D5C4C5 ENDE

%R

A4+E6 MVCL R6,R8 0E 68
A4+E8 MVI 576(R3),X'F1' 92 F1 3576
A4+EC MVN 7C2(4,R3),7C6(R3) D1 03 37C2 37C6
A4+F2 MVO 7CA(4,R3),7CE(4,R3) F1 33 37CA 37CE

A4+E8 MVI 576(R3),X'F1' 92 F1 3576
A4+EC MVN 7C2(4,R3),7C6(R3) D1 03 37C2 37C6
A4+F2 MVO 7CA(4,R3),7CE(4,R3) F1 33 37CA 37CE
A4+F8 MVZ 7C6(4,R3),7C2(R3) D3 03 37C6 37C2

STOPPED AT V'4000EC' = A4 + #'EC', EVENT: WRITE

%DISPLAY V'400578'
CURRENT PC: 004000EC CSECT: A4 *****
V'00400578' = A4 + #'00000578'
00400578 (00000578) F1D5C4C5 1NDE
```

Jedesmal, wenn im Programm eines der vier Bytes ab Adresse V'400578' verändert wurde, wird das Programm angehalten, und es werden zweimal 4 disassemblierte Befehle ausgegeben. Einmal ab der Adresse, die 6 Bytes vor der Unterbrechungsstelle liegt und einmal ab der Adresse, die 4 Bytes vor der Unterbrechungsstelle liegt (für 6-byte-lange und 4-byte-lange Schreibbefehle). Die Unterbrechungsstelle selbst liegt hinter dem Befehl, der den schreibenden Zugriff ausführte. Danach wird eine STOP-Meldung ausgegeben und mit %DISPLAY die veränderte Stelle angeschaut.

Den ersten Schreibzugriff führte der Befehl MVC (6-byte-lang) aus, den zweiten Schreibzugriff der Befehl MVI (4-byte-lang).

### 3 ESA-Unterstützung

Auf ESA-Anlagen (Enterprise System Architecture) werden neben dem Programmraum (program space), der dem bisherigen Adreßraum entspricht, weitere Adreßräume für Daten zur Verfügung gestellt. Diese Datenräume (data spaces) haben, wie der Programmraum, virtuelle Adressen von V'0' bis V'7FFFFFFF'. Sie können nur Daten enthalten, Programmcode kann in einem Datenraum nicht ausgeführt werden. Ihre eindeutige Ansprache ist über die SPID (space identification) oder über einen bzw. mehrere ALETs (access list entry token) möglich. Die SPID wird beim Anlegen eines Datenraumes vergeben und ist systemweit eindeutig und bekannt. ALETs verweisen nur innerhalb des Programms eindeutig auf einen Datenraum. Für die Adressierung mit ALETs wurden die Zugriffsregister als weiterer Registersatz parallel zu den Mehrzweckregistern eingeführt. In den Zugriffsregistern sind die ALETs enthalten. Wenn der AR-Modus (access register mode) eingeschaltet ist, werden bei der Adreßumsetzung in einem Maschinenbefehl die Zugriffsregister mit ausgewertet und so Daten in einem Datenraum adressiert.

Nur Programme, die auf ESA-Anlagen und in einer BS2000-Version  $\geq$  V11 laufen und die ESA-Befehle einsetzen, können Daten in einem solchen Datenraum ablegen. Siehe Handbuch "Makroaufrufe an den Ablaufteil".

Die ESA-Unterstützung von AID V2.0B betrifft nur das Testen auf Maschinencode-Ebene. Für die symbolische Ebene bietet AID den Wechsel auf Maschinencode-Ebene an: Daten werden über ihre virtuellen Adressen angesprochen und mit der Typmodifikation symbolisch aufbereitet.

### 3.1 Adressierungsmodus

Erweiterung im AID-Basishandbuch, Abschnitt 9.6, Systeminformationen.  
Erweiterung im Handbuch für das Testen auf Maschinencode-Ebene und in den sprach-spezifischen Handbüchern, AID-Kommando %DISPLAY.

Auf ESA-Anlagen können Sie zum einen mit 24-Bit-Adressen oder 31-Bit-Adressen arbeiten und zum anderen mit Datenräumen oder nur im Programmraum. Deshalb gibt es auf ESA-Anlagen einen weiteren Adressierungsmodus, den AR-Modus. Unabhängig vom AR-Modus können Sie die XS-Fähigkeit von ESA-Anlagen nutzen. Der Programmraum und jeder Datenraum, der angelegt wurde, kann entweder nur den unteren Adreßraum, oder den unteren und oberen Adreßraum nutzen. Die Unterstützung von XS-Programmen ist im AID-Basishandbuch, Abschnitt 2.1.6 beschrieben.

#### AR-Modus

Mit ESA-Programmen können Sie Datenräume parallel zum Programmraum nutzen. Der AR-Modus (access register mode) ist ein Teil des ASC-Modus (access space control mode) und entscheidet über die Auswertung der Zugriffsregister bei der Adreßumsetzung:

- Wenn der AR-Modus eingeschaltet ist, werden die Zugriffsregister bei der Adressierung mit ausgewertet und damit können Adressen in Datenräumen angesprochen werden.
- Wenn der AR-Modus ausgeschaltet ist, werden die Zugriffsregister nicht ausgewertet und nur Adressen im Programmraum können angesprochen werden. Das Programm arbeitet wie bisherige Programme auf Nicht-ESA-Anlagen.

Information über den AR-Modus erhalten Sie durch die Abfrage des ASC-Modus mit %DISPLAY:

```
%ASC      Information über den ASC-Modus auf ESA-Anlagen
```

Es wird ein sedezimaler Wert ausgegeben:

```
X'01'      der AR-Modus ist eingeschaltet  
X'00'      der AR-Modus ist ausgeschaltet
```

## Beispiel

```
%DISPLAY %ASC
%ASC                = 01

%DF D3=ESA.AR.DUMP
%BASE E=D3
%DISPLAY %ASC
** D3: ESA.AR.DUMP *****
%ASC                = 00
```

Das geladene Programm (E=VM) arbeitet im AR-Modus, d.h. über die Zugriffsregister wird in Datenräumen adressiert.

In der Dump-Datei ESA.AR.DUMP hingegen wurde zum Zeitpunkt der Dump-Erstellung ohne AR-Modus, also nur im Programmraum, gearbeitet.

## 3.2 %nAR, die Zugriffsregister

Erweiterung im AID-Basishandbuch, Abschnitt 9.3, Programmregister.

Erweiterung im Handbuch für das Testen auf Maschinencode-Ebene und in den sprach-spezifischen Handbüchern, AID-Kommandos %DISPLAY, %MOVE, %SET.

Auf ESA-Anlagen existiert parallel zu den Mehrzweckregistern ein weiterer Satz von Registern, die Zugriffsregister genannt werden. Über die Zugriffsregister kann das Programm auf Datenräume zugreifen. Beim Testen mit AID können Sie diese Register mit Schlüsselwörtern ansprechen:

%nAR      Zugriffsregister  $0 \leq n \leq 15$ , Speichertyp %FL4,  
            Ausgabe als Sedezimalzahl  
%AR        alle 16 Zugriffsregister in Tabellenform aufbereitet

In den Zugriffsregistern steht ein Zeiger auf einen Datenraum. Dieser Zeiger wird ALET (access list entry token) genannt. Wenn der AR-Modus eingeschaltet ist, wird bei der Adreßumsetzung das zum Basisregister korrespondierende Zugriffsregister mit ausgewertet. Ist in diesem Zugriffsregister ein Wert  $\neq 0$ , so enthält dieses Register einen ALET, und es wird eine Adresse in dem damit angesprochenen Datenraum errechnet.

Die Zugriffsregister können Sie mit %DISPLAY lesen und mit %MOVE oder %SET verändern. In der ALET/SPID-Qualifikation können sie angegeben werden.

### Beispiel

```
/%DISPLAY %0AR, %1AR, %2AR
%0AR      = 00000084
%1AR      = 00000088
%2AR      = 0000008C

/%SET 0 INTO %0AR
/%SET 255 INTO %1AR
/%SET #'C2C2C2C2' INTO %2AR

/%DISPLAY %AR
** ITN: #'0001003B' *** TSN: 0APB *****
CURRENT PC: 0000003A    CSECT: ACCESS *****
%AR    ( 0: 15)
( 0) 00000000 ( 1) 000000FF ( 2) C2C2C2C2 ( 3) 00000090 ( 4) 00000094
( 5) 000000B4 ( 6) 92020711 ( 7) 09252742 ( 8) 00000000 ( 9) 00000000
(10) 000000AC (11) 000000B0 (12) 000000B4 (13) 92020711 (14) 09252742
(15) 00000000
```

### 3.3 ALET/SPID-Qualifikation

Erweiterung im AID-Basishandbuch, Kapitel 6, Operand komplexe Speicherreferenz, und im Handbuch für das Testen auf Maschinencode-Ebene, Kapitel 3, Maschinencode-spezifische Adressierung, und Kapitel 5, AID-Kommandos %DISPLAY, %FIND, %MOVE und %SET.

Um virtuelle Adressen in einem Datenraum ansprechen zu können, benötigen Sie beim Testen mit AID eine Qualifikation, die die Adressierung mit Zugriffsregistern nachbildet. Hierzu wurde die ALET/SPID-Qualifikation eingeführt. Über die SPID eines Datenraumes oder über einen der zugehörigen ALETs kann AID die angegebene Adresse eindeutig einem Datenraum zuordnen. Die SPID wird vom Marko DSPSRV beim Anlegen eines Datenraumes zurückgeliefert. Der ALET wird vom Makro ALESRV zurückgeliefert. Siehe Handbuch "Makroaufrufe an den Ablaufteil".

Da in Datenräumen nur Daten liegen können, ist die ALET/SPID-Qualifikation nur in den Kommandos %DISPLAY, %FIND, %MOVE und %SET verwendbar.

Für die Kommandos %DISASSEMBLE, %INSERT, %CONTROLn und %TRACE ist sie ohne Bedeutung, da in diesen Kommandos Speicherreferenzen angegeben werden müssen, die im ausführbaren Teil des Programms liegen, und dieser Teil befindet sich immer im Programmraum.

Die ALET/SPID-Qualifikation ist nur verwendbar vor einer virtuellen Adresse oder vor einer komplexen Speicherreferenz, in der keine symbolischen Komponenten vorkommen.

Für das symbolische Testen, also vor einem Datennamen ist die ALET/SPID-Qualifikation nicht möglich und wird mit einer Fehlermeldung abgewiesen. Zur symbolischen Aufbereitung von Speicherinhalten aus Datenräumen bietet AID jedoch die Typmodifikation an.

Die Beschreibung der neuen Qualifikation ist so aufgebaut, daß sie den Abschnitt Qualifikationen in Kapitel 3 des Handbuchs für das Testen auf Maschinencode-Ebene ergänzt. Der vorangestellte Überblick soll die Einordnung der neuen Qualifikation erleichtern.

Qualifikationen müssen Sie immer in der Reihenfolge angeben, in der sie hier beschrieben sind. Sie werden durch Punkte getrennt. Zwischen der letzten Qualifikation und der anschließenden Adresse muß ebenfalls ein Punkt stehen.

$$[E=\left\{\begin{matrix} VM \\ Dn \end{matrix}\right\} \cdot] \left\{ \left[ \left\{ \begin{matrix} ALET=\left\{\begin{matrix} X'f\dots f' \\ \%nAR \\ \%nG \end{matrix}\right\} \\ SPID=X'f\dots f' \end{matrix}\right\} \cdot \right] \left\{ \begin{matrix} V'f\dots f' \\ kompl-speicherref \end{matrix}\right\} \right\} \left\{ \begin{matrix} schlüsselwort \\ [L=lademodul \cdot] [O=objektmodul \cdot] C=csect \end{matrix}\right\} \right\}$$

## Qualifikationen

$E=\{VM \mid Dn\}$

Die Basis-Qualifikation ist im Handbuch für das Testen auf Maschinencode-Ebene beschrieben; nicht ESA-relevant.

$$ALET=\left\{\begin{matrix} X'f\dots f' \\ \%nAR \\ \%nG \end{matrix}\right\}$$

bezeichnet einen Datenraum über einen seiner ALETs. Der ALET wird vom Makro ALESRV zurückgeliefert. Er kann als Sedezimal-Literal angegeben werden oder einem Zugriffsregister bzw. einem AID-Register entnommen werden.

$X'f\dots f'$  ist ein 8-stelliges Sedezimal-Literal

$\%nAR$  ist ein Zugriffsregister;  $0 \leq n \leq 15$

$\%nG$  ist ein AID-Register, in das der ALET zwischengespeichert wurde;  
 $0 \leq n \leq 15$

$SPID=X'f\dots f'$

bezeichnet einen Datenraum über seine SPID, ein 8-byte-langes Kennzeichen. Die SPID wird vom Makro DSPSRV zurückgeliefert. Sie kann nur als Sedezimal-Literal angegeben werden.

$X'f\dots f'$  ist ein 16-stelliges Sedezimal-Literal.

$[L=lademodul \cdot] [O=objektmodul \cdot] C=csect$

ist im Handbuch für das Testen auf Maschinencode-Ebene beschrieben; nicht ESA-relevant.



## Speicherreferenzen

V'f...f'

bezeichnet eine virtuelle Adresse.

f...f ist eine maximal 8stellige Sedezimal-Adresse.

Auf ESA-Anlagen liegt eine virtuelle Adresse ohne ALET/SPID-Qualifikation im Programmraum. Mit einer ALET/SPID-Qualifikation wird eine virtuelle Adresse in einem Datenraum angesprochen.

schlüsselwort

ist im Handbuch für das Testen auf Maschinencode-Ebene beschrieben; nicht ESA-relevant.

kompl-speicherref

Eine komplexe Speicherreferenz bezeichnet eine errechnete Adresse. Welche Operationen darin vorkommen können, hängt davon ab, ob Sie mit *kompl-speicherref* ein Speicherobjekt im Programmraum oder in einem Datenraum (mit ALET/SPID-Qualifikation) ansprechen:

- für Speicherobjekte im Programmraum können Sie symbolische und maschinen-nahe Adressierung mischen; die Möglichkeiten entsprechen denen auf Nicht-ESA-Anlagen (siehe AID-Basishandbuch, Kapitel 6)
- für Speicherobjekte im Datenraum können Sie die symbolische Adressierung nicht verwenden; Sie können nur maschinennahe Komponenten verwenden; die ALET/SPID-Qualifikation wirkt sich auf alle Komponenten der komplexen Speicherreferenz aus; es können darin vorkommen:
  - Adreßversatz (•)
  - indirekte Adressierung (->)
  - Typmodifikation (%X, %F, %A, %S, %SX)
  - Längenmodifikation (%L=(ausdruck), %Ln)

Der Typselektor %T(...) kann nur in einer abschließenden Modifikation zur symbolischen Aufbereitung des Speicherobjekts verwendet werden. Ohne explizite Bereichsqualifikation nimmt AID einen hierzu verwendeten Datennamen aus dem aktuellen AID-Arbeitsbereich im Programmraum. Die ALET/SPID-Qualifikation wirkt sich also nicht auf den Datennamen in einer abschließenden Typmodifikation aus.

## Beispiele

1.

```
/%DISPLAY V'34' %T(C#DS11)
SRC_REF: 288 SOURCE: DS2S4A PROC: DS2S4A *****
C#DS11 = |PS1-|

/%DISPLAY ALET=%2G.V'34' %T(C#DS11)
C#DS11 = |DS11|

/%DISPLAY ALET=X'0001001B'.V'34' %T(C#DS11)
C#DS11 = |DS11|

/%DISPLAY ALET=%7AR.V'34' %T(C#DS11)
C#DS11 = |DS11|

/%DISPLAY SPID=X'0000000080000200'.V'34' %T(C#DS11)
C#DS11 = |DS11|
```

Im ersten %DISPLAY wird die Speicherstelle mit der Adresse V'34' im Programmraum ausgegeben im Typ der Datendefinition C#DS11.

Alle weiteren %DISPLAY-Kommandos geben die Speicherstelle mit der Adresse V'34' im Datenraum mit der SPID X'0000000080000200' im Typ der Datendefinition C#DS11 aus. Der ALET X'0001001B' zeigt auf denselben Datenraum. Er ist in %2G zwischengespeichert und in %7AR enthalten. Somit bewirken alle vier %DISPLAY-Kommandos dasselbe.

2.

```
%DISPLAY %AR, ALET=%4AR.V'200', ALET=%4AR.V'0', ALET=%4AR.V'10'
%AR ( 0: 15)
( 0) 00000000 ( 1) 00010003 ( 2) 00010004 ( 3) 00010005 ( 4) 00010006
( 5) 00000000 ( 6) 00000000 ( 7) 00000000 ( 8) 00000000 ( 9) 00000000
(10) 00000000 (11) 00000000 (12) 00000000 (13) 00000000 (14) 00000000
(15) 00000000

V'00000200' = *ABSOLUT + #'00000200'
00000200 (00000000) 00001B30 ....

V'00000000' = *ABSOLUT + #'00000000'
00000000 (00000000) 0800C301 ..C.

V'00000010' = *ABSOLUT + #'00000010'
00000010 (00000010) 110300D1 ...J

%D ALET=%4AR.V'200'->.(V'0' %FL1) %L(V'10' %XL1)
V'00001B38' = *ABSOLUT + #'00001B38'
00001B38 (000004D0) D4E4C5D3 D3C5D940 C1D5D5C5 D3C9C5E2 MUELLER ANNELIES
00001B48 (000004E0) C5 E
```

Über den ALET in Zugriffsregister 4 wird ein Datenraum angesprochen. Alle virtuellen Adressen (V'200', V'0', V'4') der nachfolgenden komplexen Speicherreferenz werden aus diesem Datenraum entnommen. Das resultierende Speicherobjekt wird im Dump-Format ausgegeben. AID rechnet Adresse V'1B30' + #'08' = Adresse V'1B38' und gibt ab dieser Adresse 17 Bytes (X'11') aus.

## 3.4 %DS, Information über Datenräume

Erweiterung im AID-Basishandbuch, Abschnitt 9.6, Systeminformationen.

Erweiterung im Handbuch für das Testen auf Maschinencode-Ebene, und in den sprachspezifischen Handbüchern, AID-Kommando %DISPLAY.

%DS[(ALET/SPID-qua)]

Im AID-Kommando %DISPLAY können Sie das Schlüsselwort %DS[(ALET/SPID-qua)] angeben. AID gibt dann Informationen über die aktiven Datenräume aus.

Ohne Angabe einer ALET/SPID-Qualifikation gibt AID die SPID und ALETs von allen Datenräumen aus, die vom Programm benutzt werden. Außerdem gibt AID die Größe der einzelnen Datenräume in Bytes aus. Für den Programmraum wird ebenfalls eine SPID ausgegeben. Zugehöriger ALET und zugehörige SIZE erhalten den Wert 0.

Bei Angabe einer ALET-Qualifikation gibt AID die zugehörige SPID aus.

Bei Angabe einer SPID-Qualifikation gibt AID die zugehörigen ALETs aus.

Wenn die in der Qualifikation angegebene SPID oder der angegebene ALET nicht vorhanden ist, gibt AID eine Fehlermeldung aus.

### Beispiel

```
/%DISPLAY %DS
** ITN: #'000100033' *** TSN: 0BIV *****
CURRENT PC: 000000E0      CSECT: DS2S4A *****

SPID          ALET          SIZE
0001003300000031    00000000      0
00000000800000800    0001001B    409600
00000000800000800    0001001C    409600
00000000800000900    00010011    131072
00000000800000900    00010012    131072

/%DISPLAY %DS(SPID=X'00000000800000800')

    SPID  = 00000000800000800
    ALET  = 0001001B
    ALET  = 0001001C

/%DISPLAY %DS(ALET=X'00010011')

    ALET  = 00010011
    SPID  = 00000000800000900
```

Bei der Ausgabe aller aktiven Datenräume mit %DS wird jeder Datenraum mit seiner SPID, allen seinen ALETs und seiner Größe (SIZE) in Bytes ausgegeben.

Bei der Abfrage von Informationen zu einem bestimmten Datenraum, also mit einer ALET/SPID-Qualifikation, gibt AID die zugehörigen ALETs bzw. die zugehörige SPID aus.

## 3.5 %TRACE-Protokoll

Erweiterung im Handbuch für das Testen auf Maschinencode-Ebene, AID-Kommando %TRACE.

Im %TRACE-Protokoll werden die neuen Befehle, die zur ESA-Unterstützung angeboten werden, mit ausgegeben. Die Zugriffsregister werden mit ARn=... ausgegeben und Adressen, die in einem Datenraum liegen, werden mit einem Stern markiert.

### Beispiel

MODS+15008	STAM	R14,R12,C(R13)	2	*A2=0001A00C	AR14=000010A4
				AR15=00016008	AR0=00000000
				AR1=0001A0A0	AR2=00000000
				AR3=00000000	AR4=000010A0
				AR5=0001A0A0	AR6=00000000
				AR7=00000000	AR8=00000000
				AR9=00000000	AR10=6F00101A
				AR11=00000000	AR12=00000000

Da der AR-Modus eingeschaltet ist, und Zugriffsregister 13  $\neq$  0 ist, liegt die Adresse A2 in einem Datenraum. Dies ist im Protokoll an dem Stern vor A2 (\*A2=...) zu erkennen.

Mit dem ALET in Zugriffsregister 13 wird der Datenraum bestimmt. Die Adresse A2 wird dann aus dem Inhalt von Mehrzweckregister 13 + Distanz #'C' errechnet.

## 4 Meldungen

I489 ADDRESS IN %ON %WRITE COMMAND INVALID

### **Bedeutung**

Die im %ON %WRITE-Kommando angegebene Adresse ist ungültig.

### **Maßnahme**

Gültige Adresse angeben.

I490 %ON %WRITE IS NOT ALLOWED WHEN %T OR %C WITH %INSTR/%B/%BAL IS ACTIVE

### **Bedeutung**

Es ist nicht erlaubt, ein %ON %WRITE-Kommando einzugeben, solange ein %TRACE- oder ein %CONTROLn-Kommando mit maschinennahem 'kriterium' (%INSTR, %B, %BAL) aktiv ist.

### **Maßnahme**

Entfernen des %TRACE durch Eingabe des Kommandos %TRACE 1 %INSTR; entfernen des %CONTROLn durch Eingabe des Kommandos %REMOVE %Cn.

I491 %T OR %C WITH %INSTR/%B/%BAL IS NOT ALLOWED WHEN %ON %WRITE IS ACTIV

### **Bedeutung**

Es ist nicht erlaubt, ein %TRACE- oder %CONTROLn-Kommando mit maschinennahem 'kriterium' (%INSTR, %B, %BAL) einzugeben, solange ein %ON %WRITE-Kommando aktiv ist.

### **Maßnahme**

Entfernen des %ON %WRITE-Kommandos durch %REMOVE %WRITE.

I492 %ON %WRITE IS NOT ALLOWED WHEN PROGRAM IS NOT LOADED

### **Bedeutung**

Das Kommando %ON %WRITE darf nur eingegeben werden, wenn ein Programm geladen ist.

### **Maßnahme**

Programm laden und %ON %WRITE-Kommando erneut eingeben.

I493 %ON %WRITE COMMAND IS ONLY ALLOWED IN STATUS TU

### **Bedeutung**

Das Kommando %ON %WRITE ist nur im nicht-privilegierten Zustand erlaubt.

### **Maßnahme**

Übergang in den nicht-privilegierten Zustand.

- I494 TESTPOINTS MUST NOT BE SET INTO CLASS 6 MEMORY POOLS  
**Bedeutung**  
Das Setzen von Testpunkten in Klasse-6-Memory-Pools ist nicht erlaubt.
- I495 %ON %WRITE NOT SET OR ALREADY REMOVED  
**Bedeutung**  
Das Kommando %ON %WRITE wurde nicht eingegeben oder bereits gelöscht.
- I496 WARNING: PREVIOUSLY DEFINED EVENT %WRITE IS REPLACED  
**Bedeutung**  
Warnung: Da jeweils nur ein %WRITE-Ereignis angemeldet sein kann, wird bei der Eingabe eines weiteren %ON %WRITE-Kommandos das vorher gegebene überschrieben.
- I497 LENGTH OF AREA IN %ON %WRITE COMMAND EXCEEDS 1024 BYTES  
**Bedeutung**  
Mit dem %ON %WRITE-Kommando kann maximal ein Bereich von 1024 Bytes überwacht werden.  
**Maßnahme**  
Kleinere Länge angeben.
- I498 HARDWARE DOES NOT SUPPORT ALET/SPID-QUALIFICATION  
**Bedeutung**  
ALET/SPID-Qualifikationen bei Operanden werden nur auf ESA-Anlagen unterstützt.  
**Maßnahme**  
Eingabe ohne ALET/SPID-Qualifikation
- I499 VALUE OF ALET IS UNKNOWN  
**Bedeutung**  
Der Wert des ALET ist unbekannt.  
**Maßnahme**  
Richtigen Wert angeben.
- I500 NUMBER OF DATA SPACES CHANGED DURING EXECUTION OF COMMAND  
**Bedeutung**  
Die Anzahl der Datenräume hat sich während der Ausführung des Kommandos geändert.  
**Maßnahme**  
Kommando später noch einmal eingeben.
- I502 NO DATA SPACES DUMPED  
**Bedeutung**  
In der angegebenen Dump-Datei gibt es keine Dump-Bereiche aus Datenräumen.
- I503 DATA SPACE WITH SPECIFIED SPID NOT FOUND IN DUMP FILE  
**Bedeutung**  
Der Datenraum mit der angegebenen SPID befindet sich nicht in der Dump-Datei.

I504 DATA SPACE WITH SPECIFIED ALET NOT FOUND IN DUMP FILE

**Bedeutung**

Der Datenraum mit dem angegebenen ALET befindet sich nicht in der Dump-Datei.

I505 HARDWARE DOES NOT SUPPORT ESA

**Bedeutung**

Auf dieser Anlage wird ESA nicht unterstützt.

I506 NO DATA SPACES DEFINED

**Bedeutung**

Zu diesem Zeitpunkt sind keine Datenräume definiert.





# Fachwörter

## Adressierungsmodus

bestimmt, wie die Adressen in den Maschinenbefehlen umgesetzt werden sollen. AID übernimmt den Adressierungsmodus des Testobjekts. Das trifft sowohl für die Adreßbreite (24- oder 31-Bit) von Programmen auf XS-Anlagen (%AMODE), als auch für die Adressierung von Datenräumen auf ESA-Anlagen (%ASC) zu.

Die Systeminformation %AMODE betrifft die Adreßbreite. Sie kann mit %DISPLAY abgefragt werden und mit `%MOVE %MODE{24|31} INTO %AMODE` geändert werden.

Die Systeminformation %ASC betrifft die Adressierung von Datenräumen. Sie kann mit %DISPLAY abgefragt werden und gibt u.a. Auskunft über den AR-Modus.

## ALET

gibt es nur auf ESA-Anlagen. Der ALET (access list entry token) ist eine Art Zeiger auf die Zugriffsliste (AL), über die die Zugriffe auf einen Datenraum verwaltet werden. Die ALETs sind in den Zugriffsregistern enthalten und können in der ALET-Qualifikation angegeben werden.

## AR-Modus

Den AR-Modus (access register mode) gibt es auf ESA-Anlagen. Er entscheidet, ob die Zugriffsregister bei der Adreßumsetzung mit ausgewertet werden oder nicht. Wenn %DISPLAY %ASC den sedezimalen Wert X'01' ausgibt, ist der AR-Modus gesetzt, die Zugriffsregister werden ausgewertet, und somit können Adressen in Datenräumen angesprochen werden.

## Datenraum

Auf ESA-Anlagen gibt es die Möglichkeit neben dem Programmraum, der dem bisherigen Adreßraum entspricht, weitere Adreßräume für Daten, die Datenräume (data spaces), zu nutzen. Speicherobjekte, die sich in einem Datenraum befinden, können auf Maschinencode-Ebene über eine virtuelle Adresse mit ALET/SPID-Qualifikation angesprochen werden.

## ESA

(Enterprise System Architecture) ist ein neues Hardware-Konzept, mit dem zusätzliche Adreßräume für Daten nutzbar gemacht werden. Die Adressierung innerhalb eines solchen Datenraums ist über die Zugriffsregister realisiert. Bei eingeschaltetem AR-Modus wird bei der Adreßumsetzung in einem Befehl das zum Basisregister korrespondierende Zugriffsregister mit ausgewertet. Anlagen mit dieser Hardware werden ESA-Anlagen genannt; z.Zt. sind das die Anlagen H130 und Z5. Zum Speicherkonzept auf ESA-Anlagen siehe Handbuch "Makroaufrufe an den Ablaufteil".

### Programmraum

ist auf ESA-Anlagen der Adreßraum, in dem das Programm abläuft. Er entspricht dem Adreßraum auf Nicht-ESA-Anlagen.

### SPID

(space identification) ist ein systemweit eindeutiges Kennzeichen für einen Datenraum. Die SPID wird beim Anlegen eines Datenraumes vergeben.

### Zugriffsregister

$\%nAR$ ,  $0 \leq n \leq 15$ . Die Zugriffsregister sind auf ESA-Anlagen parallel zu den Mehrzweckregistern vorhanden. Sie werden zur Adressierung von Datenräumen benötigt. In ihnen ist ein ALET (access list entry token) eingetragen, der über die Zugriffsliste auf einen Datenraum verweist. Wenn der AR-Modus eingeschaltet ist, werden die Zugriffsregister bei der Adreßumsetzung mit ausgewertet.

# Literatur

## **AID (BS2000)**

Advanced Interactive Debugger

### **Basishandbuch**

Benutzerhandbuch

*Zielgruppe*

Programmierer im BS2000

*Inhalt*

Überblick über AID, Beschreibung der Sachverhalte und Operanden, die für alle Programmierspreachen gleich sind.

Meldungen, Gegenüberstellung von AID-IDA

*Einsatz*

Testen von Programmen im Dialog- und Stapelbetrieb

## **AID (BS2000)**

Advanced Interactive Debugger

### **Testen auf Maschinencode-Ebene**

Benutzerhandbuch

*Zielgruppe*

Programmierer im BS2000

*Inhalt*

Beschreibung der AID-Kommandos für das Testen auf Maschinencode-Ebene; Anwendungsbeispiel

*Einsatz*

Testen von Programmen im Dialog- und Stapelbetrieb

**AID (BS2000)**

Advanced Interactive Debugger

**Testen von COBOL-Programmen**

Benutzerhandbuch

*Zielgruppe*

COBOL-Programmierer

*Inhalt*

Beschreibung der AID-Kommandos für das symbolische Testen von COBOL-Programmen;

Anwendungsbeispiel

*Einsatz*

Testen von COBOL-Programmen im Dialog- und Stapelbetrieb

**AID (BS2000)**

Advanced Interactive Debugger

**Testen von FORTRAN-Programmen**

Benutzerhandbuch

*Zielgruppe*

FORTRAN-Programmierer

*Inhalt*

Beschreibung der AID-Kommandos für das symbolische Testen von FORTRAN-Programmen;

Anwendungsbeispiel

*Einsatz*

Testen von FORTRAN-Programmen im Dialog- und Stapelbetrieb

**AID (BS2000)**

Advanced Interactive Debugger

**Testen von PL/I-Programmen**

Benutzerhandbuch

*Zielgruppe*

PL/I-Programmierer

*Inhalt*

Beschreibung der AID-Kommandos für das symbolische Testen von PL/I-Programmen;

Anwendungsbeispiel

*Einsatz*

Testen von PL/I-Programmen im Dialog- und Stapelbetrieb

**AID (BS2000)**

Advanced Interactive Debugger

**Testen von ASSEMBH-Programmen**

Benutzerhandbuch

*Zielgruppe*

Assembler-Programmierer

*Inhalt*

Beschreibung der AID-Kommandos für das symbolische Testen von ASSEMBH-Programmen;

Anwendungsbeispiel

*Einsatz*

Testen von ASSEMBH-Programmen im Dialog- und Stapelbetrieb

**AID (BS2000)**

Advanced Interactive Debugger

**Testen von C-Programmen**

Benutzerhandbuch

*Zielgruppe*

C-Programmierer

*Inhalt*

Beschreibung der AID-Kommandos für das symbolische Testen von C-Programmen;

Anwendungsbeispiel

*Einsatz*

Testen von C-Programmen im Dialog- und Stapelbetrieb

**BS2000****Makroaufrufe an den Ablaufteil**

Benutzerhandbuch

*Zielgruppe*

- BS2000-Assembler-Programmierer (nicht privilegiert)
- Systemverwalter

*Inhalt*

- Alle Makroaufrufe an den Ablaufteil in lexikalischer Reihenfolge mit Hinweisen und Beispielen, einschließlich ausgewählter Makroaufrufe für das DVS und für TIAM
- Zusammenstellung der Makroaufrufe nach Anwendungsgebieten
- Ausführlicher Lernteil über Ereignissteuerung, Serialisation, Inter-Task-Kommunikation, Contingencies

*Einsatz*

BS2000-Anwendungsprogramme

**Bestellen von Handbüchern**

Die aufgeführten Handbücher finden Sie mit ihren Bestellnummern im *Druckschriftenverzeichnis* der Siemens Nixdorf Informationssysteme AG. Neu erschienene Titel finden Sie in den *Druckschriften-Neuerscheinungen*.

Beide Veröffentlichungen erhalten Sie regelmäßig, wenn Sie in den entsprechenden Verteiler aufgenommen sind. Wenden Sie sich bitte hierfür an Ihre zuständige Geschäftsstelle. Dort können Sie auch die Handbücher bestellen.

# Stichwörter

%ASC 12  
%DS 19  
%nAR 14  
%ON 5  
%TRACE-Protokoll 20  
%WRITE 5

## A

Adreßumsetzung im AR-Modus 12, 14  
Adreßversatz 17  
Adresse im Datenraum 15, 17  
Adresse im Programmraum 17  
Adressierungsmodus 12  
ALET 11, 14, 16, 19  
ALET/SPID-Qualifikation 15  
AR-Modus 12, 14  
ASC-Modus 12

## D

Datenraum 11, 12, 15, 19

## E

ESA 11

## I

indirekte Adressierung 17

## K

Ketten von Ereignissen 5  
komplexe Speicherreferenz 17

## L

Längenmodifikation 17  
Löschen, *write-ereignis* 5

### **M**

Metasyntax 3

### **P**

Programm mit Überlagerungsstruktur 6

Programmraum 11

### **S**

Schlüsselwort 17

Schreibüberwachung 5

SPID 11, 16, 19

Subkommando, Reihenfolge der Bearbeitung 5

symbolische Adressierung (ESA) 11

### **T**

Typmodifikation 17

### **U**

Überlagerungsstruktur 6

Überwachen, schreibender Zugriff 5

Unterbrechungsstelle 6

### **V**

virtuelle Adresse 11, 17

### **W**

*write-ereignis* 5

### **X**

XS-Programme 12

### **Z**

Zugriffsregister 11, 12, 14







# Inhalt

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Konzept des Handbuchs	1
1.2	Änderungen gegenüber AID V2.0A	2
1.3	Metasyntax	3
<b>2</b>	<b>%WRITE(speicherref), die Schreibüberwachung beim %ON</b>	<b>5</b>
2.1	Funktion der Schreibüberwachung	5
2.2	Wechselwirkungen mit anderen AID-Kommandos	7
2.3	Beispiele	8
<b>3</b>	<b>ESA-Unterstützung</b>	<b>11</b>
3.1	Adressierungsmodus	12
3.2	%nAR, die Zugriffsregister	14
3.3	ALET/SPID-Qualifikation	15
3.4	%DS, Information über Datenräume	19
3.5	%TRACE-Protokoll	20
<b>4</b>	<b>Meldungen</b>	<b>21</b>
	<b>Fachwörter</b>	<b>25</b>
	<b>Literatur</b>	<b>27</b>
	<b>Stichwörter</b>	<b>31</b>



---

# AID V2.0B (BS2000)

## Ergänzungen

*Zielgruppe*

Programmierer im BS2000/OSD

*Inhalt*

Erweiterungen zu AID V2.0B, Schreibüberwachung, ESA-Unterstützung

**Ausgabe:** April 1993

**Datei:** AID\_ERG.PDF

BS2000 ist ein eingetragenes Warenzeichen der  
Siemens Nixdorf Informationssysteme AG

Copyright © Siemens Nixdorf Informationssysteme AG, 1998.

Alle Rechte vorbehalten, insbesondere (auch auszugsweise) die der Übersetzung,  
des Nachdrucks, Wiedergabe durch Kopieren oder ähnliche Verfahren.

Zu widerhandlungen verpflichten zu Schadenersatz.

Alle Rechte vorbehalten, insbesondere für den Fall der Patenterteilung oder  
GM-Eintragung.

Liefermöglichkeiten und technische Änderungen vorbehalten.