Dynamisches Binden in Multics

AKSS Seminarvortrag

Daniel Götz

Friedrich-Alexander-Universität Erlangen-Nürnberg

Lehrstuhl für Informatik 4
Verteilte Systeme und Betriebssysteme

14.12.15

Motivation

Welche Vorteile bietet dynamisches Binden gegenüber statischem Binden?

- Programmcode wird nicht kopiert
- Programm muss nicht neu kompiliert werden, um andere Versionen zu benutzen
- Seit 2010 allein 21.000 wissenschaftliche Papiere zum Thema dynamisches Binden auf ACM

Motivation

Warum sollte man sich mit dynamischem Binden in Multics beschäftigen?

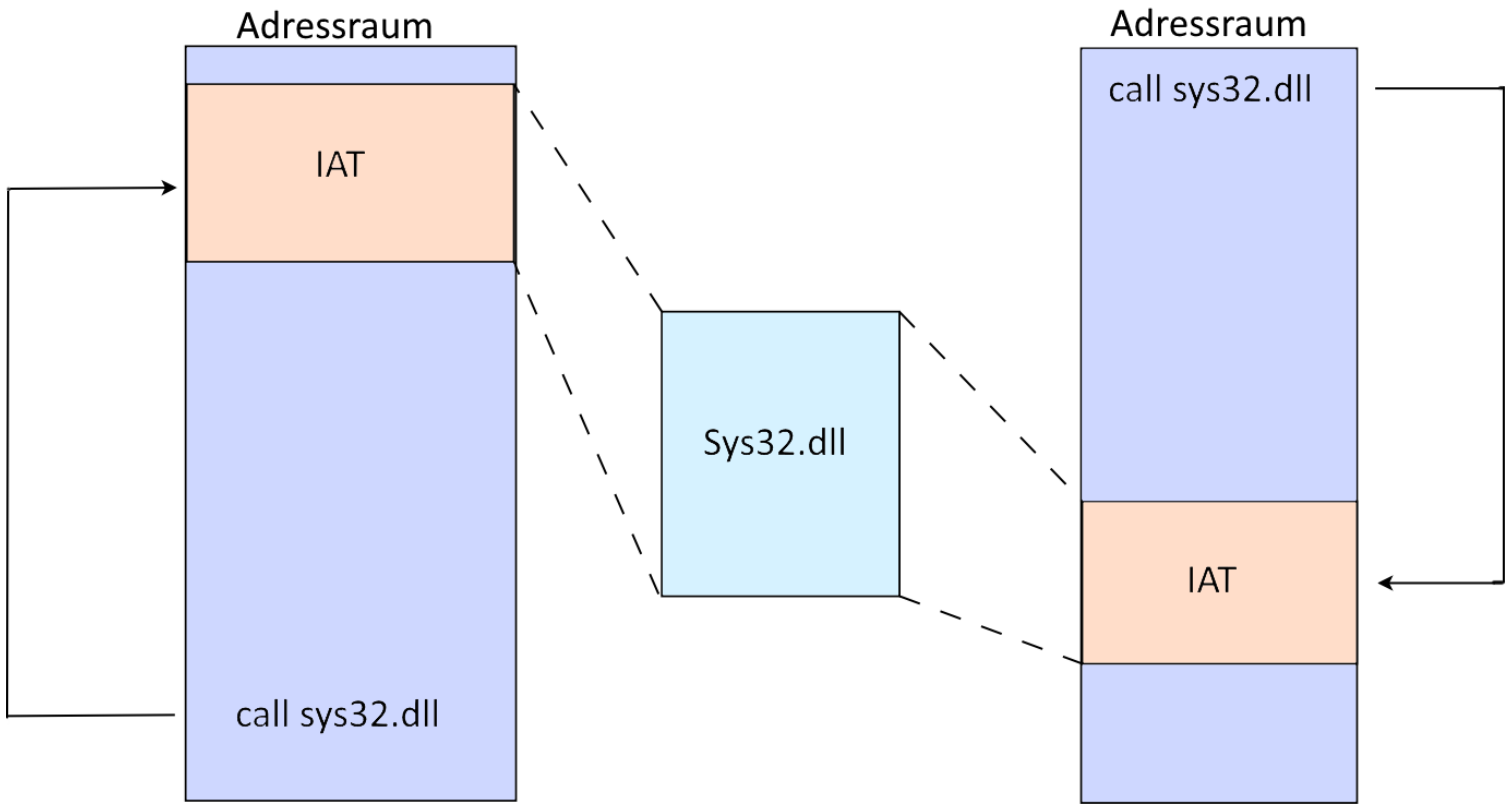
- Multics als effizientes Mehrbenutzerbetriebssystem wegweisender Konzepte
- Dynamisches Binden als innovative Kernkomponente
- Bis heute relevantes Konzept?

The establishing of links at reference time on an as-needed basis is a fundamental service of Multics

Organick, The Multics System: An Examination of Its Structure, S.55

Motivation

- In heutigen Betriebssystemen intensiv genutzt
- Windows: DLLs
- Unix/Mac OS X: Shared Libraries



Gliederung

Motivation

Konzept

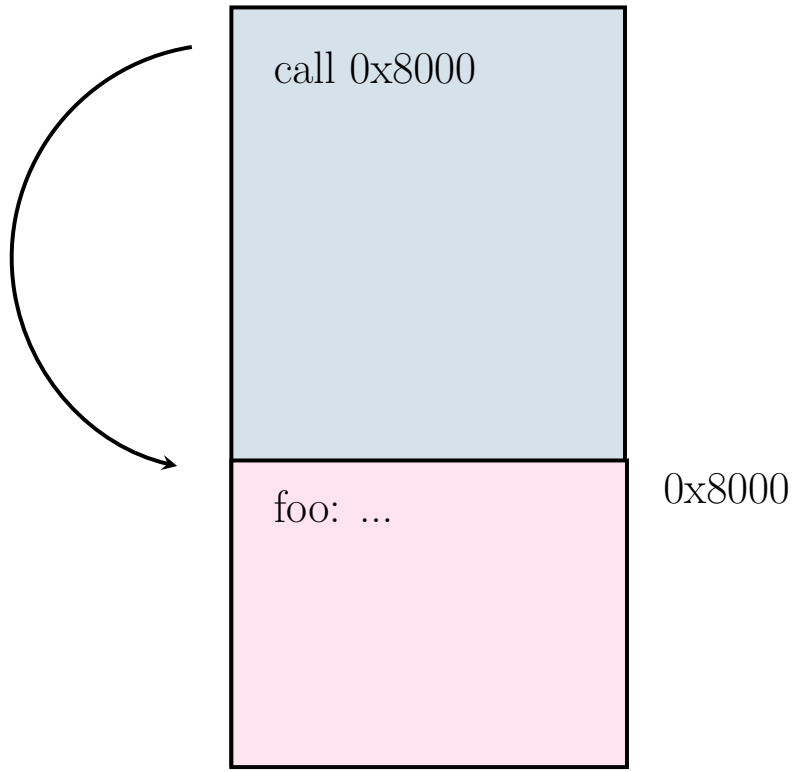
Vor- und Nachteile

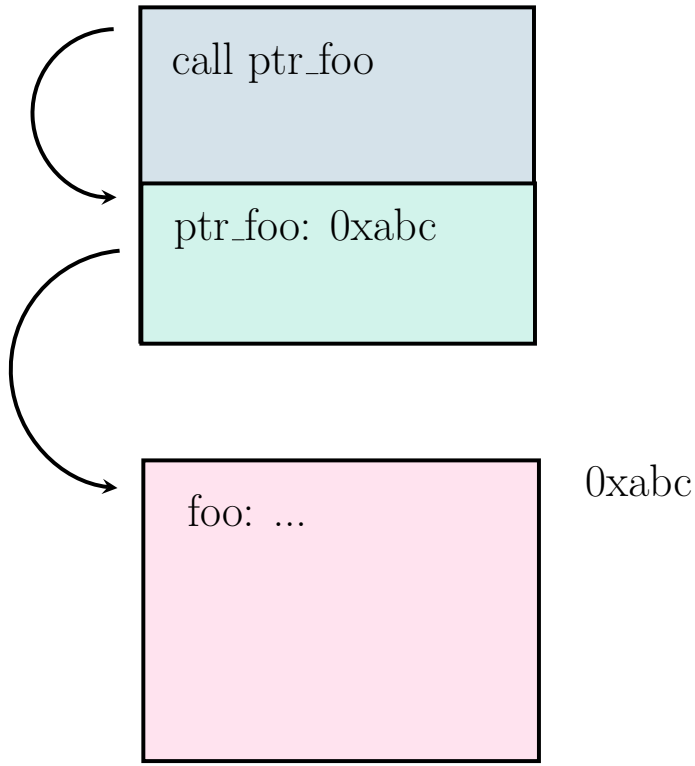
Fazit

Gliederung

Konzept

- Dynamisches Binden enthält keinen kopierten Code, sondern Zeiger
- Aufgabe des dynamischen Binders ist die Auflösung zur Laufzeit





Konzept

- Dynamisches Binden als Hinzufügen von Daten und Algorithmen zur Laufzeit
  - **Was** wird gebunden?
  - **Wann** wird gebunden?
  - **Wie** funktioniert das Binden?

Konzept

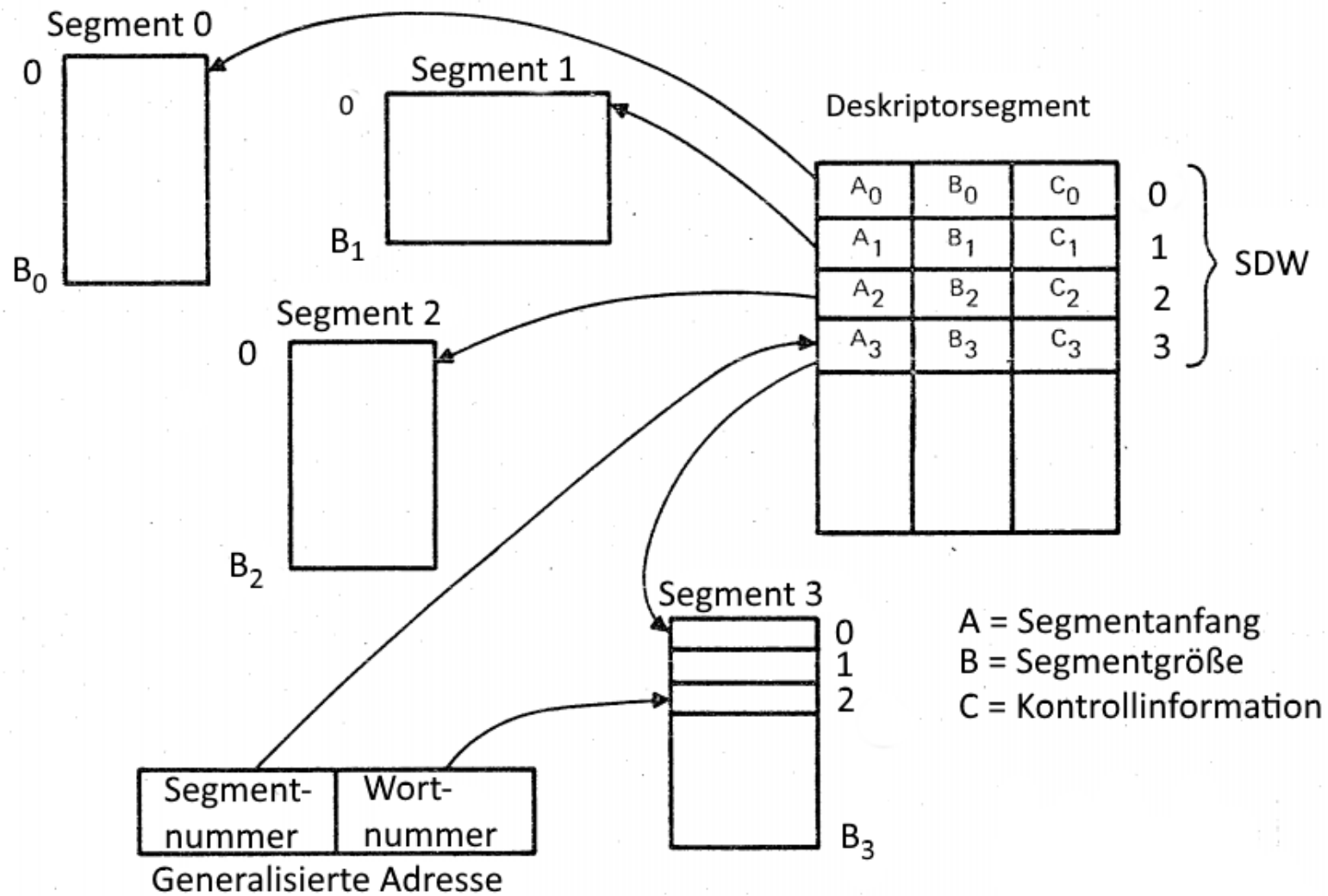
Was wird gebunden?

Multics Adressraum

- Seitennummerierte Segmentierung
- Prozess besteht aus beliebiger Anzahl von Segmenten
- Adressierung von Daten- oder Prozedursegmenten über **generalisierte Adresse** (Segmentnummer|Wortnummer)
- Multics bindet auf **Segmentebene**

Deskriptorsegment

- Datenstruktur für Segmentinformation: **Deskriptorsegment**



Deskriptorsegment

- Deskriptorsegment enthält Deskriptor für alle bekannten Segmente eines Prozesses
- Zentrale Datenstruktur des Prozesses zur Umsetzung der generalisierten Adressen
- Kritisch für das Binden: Zur Laufzeit erweiterbar

Konzept

Wann wird gebunden?

Konzept

1. Befehle können Referenzen auf segmentfremde Operanden enthalten

   `LOAD <Foo>|[x]` $\leftrightarrow$ `LOAD Link#4,bs`

2. Generierung von Verknüpfungen für dynamische Referenzen

   `LOAD Link#4,bs` $\rightarrow$ `Link#4:Ref(Foo|x)`

3. Bindevorgang bei Benutzung einer **unetablierten Verknüpfung**

   `Link#4:Ref(Foo|x)` $\rightarrow$ `?`

Befehlsformat

1. Befehle können Referenzen auf segmentfremde Operanden enthalten

`LOAD <Foo>|[x]` $\leftrightarrow$ `LOAD Link#4,bs`

- BR: Auswahl eines **Adressbasisregisters** (ABR)
- B: Setzen des B-Flags, um Nutzung des ABRs zu signalisieren
- TM/TD: Setzen des korrekten Modus für den indirekten Zugriff

3 Bit	15 Bit	9 Bit	1Bit	1Bit	1Bit	2 Bit	4 Bit
BR	ADRESSE	OPCODE	U	I	B	TM	TD

Befehlsformat

Adressbasisregister

- 4 Adressbasisregisterpaare
- Inhalt wird als generalisierte Adresse interpretiert
- Enthalten u.a. Ort der Verknüpfungen bzw. des **Bindungssegments**

paired address base register

SP	6	segment number	7	0	0		word number		1	0		7
LP	4	segment number	5	0	0		word number		1	0		5
BP	2	segment number	3	0	0		word number		1	0		3
AP	0	segment number	1	0	0		word number		1	0		1

Bindungssegment

2. Generierung von Verknüpfungen für dynamische Referenzen

- Bindungssegment enthält Verknüpfungen für alle Referenzen
- Maschinenbefehl referenziert nur die Verknüpfung
- Verknüpfung umgesetzt durch **ITS-Zeiger**

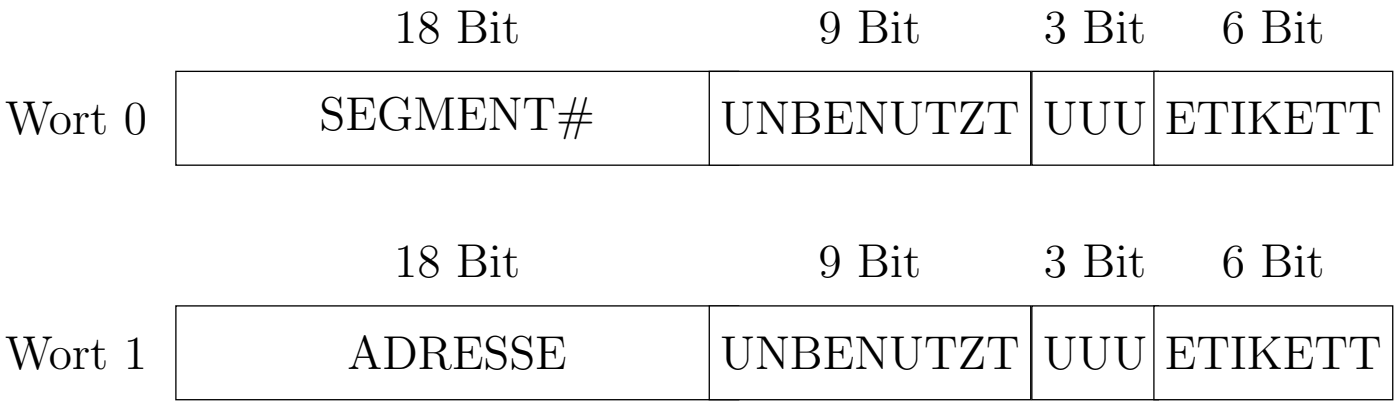
Indirekte Zeiger

- Bestehen aus jeweils 2 36-Bit Worten
- Verknüpfung zu Segment inklusive Versatz
- ITS-Zeiger (*Indirect to Segment*)
  - Verweist fest auf verknüpftes Segment
  - Adressierung über generalisierte Adresse
- ITB-Zeiger (*Indirect to Base*)
  - Verweist über Adressbasisregister auf beliebiges Segment
  - Adressierung über Wortnummer und Auswahl eines Adressbasisregisters

ITS-Zeiger

3. Bindevorgang bei Benutzung einer unetablierten Verknüpfung

- Vor der Etablierung Kennzeichnung des Segments durch symbolischen Namen
- Enthält nach Etablierung Segment- und Wortnummer
  ↪ generalisierte Adresse
- Etablierungsstatus im Etikett des 0-Wortes
  - FT2: **Aktivierung des Binders**
  - ITS: Zugriff über etablierte Verknüpfung



Konzept

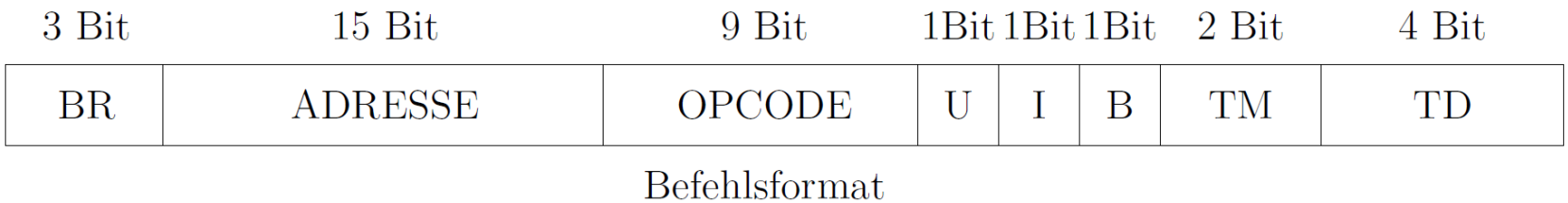
Wie funktioniert das Binden?

Konzept

1. Maschinenbefehl referenziert ITS-Zeiger
2. Verknüpfung im Bindungssegment ist unetabliert und löst einen Bindefehler aus
3. Segment wird im Speicher des aufrufenden Prozesses eingeblendet
4. Verknüpfung wird etabliert

1. Maschinenbefehl

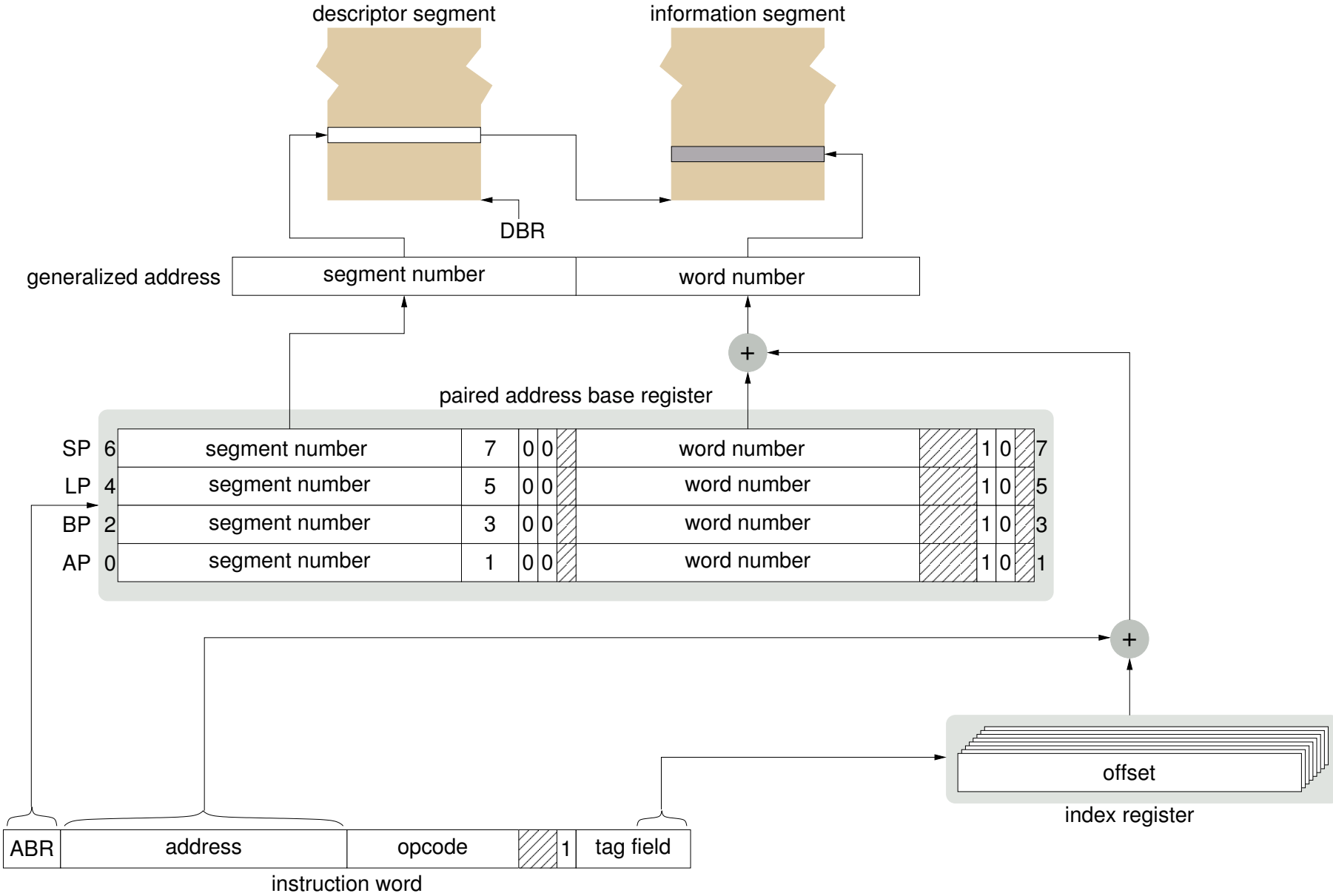
1. Maschinenbefehl referenziert ITS-Zeiger
   - Adressbasisregisterflagge und BR-Feld gesetzt
   - Adressbasisregister spezifiziert ITS-Zeiger
   - Etikett wählt indiziert-indirekten Zugriff



Befehlsformat

ABR-Adressierung

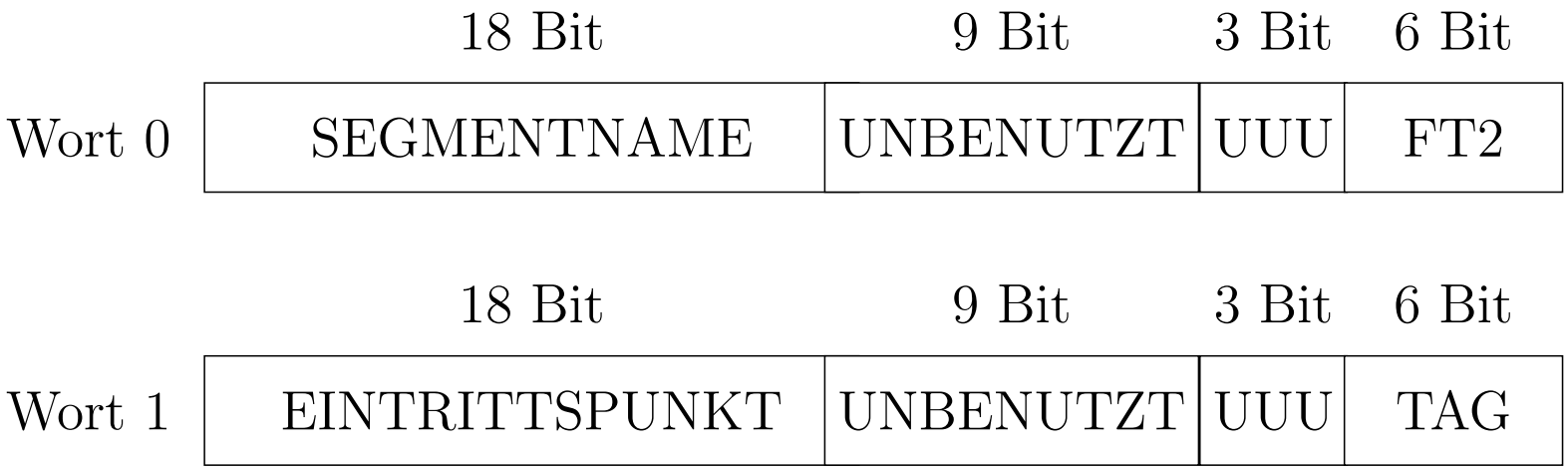
Quelle: W. Schröder-Preikschat, Betriebssystemtechnik



2. Unetablierte Verknüpfung

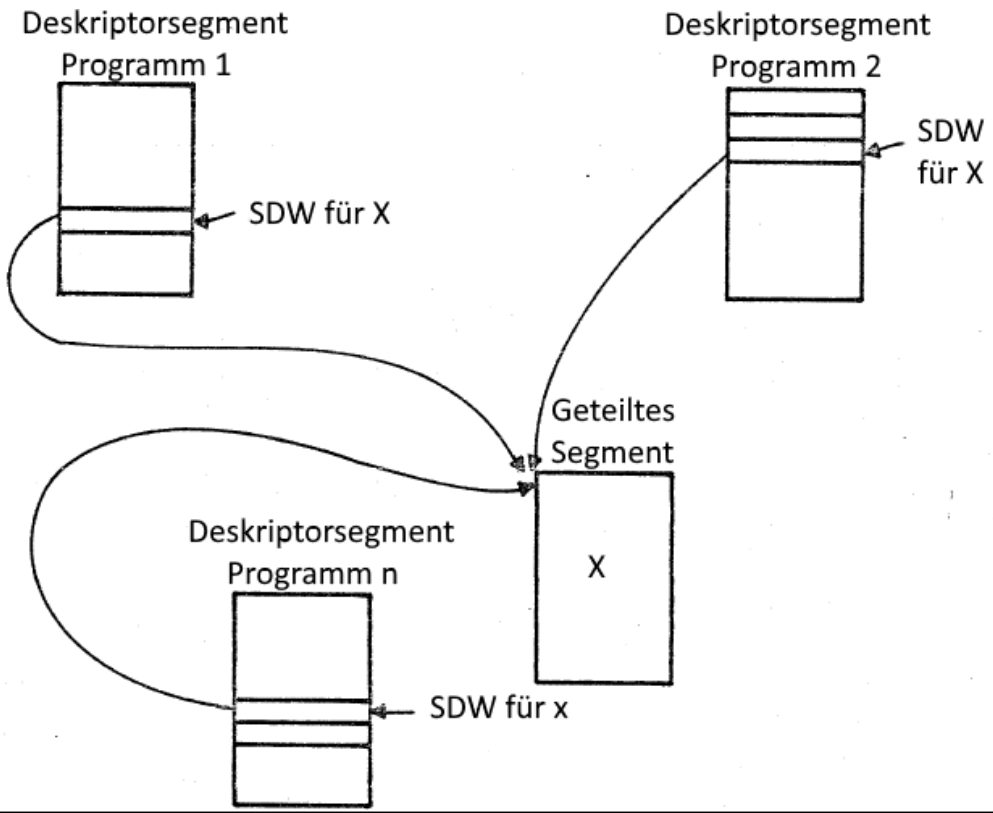
2. Verknüpfung im Bindungssegment ist unetabliert und löst einen Bindefehler aus

- Symbolischer Name im Dateisystem statt generalisierter Adresse
- Magic Bytes “FT2” im Etikett lösen Bindefehler aus
- Finden des Segmentes über Suchregeln



3. Einblendung

3. Segment wird im Speicher des aufrufenden Prozesses eingeblendet
   - Betriebssystem lädt Segment in Hauptspeicher, falls nötig
   - Deskriptorsegmenteintrag unter nächster freier Nummer
   - Nummer unterschiedlich je nach Zeitpunkt der Etablierung



4. Etablierung

4. Verknüpfung wird etabliert

- Generalisierte Adresse ersetzt symbolischen Namen
- FT2-Etikett auswechseln durch ITS
- Erneute Durchführung des Befehls

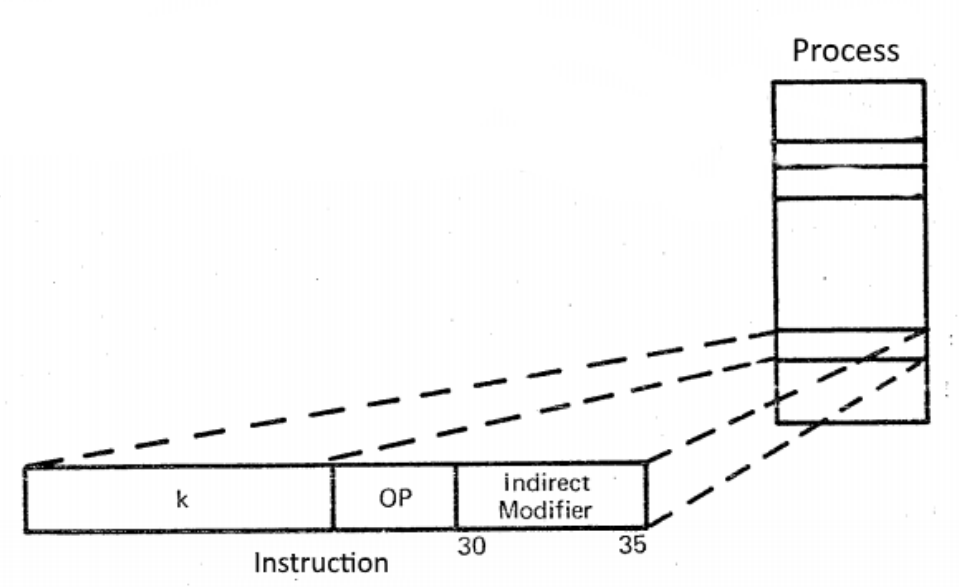
	18 Bit	9 Bit	3 Bit	6 Bit
Wort 0	SEGMENT#	UNBENUTZT	UUU	**ITS**

	18 Bit	9 Bit	3 Bit	6 Bit
Wort 1	ADRESSE	UNBENUTZT	UUU	ETIKETT

Grafische Darstellung des Bindevorgangs

vgl. GE-Manual S.26

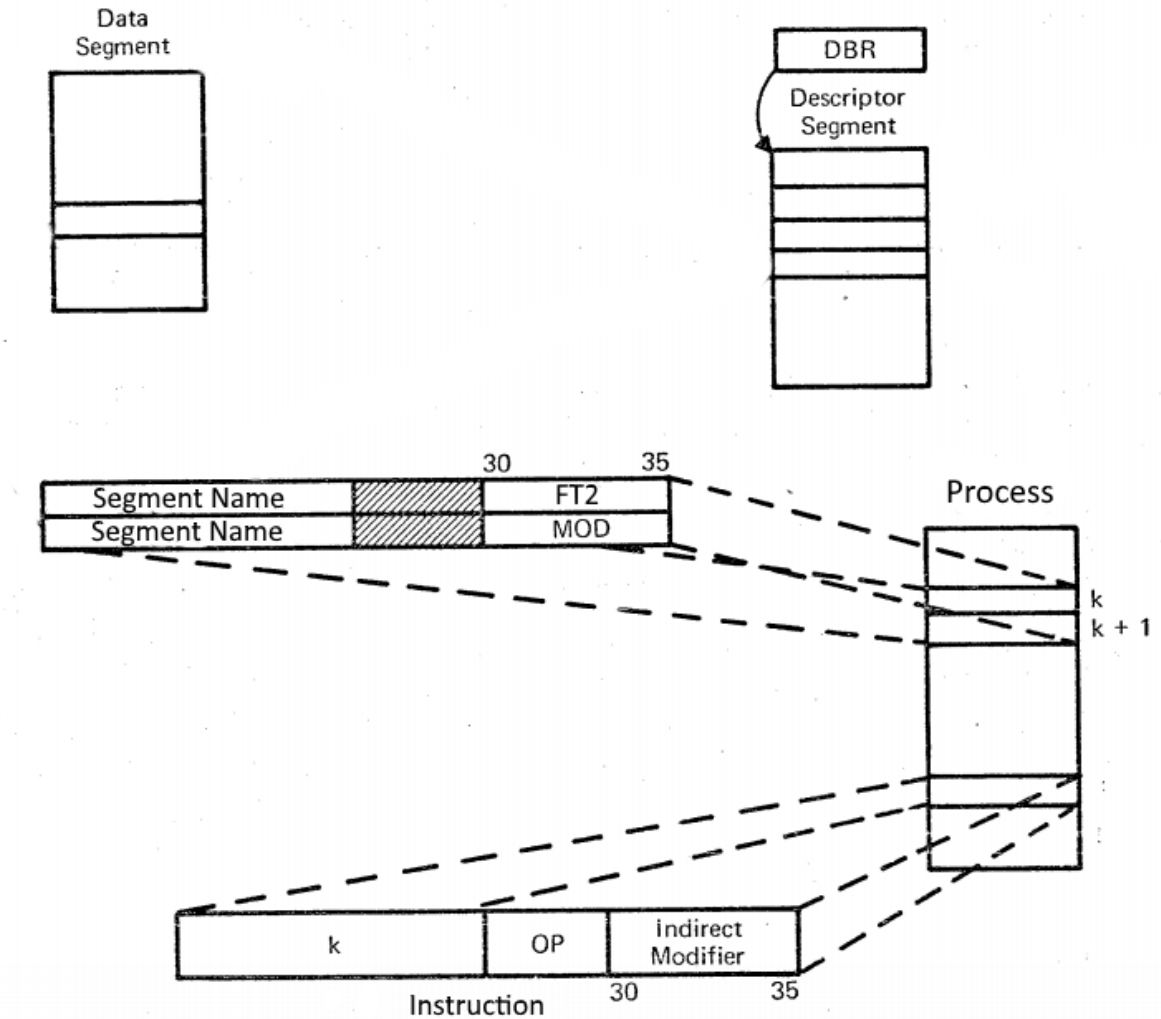
1. Maschinenbefehl referenziert ITS-Zeiger



Grafische Darstellung des Bindevorgangs

vgl. GE-Manual S.26

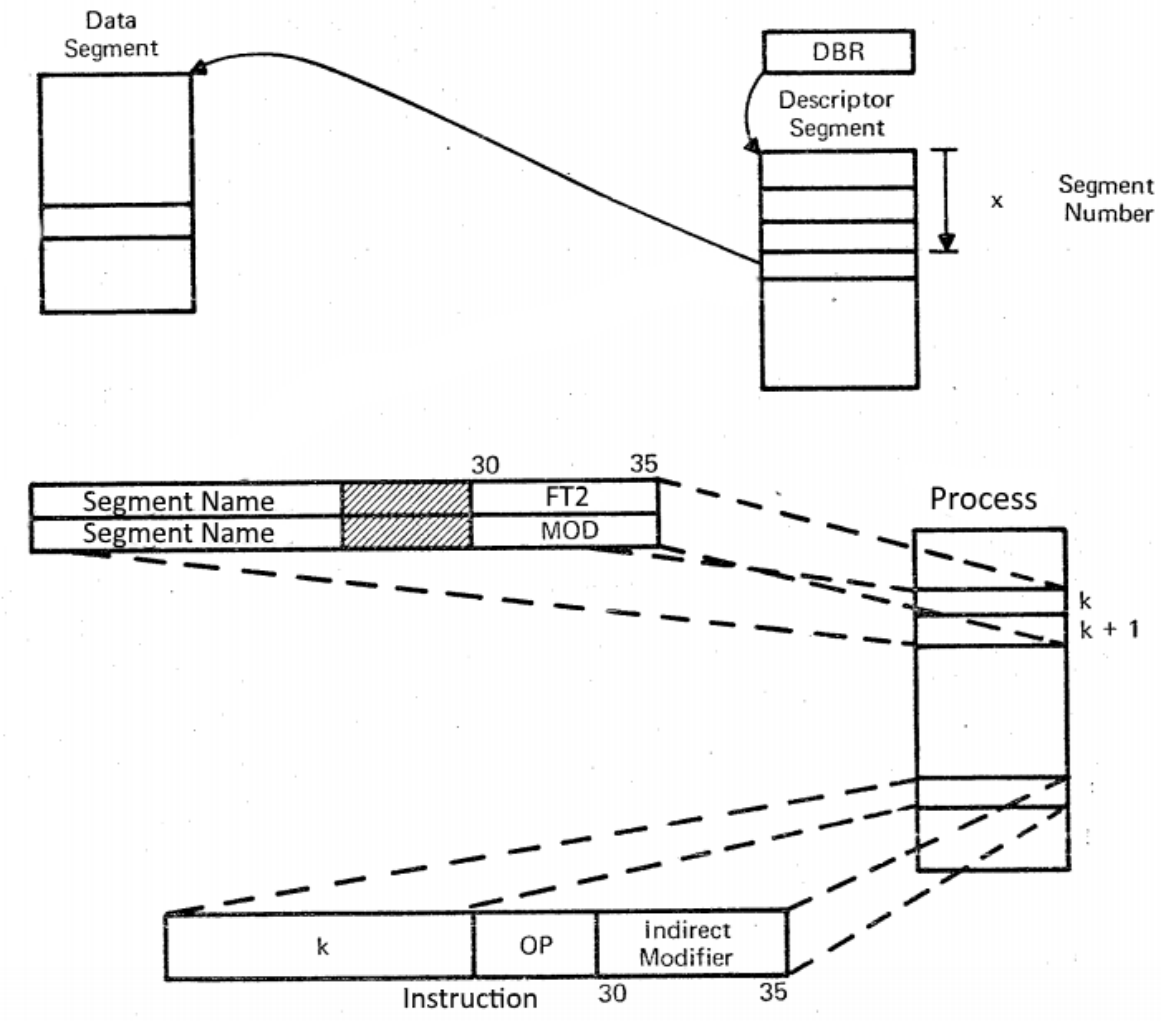
2. Verknüpfung im Bindungssegment ist unetabliert und löst einen Bindefehler aus



Grafische Darstellung des Bindevorgangs

vgl. GE-Manual S.26

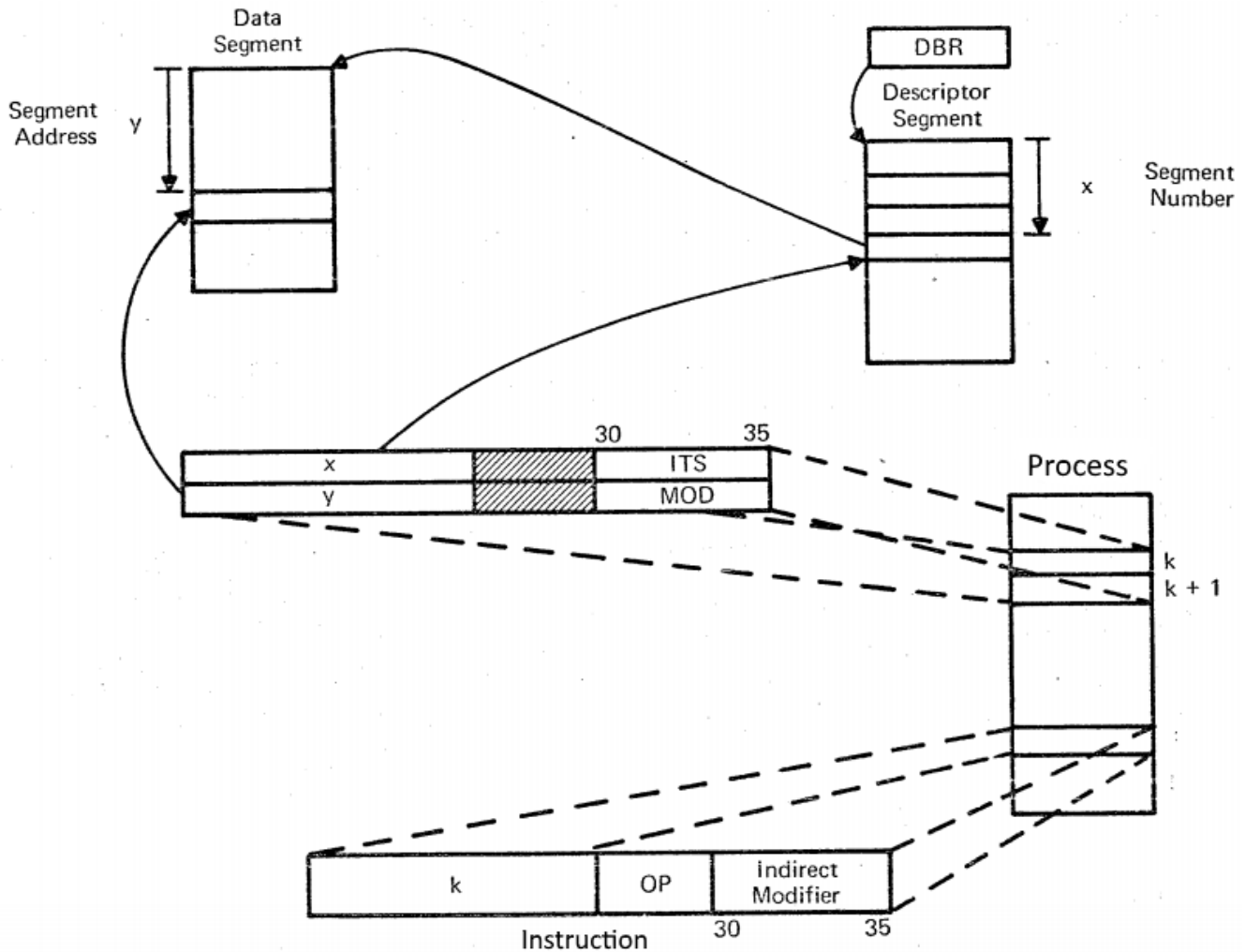
3. Segment wird im Speicher des aufrufenden Prozesses eingeblendet



Grafische Darstellung des Bindevorgangs

vgl. GE-Manual S.26

4. Verknüpfung wird etabliert



Prozedurbindung

- Prinzipiell analog zur Datensegmentbindung
- Prozeduren können wiederum Referenzen enthalten
- Beim Binden eines Prozedursegments wird ein Prototyp der Bindungssektion kopiert
- Für den nutzenden Prozess werden die Verknüpfungen dann jeweils etabliert

Gliederung

Vorteile für Multics

+ Sparen von Speicher
+ Automatisiertes, einfaches Binden über symbolischen Namen
+ Mehrfache Indirektionen möglich für Prozeduren
+ Leichtes Austauschen/Auswählen von Bibliotheken möglich

Nachteile für Multics

- Eventuelle Sicherheitslücken im Binder
- Namenskonflikte relativ wahrscheinlich bei generischen Namen (z.B. “error” für Prozedur)
- Keine Bindungssektion für Datensegmente
- Etablieren von Verknüpfungen teuer
- Eventuell viele Bindungssektionen pro Prozess

Gliederung

Fazit

- Dynamisches Binden ist Schlüsselkonzept für Programmierung für Multics
- Sinnvolle Automatismen und Freiheiten für Entwickler
- Nachteile zu dieser Zeit vertretbar
- Konzept bis heute sehr relevant

Diskussion