KfK 3156
März 1981

Quasiparalleles Arbeiten mehrerer Mikrorechner, gesteuert über einen Zentralrechner

W. Heep, K. Rietzschel, K.-D. Rusch
Hauptabteilung Datenverarbeitung und Instrumentierung

Kernforschungszentrum Karlsruhe

KERNFORSCHUNGSZENTRUM KARLSRUHE

Hauptabteilung Datenverarbeitung
und Instrumentierung

KfK 3156

Quasiparalleles Arbeiten mehrerer Mikrorechner, gesteuert über einen Zentralrechner.

W. Heep
K. Rietzschel
K.-D. Rusch

Kernforschungszentrum Karlsruhe GmbH, Karlsruhe.

Kernforschungszentrum Karlsruhe GmbH
ISSN 0303-4003

Zusammenfassung

Mehrere selbständige Komponenten (Mikrorechner) werden über eine serielle Schnittstelle an einen Hauptrechner gekoppelt. Verschiedene Basisbefehle ermöglichen den Austausch von Daten und Steuerinformation. Die Synchronisation des Prozeßablaufes erfolgt über Ereignisflaggen. Operateurbefehle erlauben die Kontrolle des Prozeßzustandes.

Abstract

A software monitor in a central computer supervising several microcomputers.
Several autonomous microcomputers are coupled by a serial interface to a central computer. A basic instruction set allows exchange of data and control information. The process is synchronized and controlled by event flags. Terminal orientated commands allow the user to supervise and modify the process activities.

Inhaltsverzeichnis

1. Einführung

Bei der Entwicklung komplexer Analysenstände führte die Modularisierung zu Standards in Hardware und unterstützender Software |1|. Die "komplexen Analysenstände" sind durch selbständige Komponenten gekennzeichnet, die jeweils durch einen Mikrorechner gesteuert werden. Besteht ein solcher Analysenstand aus n Komponenten, so ist es Aufgabe des Hauptrechners den Informationsfluß der einzelnen Komponenten zentral zu verwalten und synchronisierend die Komponenten zu aktivieren.

1.1 Ziel der Entwicklung

Im allgemeinen Fall stößt der Hauptrechner (HR) in einer Komponente K(I) die Ausführung einer Funktion P(I,F) an. Der Funktionsablauf innerhalb K wird durch den Mikrorechner (MR) gesteuert und durch ein Software-Programm beschrieben. Der erfolgreiche oder auch fehlerhafte Abschluß der Funktion P(I,F) wird dem HR als Fertigmeldung mitgeteilt.
Ziel der Entwicklung ist es, den gesamten Prozeßablauf zentral im HR beschreiben zu können. Diese Prozeßbeschreibung sollte befreit sein von der Kommunikationsebene Hauptrechner-Komponenten. Die Synchronisation der verschiedenen Komponenten (K) erfolgt durch Ereignisflaggen.

1.2 Randbedingungen

Bei der Kommunikation zwischen Hauptrechner und Komponenten werden Mikrorechner, die als Zentraleinheit den Mikroprozessor INTEL 8o8o enthalten, eingesetzt; der Hauptrechner ist eine PDP11/4o. Die PDP arbeitet unter dem Betriebssystem RT11.

2. Kommunikation zwischen Hauptrechner und einer Komponente

Der Datenaustausch zwischen MR und HR des Analysenstandes erfolgt über ein serielles Interface. In Richtung HR sind Meßdaten, Status- und Alarmmeldungen und Parameterwerte zu übertragen, in Richtung Komponente Steuerinformationen und Parameter.
Für die Datenübertragung wurde das TTY kompatible ASCII-Format gewählt; der Datenaustausch zwischen HR und K ist somit auf einem Druckgerät unmittelbar protokollierbar. Diese Betriebsart ist während der Programmentwicklung, der Testphase und der Inbetriebnahme von Bedeutung: eine Komponente kann durch eine Teletyp simuliert werden.
Daten- und Befehlsaustausch erfolgen auf der Interruptebene.
Vom HR zu sendende Befehle werden zeichenweise an die Komponente übertragen. Jedes Zeichen wird von der Komponente zurück an den HR geechot, um dort auf Gleichheit überprüft zu werden. Im Fehlerfall wird der schon gesendete Befehlsteil verworfen und der vollständige Befehl neu übertragen.
Der allgemeine Befehlsaufbau setzt sich aus zwei Befehlsteilen zusammen: Im Teil 1 wird an die Komponente Information übertragen (Auftrag), im Teil 2 werden von der Komponente an den Hauptrechner Daten übermittelt.

2.1 Befehle vom Hauptrechner an die Komponente

Aufgabe des Hauptrechners ist es

- Daten zu lesen
- Parameter zu lesen
- Statusinformation zu lesen
- Steuerinformation zu schreiben
- Parameter zu schreiben.

Jede Komponente besitzt ein Statusfeld und ein Feld zur Aufnahme der Steuerinformation, das Controlfeld. Status- und

Controlfeld sind in ihrer Länge variabel. Die jeweilige Länge wird im führenden Byte des Feldes abgelegt (Abb. 1). Status- und Controlfeld sind byteorientiert.
Jede Komponente besitzt ein Datenfeld und ein Parameterfeld. Das Datenfeld ist wortweise organisiert, das Parameterfeld doppelwortweise (Abb. 2, Abb. 3). Das erste Wort des jeweiligen Feldes enthält die Anzahl der Daten bzw. Parameter des jeweiligen Feldes. Das Doppelwort des Parameterfeldes enthält für einen Parameter den aktuellen Parameterwert und den Anfangswert. Parametermodifikationen können durch Vergleich beider Teilworte erkannt werden. Status-, Control-, Daten- und Parameterfelder sind lokale Felder des Mikrorechners einer Komponente.

Grundlage für den Informationsaustausch zwischen HR und K sind fünf Basisbefehle.

2.1.1 Lesen von Daten

RD, w_1	Teil 1
w_3	Teil 2

Das durch "w_1" spezifizierte Datenwort wird ausgelesen. Der numerische Wert des derart adressierten Feldelements wird in "w_3" abgelegt und an den HR übertragen.

2.1.2 Lesen von Parametern

RP, w_1	Teil 1
w_3, w_4	Teil 2

Der durch "w_1" spezifizierte Parameter wird ausgelesen. Der aktuelle Parameterwert wird als Teil 2 des Befehls an den HR übertragen (w_3). "w_4" enthält den Parameteranfangswert. Sind die Werte "w_3" und "w_4" identisch, so entfällt das Übertragen von "w_4".

2.1.3 Schreiben eines Parameters

WP, w_1, w_2	Teil 1
entfällt	Teil 2

Der Hauptrechner überträgt an die Komponente den aktuellen Wert "w_2". Der zu modifizierende Parameter wird durch seine laufende Nummer "w_1" ausgewählt.

2.1.4 Schreiben von Steuerinformation

WC, w_1	Teil 1
entfällt	Teil 2

Das lokale Controlfeld einer Komponente wird durch die Maske "w_1" überschrieben. Jedem Bit in der Maske ist genau eine Komponentenfunktion zugeordnet; diese Funktion wird angestoßen.

2.1.5 Lesen des Komponentenstatus

RS	Teil 1
w_3	Teil 2

Der Hauptrechner liest den aktuellen Komponentenstatus. Das gelesene Statusfeld wird in "w_3" an den HR übertragen.

2.2 Alarme von der Komponente an den Hauptrechner

Wesentlich für die Steuerung komplexer Analysenstände ist, daß eine Komponente zu ihrer Prozeßperipherie hin große, zum Hauptrechner jedoch nur sehr geringe Eigenaktivität entfaltet. Diese Komponenteneigenschaft spiegelt sich auch im Befehlsvorrat wider; nur ein Befehl "LOOK AT ME" erlaubt der Komponente eigeninitiativ mit dem HR zu kommunizieren.

L	Teil 1
w_3	Teil 2

Nach Abschluß einer Steuerfunktion (WC) meldet die Komponente dem HR den Funktionsabschluß. Im Teil 2 wird gleichzeitig das Statusfeld der Komponente an den HR übertragen. Die Statusinformation ist in ihrer Länge variabel (Abb. 1), fest jedoch ist die Bitzuordnung für die zwei führenden Bytes:

Byte Ø:	Bit O-7	Länge
Byte 1:	Bit O	Test
	1	Handbetrieb
	2	Teilautomatik
	3	Automatik
	4	Betriebswechsel
	5	frei
	6	Fehler
	7	Ready

Von allgemeiner Bedeutung ist somit Byte 1, Bit 6, das Error-Bit. Wurde eine Steuerfunktion nicht fehlerfrei abgeschlossen, so ist dieses Bit gesetzt; mit dem nächsten Control-Befehl (WC) wird es durch K gelöscht.

2.3 Fehlerdiagnose

Nach 2.2 existiert ein Fehlerbit. Zur weiteren Fehlerdiagnose wird das Datenwort mit der laufenden Nummer eins herangezogen (Abb. 4). Bit O bis 7 enthalten den individuellen Fehlercode; Bit 13 bis 14 klassifizieren die Fehlerart.

Bit 13 gesetzt:	Warnung
14	Reparabler Fehler
15	Irreparabler Fehler

Die unterschiedliche Fehlerklassifizierung erlaubt angemessene Reaktionen auf den Fehler (Handeingriff, Wiederholen, etc.)

3. Systemarchitektur für parallelen Komponentenbetrieb

Der beschriebene Basisbefehlssatz ermöglicht einen transparenten Daten- und Befehlsaustausch zwischen dem HR und einer Komponente. Ein Analysenstand wird allgemein nicht von einer Komponente geführt, sondern erfüllt seine Aufgabe erst durch das sinnvolle Zusammenarbeiten mehrerer Komponenten bzw. Mikrorechner. Zu unterscheiden sind zwei grundsätzliche Teilaufgaben: Synchrone und asynchrone Vorgänge. Von den bekannten Basisbefehlen dürfen die Befehle RD, RP, WP, RS asynchron vom HR gesendet werden, der Befehl WC jedoch (= schreibe Steuerinformation) wird durch den aktuellen Zustand der Komponenten erst sinnvoll.
Als synchronisierendes Element wird "das Ereignis" eingeführt. Ereignisse kennen die beiden Zustände "eingetroffen" und "nicht eingetroffen".

3.1 Ereignissteuerung

Ein Ereignis ist durch seine Eigenschaft und durch seinen Zustand gekennzeichnet. Jeder Komponente können verschiedene Ereignisse zugeordnet werden - die Abfrage, ob ein Ereignis eingetroffen ist, bezieht sich nur auf eine Komponente K(I).
Ein Auftrag an eine Komponente K(I) wird vom Zustand eines Ereignisses abhängig gemacht; ist das erwartete Ereignis noch nicht eingetroffen, so wird der Prozessablauf in anderen Komponenten ungestört fortgesetzt. Abb. 5 zeigt das zyklische Abarbeiten des Ereignismodels. Haben alle n Komponenten ihre unterschiedlichen Aufträge noch nicht abgeschlossen, so reduziert sich der dynamische Programmablauf auf eine Warteschleife.

3.1.1 Formalisierte Ereignisabfrage

Die bestimmenden Größen bei der Ereigniserkennung bzw. -abfrage sind die Art des Ereignisses und die Komponentenzuordnung. Diese Informationen werden einem Unterprogrammaufruf übergeben:

```
JSR R5, EREIG
.WORD  ADR
JMP    NEIN
```

Hierbei enthält ADR eine Verweisadresse auf die bestimmenden Parameter (Abb. 6). Ist das zu prüfende Ereignis erfüllt, so wird der dynamische Programmablauf sequentiell fortgesetzt. Ist das Ereignis noch nicht eingetroffen, so wird der Programmablauf mit der Zieladresse "NEIN" fortgesetzt.

3.1.2 Formalisierter Prozeßblock

Die in Abb. 5 enthaltenen Prozeßblöcke 1 bis n sind alle von gleicher Struktur. Allen Blöcken ist gemeinsam, daß der lokale Programmfluß unterbrochen wird, falls die Komponente mit der Ausführung eines Befehls beauftragt wurde; erst nach Ausführung des Befehls (Auftrag) wird der Prozeßblock sequentiell weiter bearbeitet. Die Komponenten-Verweilzeit jedoch ist durch andere Komponenten zu nutzen. Abb. 7 zeigt die Struktur eines Prozeßblockes: zu unterscheiden ist der Kopf, der Rumpf und das Ende des Blockes. Der Kopf bildet den logischen Start des Prozeßablaufs und wird zum Sprungverteiler, falls der Programmablauf durch Komponentenaufträge unterbrochen wurde. Der Rumpf des Prozeßblockes enthält die Beschreibung des Prozeßablaufs. Abhängig von der individuellen Problemstellung kann der Benutzer den Programmablauf "unterbrechbar" gestalten. Eine Unterbrechungsstelle wird definiert durch

- Beschreiben des Ereignisses und
- durch einen Unterprogrammaufruf "RAUS"

Der "JSR R5, RAUS" Aufruf bewirkt

- die dynamische Programmunterbrechung und
- Übernahme der dem RAUS-Aufruf folgenden Adresse als Wiedereintrittsstelle.

Die Anzahl der so definierten Unterbrechungsstellen ist beliebig.
Das logische Ende des Blockes wird durch einen "JSR R5, TAUSCH" Aufruf gekennzeichnet. Der Unterprogrammaufruf korrigiert die Registerstände und kennzeichnet den Prozeßblock als logisch beendet.

3.1.3 Formalisierte Komponentenbeschreibung

Faßt man nach 3.1.1 und 3.1.2 Ereignisabfrage und Blockbeschreibung zusammen, ergibt sich die Beschreibungsform für eine Komponente.
Charakteristisch für eine Komponentenbeschreibung sind also:

- eröffnende formalisierte Ereignisabfrage
- formalisierte Blockbeschreibung
- Blockkopf (REIN)
- Blockrumpf (Prozeßbeschreibung mit Unterbrechungsstellen)
- Blockende (Registerstände)

Die Verknüpfung mehrerer Komponentenbeschreibungen (mehrerer Prozeßabläufe) erfolgt durch zyklisches Abarbeiten (Abb. 8).

3.1.4 Verschiedene Ereignisformen

Ereignisse erteilen Auskunft über einen Prozeßzustand. Die möglichen Zustandsformen werden durch eine laufende Nummer NR

gekennzeichnet. Die nähere Spezifikation des zu überprüfenden Zustandes erfolgt durch Vorgabe eines Parameterfeldes PAR.

- das sichere Ereignis?

  NR=0; PAR=keine

Das sichere Ereignis ist stets erfüllt. Der Programmablauf einer Komponente wird unterbrochen, jedoch unabhängig von Prozeßzuständen beim nächsten Zyklusdurchlauf (Abb. 8) hinter der Unterbrechungsstelle fortgesetzt.

- steht ein LAM an? (Komponentenalarm)

  NR=1; PAR=KOMP

Wurde eine Komponente mit einer Steuerfunktion (WC) beauftragt, so wird das Ende der Ausführung durch einen Alarm (LAM) dem HR mitgeteilt. Ereignis Nr. 1 ist somit eingetroffen, wenn die initiierte Funktion beendet ist.

- ist die Übertragungsstrecke frei?

  NR=2; PAR=KOMP

Das Ereignis ist eingetroffen, falls eine Datenübertragung an die durch KOMP definierte Komponente abgeschlossen ist, bzw. falls eine neue Übertragung erlaubt ist.

- darf ein Auftrag mit bekanntem Funktionscode erteilt werden?

  NR=3; PAR=KOMP, OPCODE

Basisbefehle haben entweder Steuer/Controlcharakter oder sie ermöglichen den Datenaustausch zwischen HR und Komponenten. Diese beiden Befehlsarten unterscheiden sich grundsätzlich dadurch, daß bei Steuerbefehlen die Funktion nur angestoßen wird, bei Datenbefehlen beinhaltet die Befehlsübertragung den Funktionsabschluß. Ereignis Nr. 3 ist dann eingetroffen, wenn es physikalisch und logisch erlaubt ist, an die Komponente "KOMP" den Basisbefehl "OPCODE" zu übertragen.

- ist ein Auftrag mit bekanntem Funktionscode abgeschlossen?

  NR=4; PAR=KOMP, OPCODE

Das Ereignis Nr. 4 gilt als eingetroffen, falls ein Auftrag an die Komponente "KOMP" mit dem Operationscode "OPCODE" von der Komponente abgearbeitet ist.

- wurde ein Schreib/Leseauftrag erteilt?

  NR=5, PAR=KOMP

Schreib/Leseaufträge werden durch die Basisbefehle RD, RP, RS, WP realisiert. Aufträge dieser Art dürfen asynchron gesendet werden. Voraussetzung für die Befehlsübermittlung ist, daß die notwendigen Übertragungsstrecken frei sind. Das Ereignis Nr. 5 ist eingetroffen, wenn ein Auftrag dieser Art erteilt wurde, aber noch zur Bearbeitung ansteht.

- wurde ein Controlauftrag erteilt?

  NR=6; PAR=KOMP

Controlaufträge werden durch den Basisbefehl WC realisiert. Bei der Ausführung von Controlaufträgen sind grundsätzlich drei Verarbeitungsphasen zu unterscheiden:

  - die Auftragserteilung an das Steuersystem
  - die Befehlsübermittlung an die Komponente und
  - die Bestätigung der Funktionsausführung an den Hauptrechner.

Das Ereignis Nr. 6 ist eingetroffen, wenn ein WC-Auftrag erteilt wurde.

- wurde ein Auftrag erteilt, übermittelt und dessen Funktionsabschluß bestätigt?

  NR=7; PAR=Parameterbeschreibung

Die Parameterbeschreibung kennzeichnet genau die Art des zu übertragenen Befehls. Die typischen Bestimmungsgrößen sind:

  X: interner Schalter. Bei Aufruf X=O setzen

  KOMP: Komponentennummer

OPCODE: Operationscode
Wert1: Nummer des Datenwortes
Wert2: Wert des Datenwortes
FELD: Wert des Controlfeldes
KSEND: Antwort der Komponente
ERROR: Fehlerwort der Komponente

Das Ereignis Nr. 7 gilt dann als eingetroffen, wenn der durch die Parameterbeschreibung definierte Befehl an die Komponente übertragen wurde, und die somit initiierte Funktion abgeschlossen ist. Zwischen Daten- und Steuerbefehlen wird nicht unterschieden. Bei Datenbefehlen ist das Ereignis eingetroffen, wenn Teil 2 (siehe Punkt 2.1) an den Hauptrechner übertragen wurde, bei Steuerbefehlen (WC) bewirkt der Komponentenalarm (LAM) das Ereignis.

- Ist eine Ereignisflagge gesetzt?

  NR=8; PAR=FLAGNR

Die Synchronisation der Prozeßabläufe wird durch ein globales Flaggenfeld gesteuert. Das Flaggenfeld ist byteorientiert; jede Flagge wird durch Angabe ihrer laufenden Nummer (0, 1, 2,....) adressiert (Abb. 9). Eine Flagge kann den Zustand gesetzt (=1) oder gelöscht (=0) annehmen.
Das Ereignis Nr. 8 ist eingetroffen, falls die Flagge mit der laufenden Nummer FLAGNR gesetzt ist.

- Ist eine Ereignisflagge gelöscht?

  NR=9; PAR=FLAGNR

Das Ereignis Nr. 9 ist eingetroffen, falls die entsprechende Flagge glöscht ist (Abb. 1o).

- wurde eine Komponente gesperrt?

  NR=10; PAR=KOMP

Die Ausführung von Steuerbefehlen (WC) kann temporär gesperrt werden. Diese Blockade ist dynamisch durch den Programmablauf möglich oder auch durch den Prozeßdialog. Aufgehoben werden kann die Blockade nur durch den Prozeßdialog.

Das Ereignis Nr. 1o ist eingetroffen, falls die Komponente KOMP blockiert ist, d.h. es dürfen keine Steuerbefehle abgesandt werden.

3.2 Komponentensteuerung

Die Komponentensteuerung ist Teil des steuernden Systems; sie ist als Systemprogramm unabhängig von der aktuellen Prozeßbeschreibung. Die Komponentensteuerung führt Aufträge (Basisbefehle) an eine Komponente aus. Aufträge der Gruppe 1 (RD, RP, WP, RS) und der Gruppe 2 (WC) werden parallel bearbeitet.
Besteht der Prozeßablauf aus mehreren Komponenten, so muß für jede der Komponenten (Mikrorechner) eine Komponentensteuerung (KST) in das System implementiert werden.
Enthält ein System mehrere KST, so sind alle KST's identisch mit Ausnahme der zwangsläufig unterschiedlichen Interfaceadressen.

3.2.1 Auftragsvergabe

Ein Auftrag an eine KST enthält die bestimmenden Größen des Befehls, den die KST an ihre Komponente zu übertragen hat. Diese bestimmenden Größen sind in einem Parameterblock formal zusammengefaßt.

- KOMP Komponentennummer der Zielkomponente
- OPCODE Operationscode des Basisbefehls
  - 1 = RD
  - 2 = RS
  - 3 = WC
  - 4 = WP
  - 5 = RP

- WERT1 Nummer des Datenwortes
- WERT2 Wert des Datenwortes
- FELD Wert des Controlfeldes
- KSEND Antwortdaten der Komponente
- ERROR Fehlercode bei WC-Befehlen

Der Parameterblock enthält jeweils die Adressen der Parameter. Falls die Angabe eines Parameters nicht für die Befehlsbeschreibung erforderlich ist, muß der entsprechende Platz im Parameterblock freigehalten werden (z.B. der Parameter FELD wird bei dem Basisbefehl RD nicht benötigt).

Ein globales Auftragsfeld AUFELD beschreibt den jeweils aktuellen Komponentenzustand. Jeder Komponente ist ein Doppelwort zugeordnet; das niederwertige Wort enthält die Adresse des Auftragsblokkes für Datenbefehle (RD, RP,...), daß höherwertige Wort den Verweis auf den Auftragsblock für Steuerbefehle.
Wurde der Komponente kein Auftrag erteilt, bleibt das entsprechende Wort im Auftragsfeld leer (Abb. 11).
Das Eintragen der notwendigen Verweisadressen in das Auftragsfeld erfolgt implizit durch den Ereignismechanismus (siehe Ereignis Nr. 7).

3.2.2 Abarbeiten von Aufträgen der Gruppe 1

Die Komponentensteuerung baut aus den Parametern des Auftragsblockes den gewünschten Befehl auf. Nach dem Eintreffen des Ereignisses Nr. 2 (ist die Übertragungsstrecke an die Komponente K frei?) wird der in Auftrag gegebene Befehl übertragen. Die Befehlsübertragung wird auf Übertragungsfehler überprüft. Im Fehlerfall wird die Befehlsübertragung neu initiiert und der gleiche Befehl erneut übertragen.
Die Fehlererkennung durch das lokale Unterprogramm TSTUEB erfaßt die Anzahl der fehlerhaften Übertragungsversuche. Nach dem Überschreiten einer vorgegebenen Versuchszahl n, wird die Fehlermeldung "Transmit-Error" mit Angabe der aktuellen Komponente ausgedruckt.

Nach erfolgreicher Befehlsübertragung wird die Antwort der Komponente (Daten, Parameter) im globalen Feld KSEND abgelegt; sie steht dem Benutzer somit zur weiteren Bearbeitung zur Verfügung. Der Auftrag wird im Auftragsfeld gelöscht.

3.2.3 Abarbeiten von Aufträgen Gruppe 2

Der Aufbau des Befehls und die physikalische Übertragung erfolgen identisch zur Gruppe 1. Zusätzlich jedoch werden die Alarmmeldungen der Komponente erfaßt. Falls die Befehlsübertragung fehlerfrei abgeschlossen wurde, wartet die Komponentensteuerung auf das Ereignis Nr. 1 (LAM). Das gleichzeitig mit dem LAM übermittelte Statusfeld wird analysiert. Die Interpretation des Statusfeldes durch die KST erteilt Auskunft über den Fehlerzustand der ausgeführten Funktion. Im fehlerfreien Fall gilt der Auftrag als ausgeführt und er wird durch die KST im Auftragsfeld ausgetragen. Wurde die Funktion nicht fehlerfrei ausgeführt, so werden typische Daten wie Datum, Uhrzeit und Fehlercode als Fehlermeldung ausgedruckt. Die Komponente wird für weitere WC-Funktionen gesperrt. Die Komponentensperre kann nur durch den Operateur über die Dialogeingabe aufgehoben werden.

3.2.4 Blockieren von Komponentensteuerungen

Die Beschreibung des logischen Prozeßablaufes erfolgt in einem vom Benutzer erstellten Prozeßmodul. Allgemein wird der Benutzer für jede selbständige Komponente auch ein selbständiges Modul erstellen. Die Synchronisation erfolgt nach 3.14 über Ereignisse. Diese Modulerstellung berücksichtigt nicht physikalische Randbedingungen der Übertragungsvorgänge; diese physikalischen Forderungen werden zentral durch die Komponentensteuerung erfüllt. Das Sperren bzw. das Freigeben der Komponentensteuerung bewirkt somit den physikalischen HALT oder CONTINUE des Prozeßmoduls.

Nach 3.2.3 blockiert sich die KST selbst, falls Steuerfunktionen fehlerhaft abgeschlossen wurden. Die KST führt solange keine Aufträge der Gruppe 2 (WC) mehr aus, bis der Operateur dies per Dialog erlaubt. Abhängig von den Ereigniszuständen wird der Prozeßablauf der anderen Komponenten nicht beeinträchtigt. Die Komponentenblockade wird intern durch das globale Feld KBLOCK realisiert (Abb. 12). Dieses Feld enthält für jede Komponente einen Merker, der den Blockadezustand (gesperrt, frei) beschreibt. Die Länge des Feldes ist somit abhängig von der Anzahl der Komponenten. Sowohl das Sperren als auch das Freigeben einer Komponente ist dialogorientiert möglich:

* UNLOCK, Ki ∉	Freigeben Ki
* LOCK, Ki ∉	Sperren Ki
	(i=1, 2, 3,...N)

3.3 Operateurbefehle

Die prozeßbeschreibende Befehlsfolge ist in verschiedenen Prozeßmoduln zusammengefaßt. Die Befehlsfolge wird durch die Komponentensteuerung zentral überwacht. Befehle, die unabhängig vom typischen Prozeßablauf sind, werden als Operateurbefehle bezeichnet. Ihrem Charakter entsprechend haben sie Monitorfunktion; sie dienen der Überwachung des Prozeßablaufes und ermöglichen Eingriffe in den funktionellen Ablauf. Diese Befehle werden nicht von den Komponentensteuerungen erfaßt.

- Zwangsabbruch	*STOP ∉
- Blockade	*LOCK, Ki ∉
- Freigabe	*UNLOCK, Ki ∉
- Uhrzeit	*TIM ∉
- Datum	*DAT ∉
- Ereignisflagge setzen	*SETF, Nr ∉
- Ereignisflagge löschen	*CLRF, Nr ∉

- Lese Daten *RD, Ki, Nr ⊄
- Lese Parameter *RP, Ki, Nr ⊄
- Lese Status *RS, Ki ⊄
- Schreibe Parameter *WP, Ki, Nr, Wert ⊄
- Schreibe Control *WC, Ki, Wert ⊄
- Liste alle Parameter (Symbolisch) *LISTP ⊄
- Liste den Parameter "name" (Symbolisch) *LISTP, name ⊄

Die Eingabe der Operateurbefehle erfolgt über das Systemeingabeschreibwerk.

3.3.1 Definition neuer Operateurbefehle

Die oben aufgeführten Befehle sind von allgemeiner Bedeutung. Den unterschiedlichen Anwendungsfällen entsprechend ist die Notwendigkeit neue, prozeßtypische Operateur-Befehle zu definieren, wahrscheinlich. (Gerätefreigabe, Normierungsvorgänge, speziellen Steuerfunktionen symbolische Namen zuordnen,...). Bei der Definition neuer Befehle sind einige Konventionen zu beachten:

- Befehlstabelle BEFTAB

Ein neuer Befehl wird mit seinem symbolischen Namen in die BEFTAB aufgenommen. Die Befehlsbeschreibung enthält die Anzahl und Art der individuellen Parameter.
Der Modifikationsvorgang der Befehlstabelle wird durch einen Makroaufruf auf die Namensdefinition und Parameterklassifizierung reduziert.

3.3.2 Vergabe symbolischer Parameternamen

Nach Punkt 2.1 wird ein Komponentenparameter durch seine laufende Nummer im Parameterfeld angesprochen. In vielen Anwendungsfällen ist die symbolische Beschreibung des Parameter sinnvoll bzw. notwendig. Die Zuordnung eines symbolischen Namens zu einem Parameter ist in transparenter Form durch Modifikation der Systemtabelle PARTAB möglich:

- Parametertabelle PARTAB

Die kennzeichnenden Größen eines Parameters sind seine Zielkomponente und seine laufende Nummer. Ein Makro reduziert den Modifikationsaufwand zu einem Macro-CALL mit den Argumenten Name, Komponente und Nummer.

4. Prozeßbeschreibung

Nach Punkt 3. existiert im allgemeinen für jede Komponente eine Ereignissteuerung und eine Komponentensteuerung. Die Ereignissteuerung beschreibt den Prozeßablauf und enthält synchronisierende Elemente zu anderen Komponenten. Die Komponentensteuerung ist prozeßunabhängig und einheitlich für jede Komponente.
Abb. 13 zeigt die Struktur einer Prozeßbeschreibung, hier für zwei Komponenten. Die Anzahl der Komponenten bestimmt gleichzeitig die Anzahl der dynamisch zu durchlaufenden Komponentensteuerungen KSTO1 und KSTO2. Die Funktionsblöcke PROCESS1 und PROCESS2 enthalten die komponententypischen Prozeßbeschreibungen, unterstützt durch Elemente der Ereignissteuerung. Die Operateurbefehle werden im Block SPRVTL (Sprungverteiler) erfaßt und bearbeitet.
Die Kette Prozeßsteuerung - Komponentensteuerung - Operateurbefehle wird zyklisch abgearbeitet. Liegen keine Prozeßaufträge und keine Operateurbefehle an, so reduziert sich der Zyklus auf eine Warteschleife. Diese Warteschleife wird durch das Eintreffen eines Ereignisses unterbrochen und somit die Prozeßbearbeitung fortgesetzt.

Das einleitende INIT-Modul hat die übliche Aufgabe, Anfangsbedingungen zu setzen, Geräte zu eröffnen und die Übertragungsstrecken für die Komponentenkommunikation aufzubereiten.

Der statische Vereinbarungsteil der Prozeßbeschreibung enthält u.a. Platzhalter für:

- das Ereignisflaggenfeld FLAG
- das Blockadefeld KBLOCK
- das Auftragsfeld AUFFLD
- das Statusfeld KSTAT
- das Antwortfeld KSEND
- das Alarmfeld LKOMP

Die Länge dieser Felder ist anwendungstypisch; die Anzahl der Komponenten wirkt sich als Faktor auf die Gesamtlänge aus. Die Länge des Flaggenfeldes ist abhängig von der Anzahl der gewünschten Flaggen.

5. Anwendungen und geplante Einsätze

Das hier vorgestellte System wurde im praktischen Einsatz angewandt und erfolgreich erprobt |2|, |3|.

5.1 Alpha-Meßplatz

Der Alpha-Meßplatz wird als komplexer Analysenstand von einem Hauptrechner PDP11/4o zentral gesteuert. Er enthält folgende Funktionseinheiten:

- Meßelektronik
- Spektrendisplay
- Protokollschreibwerk
- Probenwechsler
- Probenflaschenwechsler
- Alphapräparation (Abb. 14)

Durch Mikrorechnersysteme werden Probenflaschenwechsler und Alphapräparation unterstützt. Die anderen Komponenten werden durch CAMAC gesteuert. Alle sechs Funktionseinheiten werden durch das hier vorgestellte System synchronisiert; der Daten- und Befehlsaustausch zur 8o8o-Schnittstelle ist nach Punkt 2. und 3. im System implementiert. Die Programmierarbeiten am Alpha-Meßplatz sind erfolgreich abgeschlossen.

5.2 Gamma-Meßplatz

Der Gammameßplatz ist von ähnlicher Struktur wie der Alphameß-platz (Abb. 14, 15). Trotz der unterschiedlichen Aufgabenstellung wird auch dieser Meßplatz durch das gleiche System verwaltet. Die Unterschiede der beiden Systeme werden bei der Systemgenerierung berücksichtigt:

- nur eine Komponentensteuerung
- modifizierter Prozeßdialog
- unterschiedliche Device-Adressen.

Die Systemgenerierung erfolgt auf der Ebene des Linkage-Editor's. Die Programmierarbeiten am Gammameßplatz sind erfolgreich abgeschlossen.

5.3 Nachfolgelösungen

Im Rahmen der CALAS-Nachfolge wird für das Institut für Technische Physik ein Programmkonzept entwickelt. Dieser Programmentwurf berücksichtigt einen Dilatationsmeßplatz und vier weitere Meßapparaturen, zentral verwaltet durch das hier vorgestellte System.
Nach Punkt 5.1 befindet sich ein Alpha-Meßplatz im Einsatz.
Ein weiteres Projekt - Alphameßplatz 2 - befindet sich in der Realisierungsphase. Dieses Projekt unterscheidet sich von seinem

Vorgänger durch Modifikationen des mechanischen Aufbaus. Gleichzeitig wird durch eine verbesserte Komponentenzusammensetzung der Probendurchsatz erhöht.
Das unterstützende System bleibt unverändert.

6. Anpassungsmöglichkeiten des Systems

Nach Punkt 5. wurde das System im praktischen Einsatz angewandt. Die genannten Anwendungsfälle haben als gemeinsame Eigenschaften

- unkritische Übertragungsrate
- ausreichender Kernspeichervorrat
- kurze Übertragungsstrecken
- individueller Prozeßdialog (Completion-Routine)
- kein Datenblocktransfer

Diese typischen Merkmale wurden bei der Systemkonzeption berücksichtigt. Eine mögliche Systemoptimierung kann sich an diesen Eigenschaften orientieren, also:

- erhöhen der Übertragungsrate
- Reduktion des Kernspeicherbedarfs
- Verbesserung der Fehlererkennung und Beseitigung auf den Übertragungsstrecken
- Modularisierung des Prozeßdialoges.

Literatur

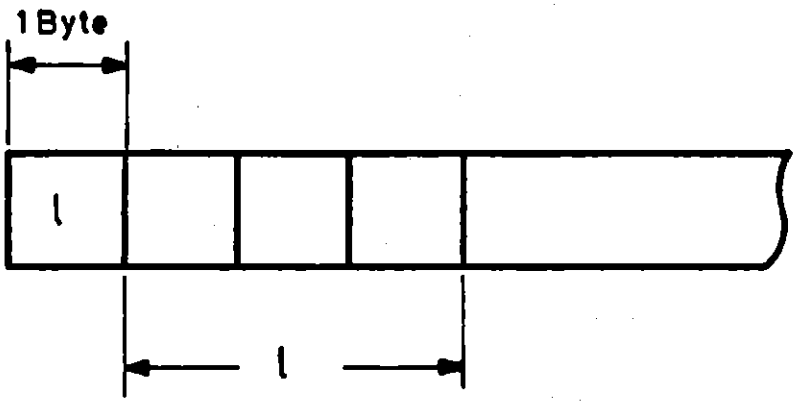
|1| G. Gütle, W. Heep, S. Radek
Unveröffentlichter Bericht des KfK, 1976.

|2| J. Augenstein, W. Heep, R. Kaufmann, K-D. Rusch, H-P. Zinecker
Unveröffentlichter Bericht des KfK, 1981.

|3| J. Augenstein, K.-D. Rusch, F. Süß
Unveröffentlichter Bericht des KfK, 1981.

Abkürzungen und Symbole

MR	Mikrorechner
HR	Hauptrechner
K	Komponente
K(I)	Komponente mit der lfd. Nr. I
P(I, F)	Funktion F, der Komponente I
TTY	Teletyp
CRLF	Wagenrücklauf, Zeilentransport (ASCII-Zeichen)
RD	Basisbefehl: Lese Daten
RP	" Lese Parameter
RS	" Lese Statusfeld
WP	" Schreib Parameter
WC	" Schreib Control/Steuerinformation
L	" Look at me, Komponentenalarm
KOMP	Ereignisparameter: Komponentennummer
OPCODE	" Operationscode eines Basisbefehls
KST	Komponentensteuerung



l = Länge des Feldes , hier l = 3 Byte

Abb. 1 Statusfeld und Controlfeld der Komponente

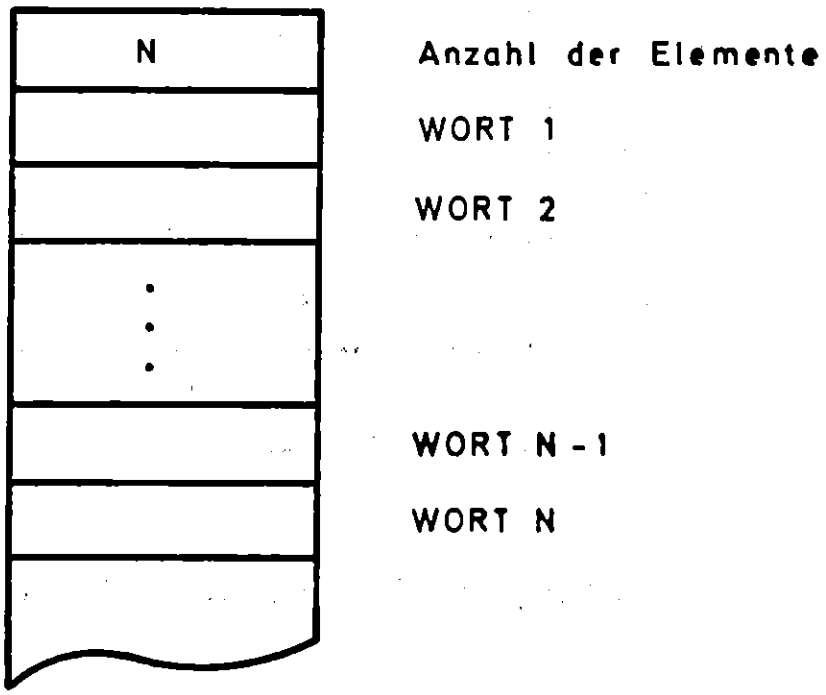


Abb. 2 Datenfeld der Komponente

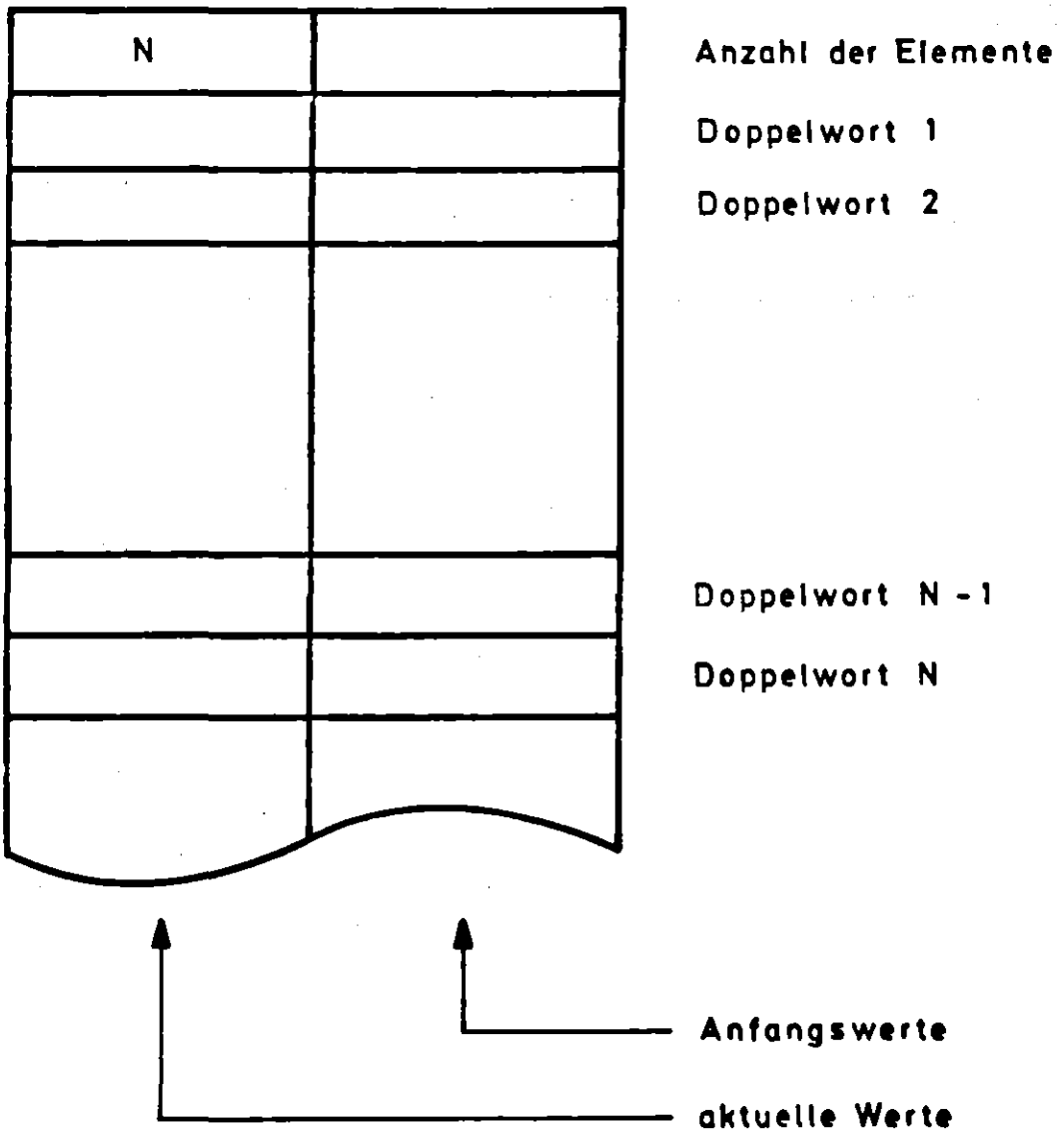


Abb. 3 Parameterfeld der Komponente

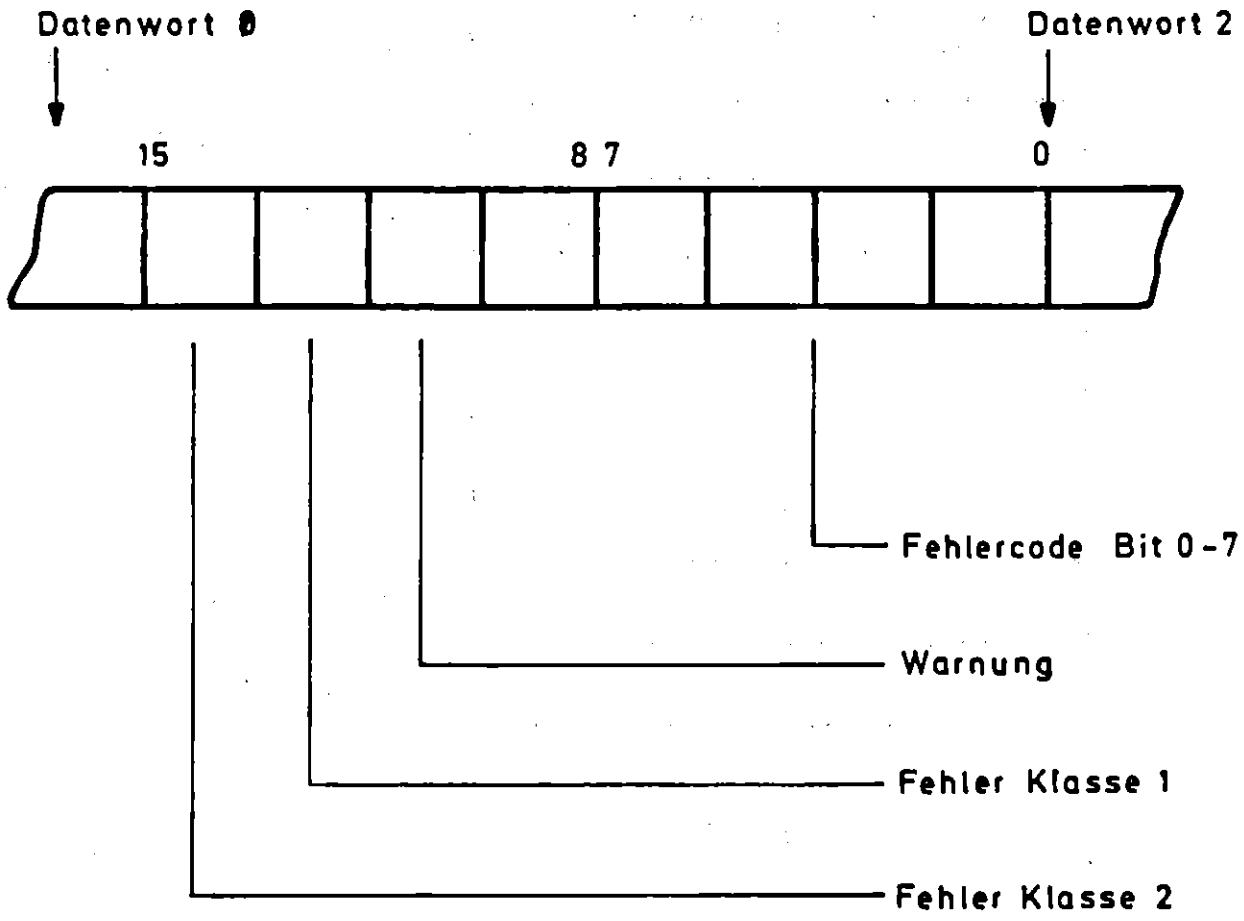


Abb. 4 Das Datenwort 1

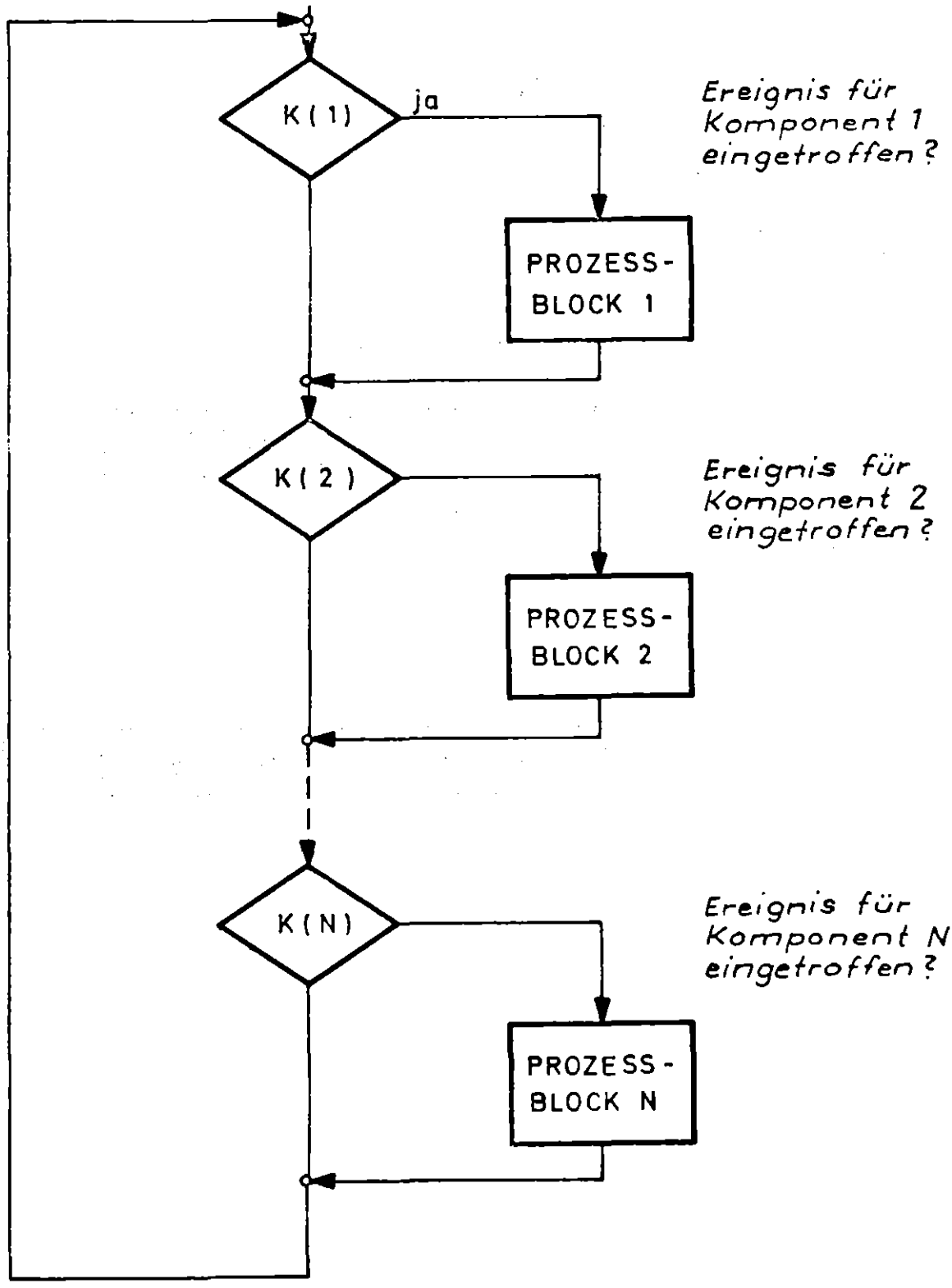


Abb. 5 Modell der Ereignissteuerung

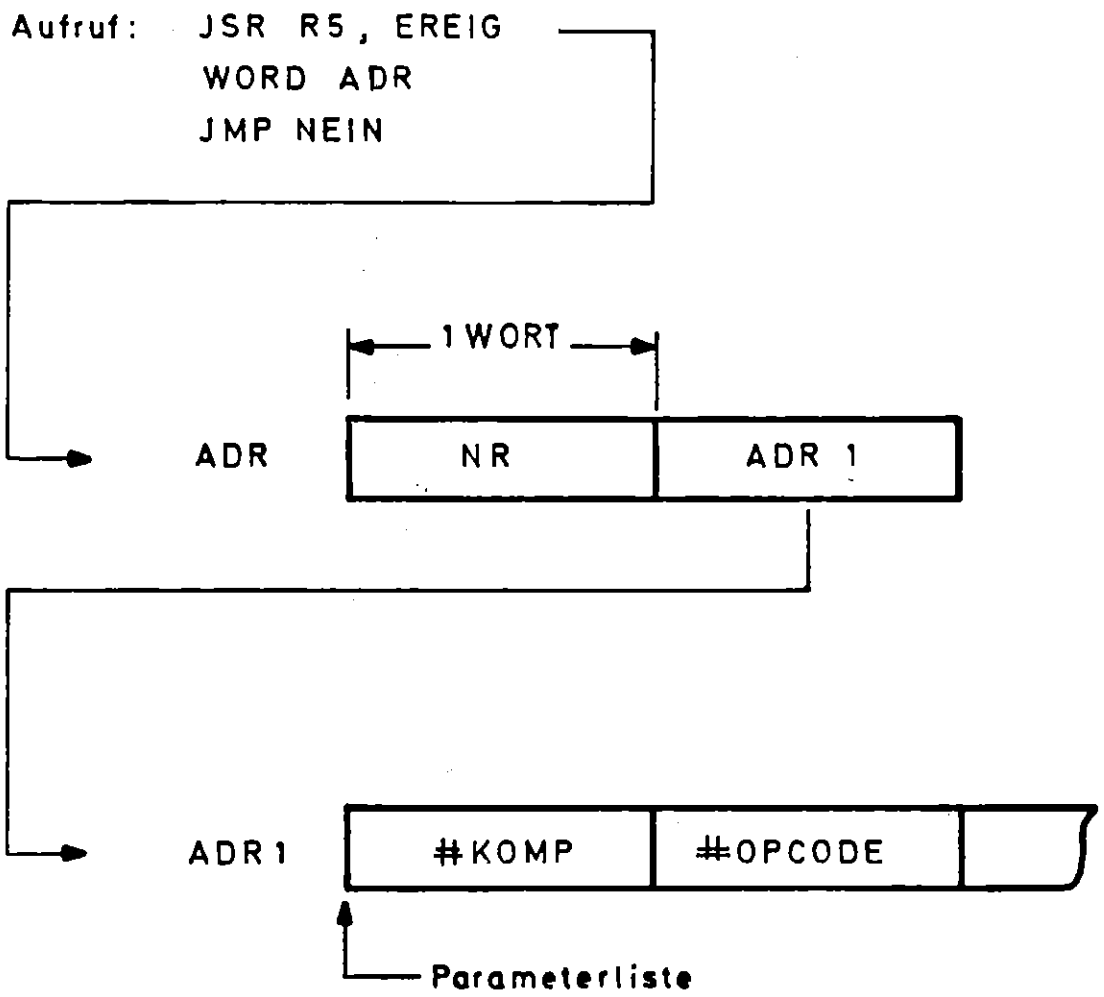


Abb. 6 Parameterübernahme

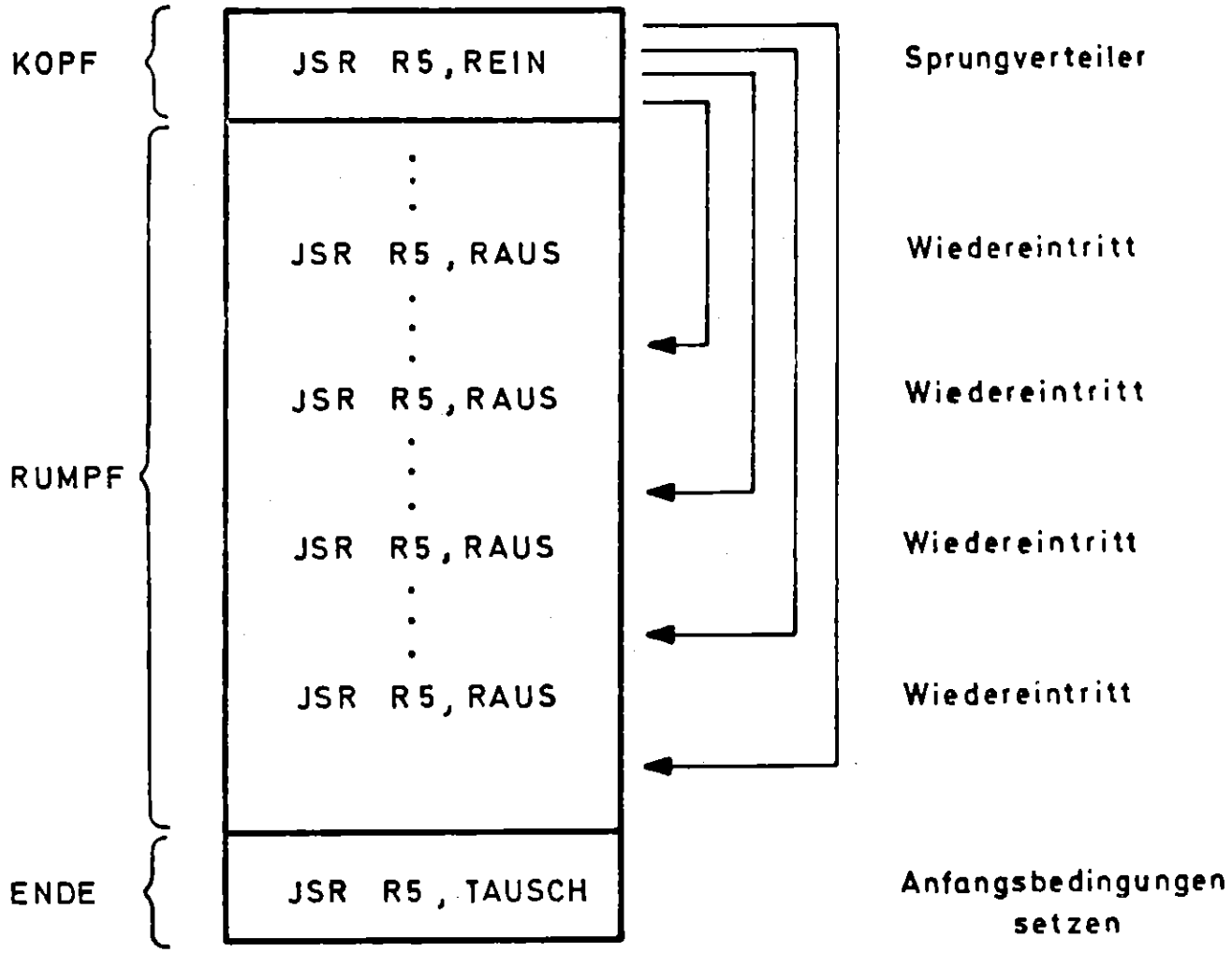


Abb. 7 Struktur des Prozeßblocks

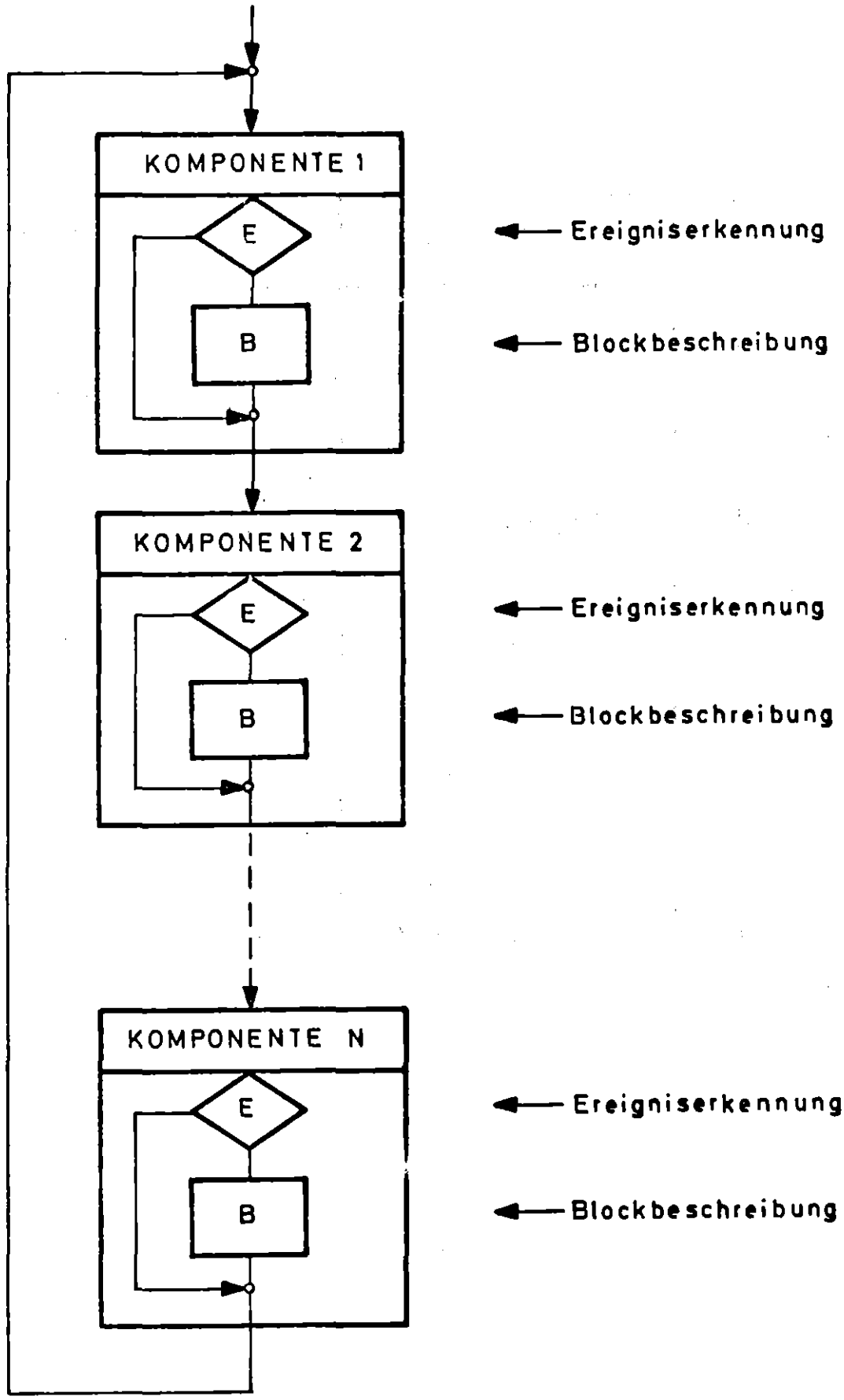
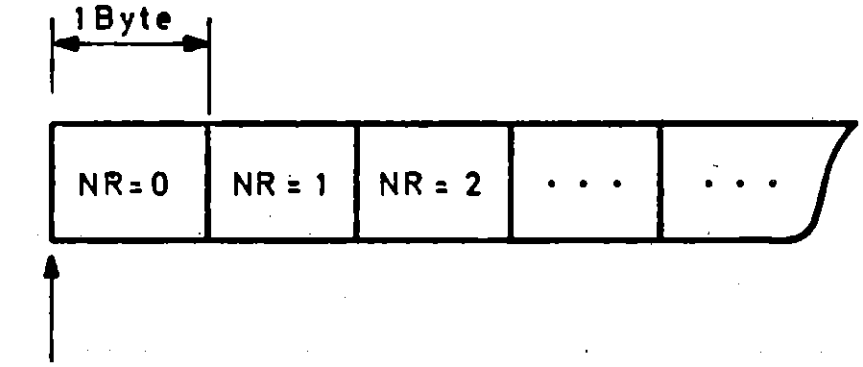


Abb. 8 Komponentenverknüpfung

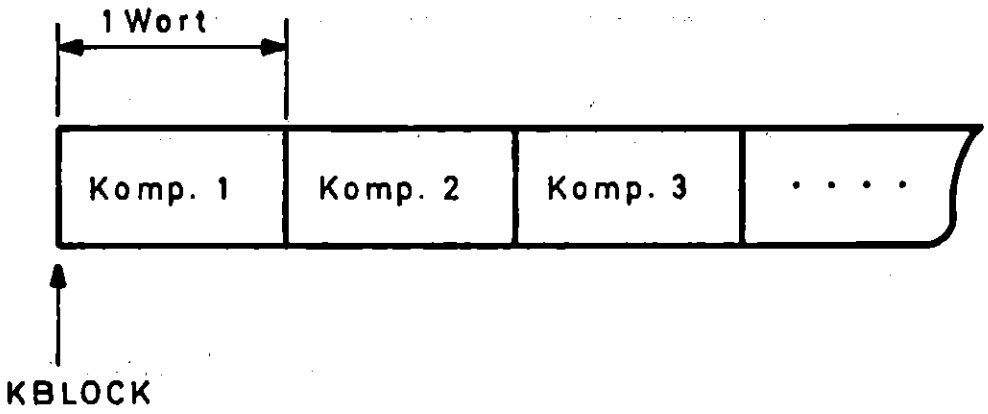


FLAG = globale Marke

<FLAG + K> = 0 Flagge K gelöscht

<FLAG + K> = 1 Flagge K gesetzt

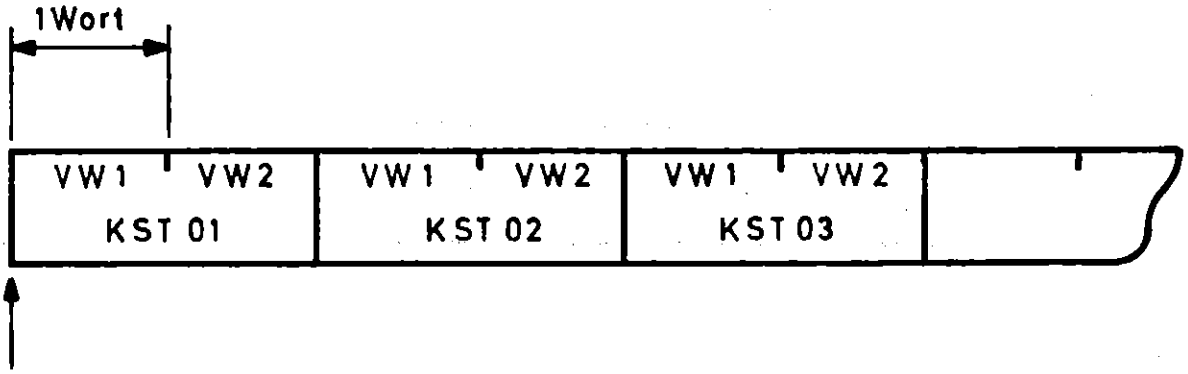
Abb. 9 Das Flaggenfeld



<KBLOCK + K · 2> = 0 Komponente K frei

<KBLOCK + K · 2> = 1 Komponente K blockiert

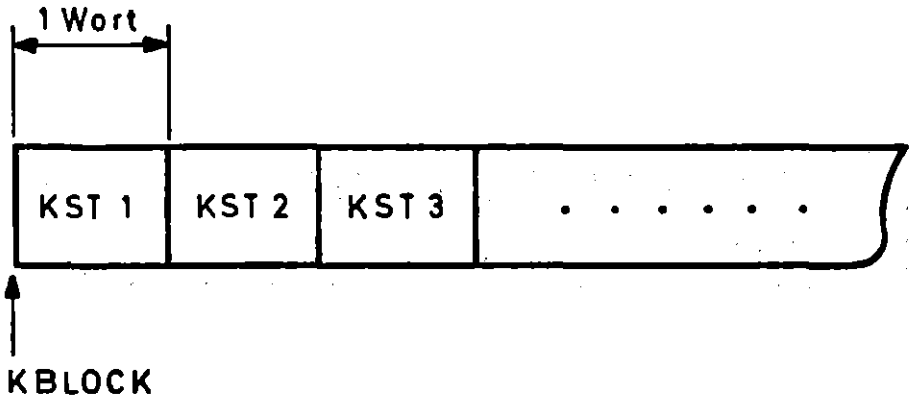
Abb. 10 Komponentenblockade



<VW 1> = Verweis auf Aufträge der Gruppe 1

<VW 2> = Verweis auf Aufträge der Gruppe 2

Abb. 11 Das Auftragsfeld



<KSTi> = 0 : Komponente ist frei

<KSTi> = 1 : Komponente ist gesperrt

Abb. 12 Die Komponentenblockade

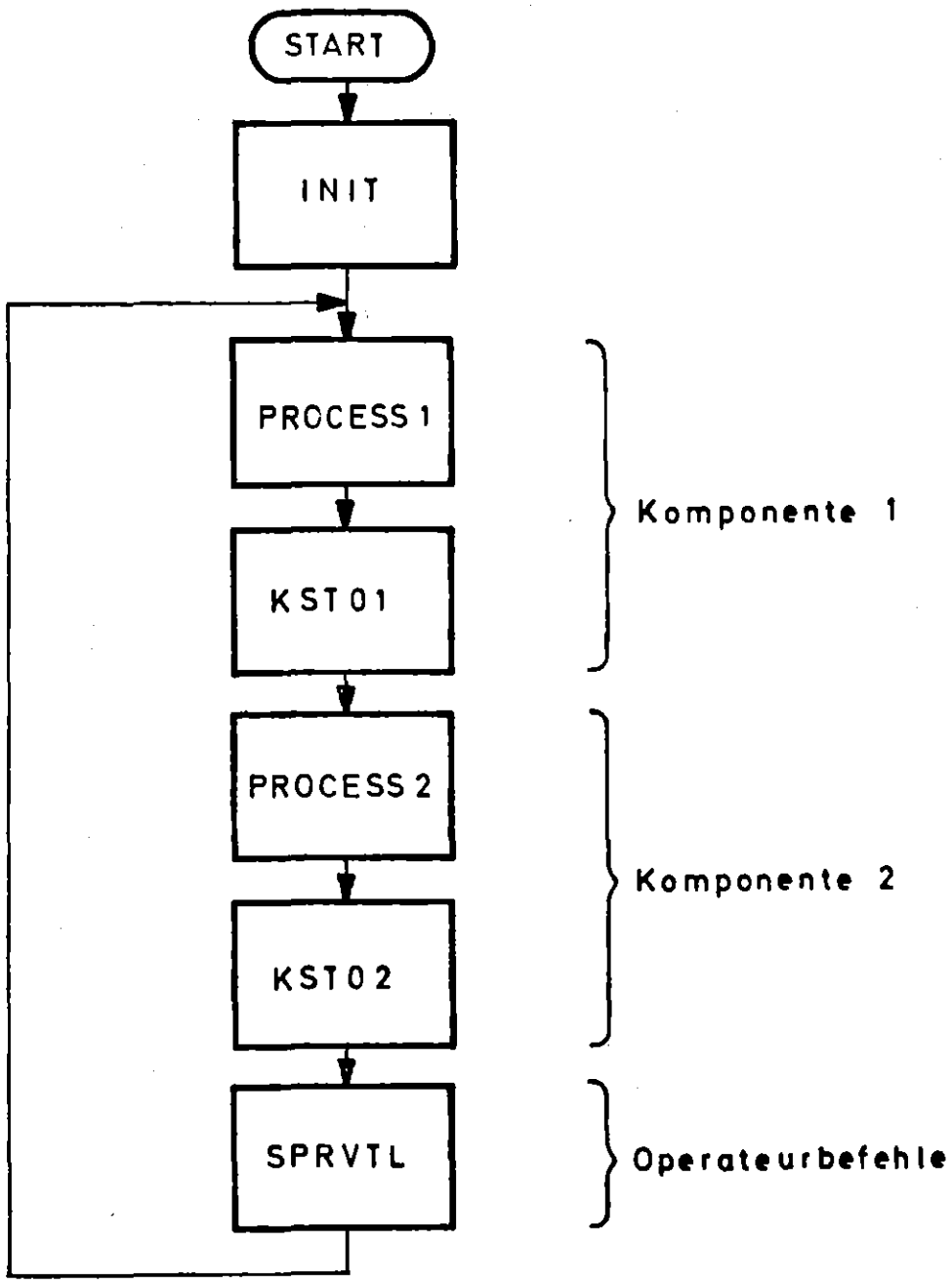


Abb. 13 Struktur des Hauptprogramms

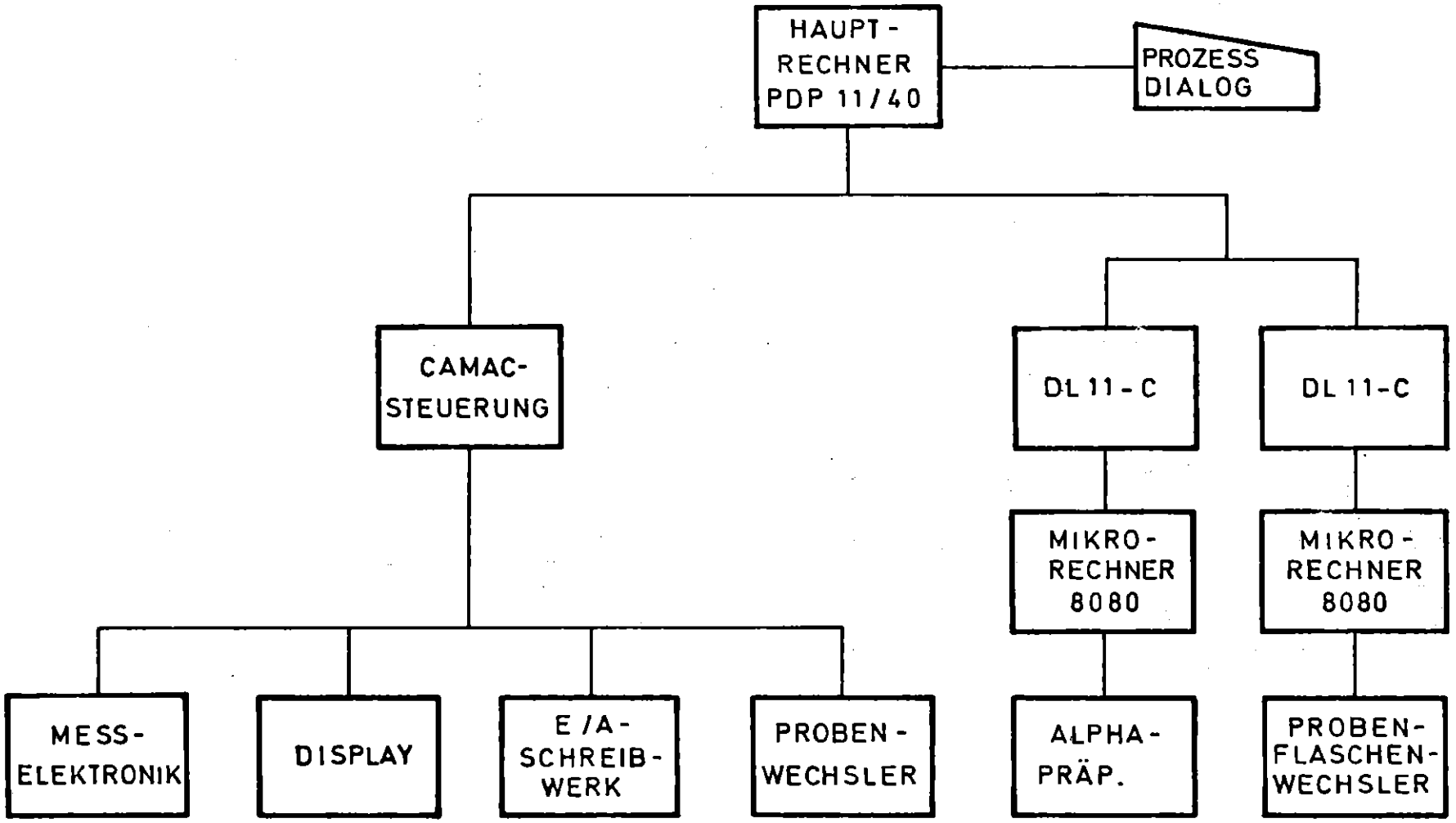


Abb. 14 Der ALPHA - Meßplatz

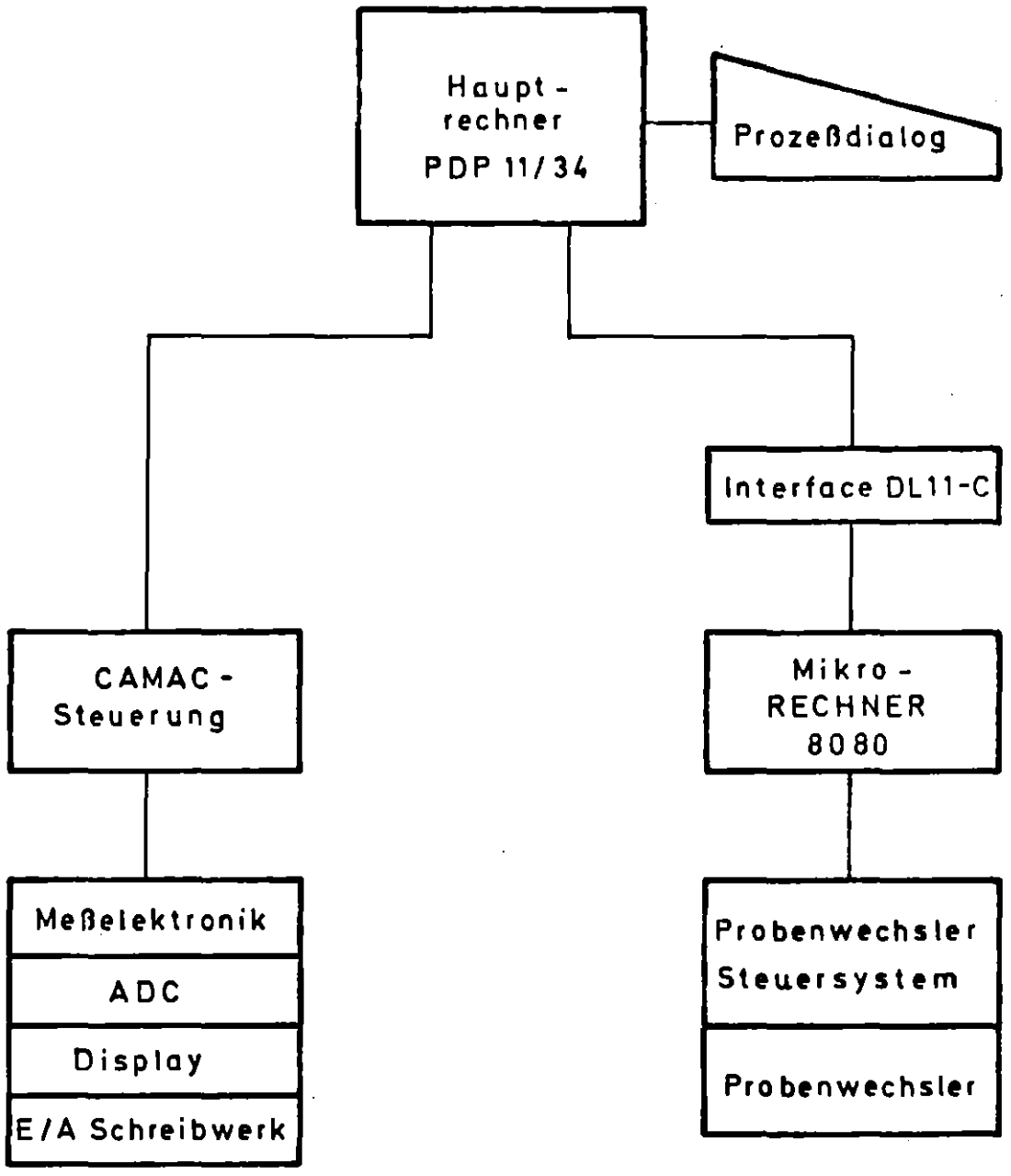


Abb. 15 Der Gamma - Meßplatz